AZƏRBAYCAN RESPUBLİKASI TƏHSİL NAZİRLİYİ
BAKI DÖVLƏT UNİVERSİTETİ

AYDIN YUNUS OĞLU ƏLİYEV

İNFORMATİKA VƏ PROQRAMLAŞDIRMA

Ali məktəblər üçün dərs vəsaiti

Azərbaycan Respublikası Təhsil Nazirliyi tərəfindən təsdiq edilmişdir (əmr № 289, 03 mart 2008-ci il).

Bakı – Mütərcim – 2008

UOT 519.682

Elmi redaktor :BDU-nun «İnformasiya texnologiyaları və proqramlaşdırma» kafedrasının müdiri, t.e.d, professor Ə.Ə. Əliyev

Rəyçilər :BDU-nun «Hesablama riyaziyyatı» kafedrasının professoru, f.-r.e.d. V.R.İbrahimov
AMEA-nın Kibernetika İnstitutunun bölmə müdiri, f.-r.e.d. K.Ş. Məmmədov

Aydın Yunus oğlu Əliyev. İnformatika və proqramlaşdırma. Ali məktəblər üçün dərs vəsaiti. Bakı, Mütərcim, 2008, 404 səh.

BDU-nun «Hesablama riyaziyyatı» kafedrasının dosenti, f.r.e.n A.Y.Əliyev tərəfindən yazılmış dərs vəsaitində informatika elminin əsasları, kompüterlər, onların inkişaf tarixi, quruluşu, iş prinsipi, şəbəkələri, proqram təminatı (MS DOS, Norton Commander, Windows, MS Paint, MS Word, MS Excel) nəzərdən keçirilir. Burada proqramlaşdırmanın əsasları, say sistemləri, alqoritm anlayışı, alqoritmlərin tipləri, ifadə formaları və qurulma qaydaları araşdırılır. Eləcə də BASIC (QBASIC), TURBO PASCAL və C++ proqramlaşdırma dilləri ətraflı surətdə nəzərdən keçirilir və proqramlaşdırma üzrə 1000-dən artıq misal və məsələ gətirilir.

Ə $\frac{4306010000}{026}$ 39 – 08

I BÖLMƏ

İNFORMATİKA VƏ HESABLAMA TEXNİKASI

I FƏSİL

İNFORMATİKA, ELEKTRON HESABLAMA MAŞINLARI (EHM)

1.1. İnformatika elmi haqqında

İnformatika – EHM-lər vasitəsilə informasiyanın tədqiqinin (emalının) ümumi qanunlarını, yəni informasiyanın verilməsi, yığılması və tətbiqi proseslərini araşdıran bir elmdir.

İnformatika – «Informatique» termini keçən əsrin 60-70-ci illərində fransızlar tərəfindən təklif olunmuşdur. Lakin onlardan əvvəl amerikalılar, hesablama texnikasının əsasında informasiyanın tədqiqini əhatə edən elmləri işarə etmək üçün «Computer Science» (hesablama elmləri) termini tətbiq etmişlər. Hal-hazırda «Informatique» və «Computer Science» terminlərindən ekvivalent mənada istifadə olunur.

EHM (elektron hesablama maşını) – informasiyanın avtomatik tədqiqini təmin edən elektron qurğudur.

Bizi əhatə edən aləm haqqında bütün elm və məlumatlara – informasiya deyilir. İnsan həmişə informasiyanı tədqiq edir:

1) yeni məlumatlar öyrənir (qəzetlərdən, kitablardan, radio və televiziyadan);

2) məlum informasiyadan istifadə edir (işində, həyatda);

3) yeni informasiyanı yaradır (elmdə, incəsənətdə, ədəbiyyatda və s.).

Müasir aləmdə informasiya çox böyük məna kəsb edir. İnsan fəaliyyətinin bütün sahələrində informasiya tədqiq etmək lazım gəlir. Xüsusi ilə isə elm, idarəetmə və siyasətdə informasiyanın tədqiqi əsas fəaliyyət sahəsidir. Bütün dünya əhalisinin yarısından çoxu informasiya tədqiqi ilə məşğuldur. İnsan təkcə bu qədər nəhəng həcmdə informasiyanı tədqiq etmək iqtidarında deyildir və onun köməyinə EHM-lər gəlir. Buna görə də müasir aləmdə hər bir insan öz işində EHM-dən istifadə etməyi bacar-

malıdır.

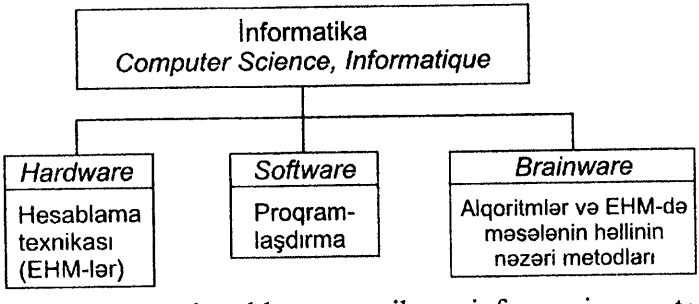
Qeyd edək ki, öz məsələlərinin həllində hesablama texnikasından istifadə edən şəxslərə kompüter istifadəçiləri deyilir. Onları iki kateqoriyaya bölmək olar:

1) proqramçı istifadəçilər
2) proqramçı olmayan istifadəçilər.

Birinci kateqoriyaya proqramçı-analitiklər (məsələlərin qoyuluşu, analizi və həllinin ümumi prinsiplərini araşdıran şəxslər), sistem proqramçıları (sistem və alət proqramlarını hazırlayan şəxslər) və tətbiqi proqramçılar (konkret elm sahəsindəki məsələlərin həlli üçün tətbiqi proqramlar hazırlayan şəxslər) aiddir.

İkinci kateqoriyaya isə kompüter operatorları (kompüterlərin proqram təminatının, birinci növbədə sistem və alət proqramların tətbiqi ilə məşğul olan şəxslər) və sadəcə istifadəçilər (konkret məsələləri tətbiqi proqramlarla həll edən və ya hər hansı bir informasiya alan şəxslər) aiddir.

İnformatikanı üç hissəyə bölmək olar:



1) *Hardware* - hesablama texnikası, informasiyanın tədqiqi üçün lazım olan qurğular;

2) *Software* - proqramlaşdırma, EHM-də istifadə olunan bütün proqramlar və onların qurulması;

3) *Brainware* - məsələlərin düzgün həlli üçün lazım olan bilik və vərdişlər.

1.2. EHM-lər və onların inkişaf tarixi

EHM-lərin yaranması, birinci növbədə fizika, riyaziyyat və texniki elmlərin tələbatı ilə diktə olunmuşdur. İnkişaf etməkdə olan elm və texnikanın məsələlərinin həlli üçün astronomik həcm-

də hesablamalar aparmaq lazım gəlirdi.

EHM-lərin adına baxmaqla, belə fikir yarana bilər ki, bu maşınlar yalnız hesablama aparmağa yarayır. Lakin bu belə deyildir, EHM-lər ixtiyari informasiyanı tədqiq edə bilir. EHM vasitəsilə, misal üçün təyyarələrin uçuş cədvəlini tərtib etmək və təyyarə biletlərini satmaq, abituriyentlərin imtahan cavablarını yoxlayıb, ali məktəblərə daxil olanların siyahısını tərtib etmək, dünyanın ixtiyari kitabxanası ilə əlaqə saxlayıb, lazım olan kitab və ya məqalənin surətini almaq olar. Xüsusi proqramlarla xarici dilləri öyrənmək, xəstəliyin düzgün diaqnozunu qoymaq, musiqi yazmaq, cizgi filmləri çəkmək, reklam hazırlamaq olar. Eləcədə EHM-lə şahmat və ya digər oyunlar oynamaq olar.

EHM-ləri «computer» - kompüter termini ilə işarə edirlər. Bu söz ingiliscə hesablayıcı, yəni hesablama üçün qurğu deməkdir. Ümumiyyətlə, kompüter - informasiyanı avtomatik tədqiq edə bilən ixtiyari qurğuya deyilir. Hal-hazırda bütün kompüterlər elektron tiplidir. Lakin ən birinci kompüter mexaniki tipli idi. İlk mexaniki kompüterin proyekti 1833-cü ildə İngiltərədə Bebbic tərəfindən təklif olunmuşdu.

İlk elektron tipli kompüter isə 1946-cı ildə ABŞ-da Pensilvan universitetində hazırlanmışdı. Bu EHM ENIAC (Electronic Numerical Integrator And Calculator) adlanırdı. Bu maşın həddindən artıq böyük idi və onu hətta bir yerdən başqa yerə aparmaq belə mümkün deyildi. Onun çəkisi 30 ton idi. Bu maşında 18000 elektron lampadan istifadə olunmuşdu və o, saniyədə 5000 əməliyyat apara bilirdi.

ENIAC və elektron lampalardan istifadə edilən digər EHM-lər, I nəsil EHM-ləri təşkil edir və bu 1940-1955-ci illəri əhatə edir.

1955-ci ildə növbəti II nəsil EHM-lər meydana gəldi. Bu EHM-də elektron lampalar əvəzinə yarımkeçiricilərdən – tranzistorlardan istifadə olunurdu. Bu EHM-lər ölçücə kiçik idi, onların işləməsi üçün daha az elektrik enerjisi tələb olunurdu. Bundan əlavə işləmə sürəti də artmışdı və saniyədə bir neçə on min əməliyyata bərabər idi. Həmin vaxtdan etibarən də proqramlaşdırma dillərindən istifadə edilməyə başlandı.

Bir müddətdən sonra elektron sənayesi inteqral sxemlər hazırlamağa başladı. İnteqral sxemlər – özündə bir neçə yüz və

hətta bir neçə min tranzistoru birləşdirən, kiçik yarımkeçirici kristallardır. İnteqral sxemlər əsasında qurulan EHM-lər III nəsil kompüterlərdir. Bu EHM-lər böyük yaddaşa və sürətə - saniyədə bir neçə milyon əməliyyat yerinə yetirmək qabiliyyətinə malikdirlər.

Müasir EHM-lər IV nəsil EHM-ləridir. Onlar 70-ci illərin əvvəllərində meydana gəlmişlər. Bu kompüterlərin əsas elementini mikroprosessor və digər böyük inteqral sxemlər (BİS) təşkil edir. Bu inteqral sxemlər özündə artıq bir neçə yüz min tranzistoru birləşdirir.

Mikroprosessorun icad edilməsi isə informatikada inqilaba səbəb oldu. Nəticədə iş üçün ən əlverişli EHM-lər fərdi kompüterlər (PC – Personal Computer) meydana gəldi. Bu EHM-lər kiçik ölçüyə malikdir, lakin onların böyük yaddaşı var və informasiyanı onlar çox böyük sürətlə tədqiq edirlər. Fərdi kompüterlərin bir-birilə və ya digər böyük EHM-lərlə, məsələn adi telefon xətti ilə birləşdirmək olar. Onda EHM şəbəkəsi əmələ gəlir və digər kompüterlərin də yaddaşında olan çox böyük həcmli informasiyadan istifadə etmək mümkün olur.

1.3. Say sistemləri

EHM - elektron rəqəm qurğusudur. Elektron qurğudur, ona görə ki, burada ixtiyari informasiya elektrik siqnallarının köməyi ilə təsvir olunur. Rəqəm qurğusudur, ona görə ki, EHM-də ixtiyari informasiya rəqəmlərin köməyi ilə ifadə olunur.

Rəqəmləri yazmaq üçün, hər hansı bir say sistemindən istifadə etmək lazımdır. Say sistemləri ədədlərin yazılması və onlar üzərində cəbri əməliyyatların aparılması qaydalarını təyin edir.

Biz əsasən onluq say sistemindən istifadə edirik. Bu say sistemində ixtiyari ədəd on rəqəm, 0, 1, 2, ... , 9 rəqəmlərinin köməyi ilə yazılır.

Ümumiyyətlə isə say sistemləri iki tipə bölünür: mövqeli və mövqesiz say sistemləri. Mövqesiz say sistemlərində ixtiyari rəqəmin qiyməti, onun ədəddə tutduğu mövqedən asılı olmur. Bu say sistemlərinə misal olaraq Roma say sistemini göstərmək olar. Bu sistemdə məsələn, XXX ədədində X rəqəmlərinin durduğu yer onun qiymətinə heç bir tə'sir etmir.

Mövqeli say sistemində isə ixtiyari rəqəmin qiyməti, onun ədəddə tutduğu mövqedən asılıdır. Bizim ən çox istifadə etdiyimiz onluq say sistemi də mövqeli say sistemlərinə aiddir. Məsələn, bu say sistemində götürdüyümüz 1998 ədədində 1 rəqəmi 1×10^3, yəni min, 9 rəqəmi 9×10^2, yəni doqquz yüz, 9 rəqəmi 9×10^1, yəni doxsan və nəhayət 8 rəqəmi 8×10^0, yəni səkkizi göstərir.

Mövqeli say sistemlərində, ədədləri ifadə etmək üçün, bu say sisteminin əsası adlanan müəyyən sayda rəqəmlərdən istifadə edilir. Məsələn, onluq say sistemində bu 0, 1, 2, ... , 9 rəqəmləridir. Məhz buna görə də bu sistem onluq say sistemi adlanır. Mövqeli say sistemlərinə misal olaraq, eləcə də ikilik (0, 1), səkkizlik (0, 1, 2, 3, 4, 5, 6, 7), onaltılıq (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F) və s. göstərmək olar.

Hər bir say sistemində ədədlər rəqəmlər ardıcıllığı kimi yazılır. Rəqəmin ədəddəki yeri onun mərtəbəsini, rəqəmlərin sayı isə ədədin neçə mərtəbələri olduğunu göstərir. Məsələn, onluq say sistemində verilmiş 1998,437 ədədini mərtəbələrə aşağıdakı kimi ayırmaq olar:

$$\overset{3\ 2\ 1\ 0\ -1\ -2\ -3}{1998,437} = 1\times10^3 + 9\times10^2 + 9\times10^1 + 8\times10^0 + 4\times10^{-1} + 3\times10^{-2} + 7\times10^{-3}$$

Bu qaydadan istifadə etməklə, digər say sistemlərində verilmiş ədədləri, dərəcələrinə ayırıb, onların etalon kimi qəbul etdiyimiz onluq say sistemindəki ekvivalent qiymətlərini tapa bilərik. *Məsələn:*

$$\overset{5\ 4\ 3\ 2\ 1\ 0}{110110}_{(2)} = 1\times2^5 + 1\times2^4 + 0\times2^3 + 1\times2^2 + 1\times2^1 + 0\times2^0 = 32 + 16 + 4 + 2 = 54_{(10)}$$

$$\overset{2\ 1\ 0}{175}_{(8)} = 1\times8^2 + 7\times8^1 + 5\times8^0 = 64 + 56 + 5 = 125_{(10)}$$

$$\overset{2\ 1\ 0}{A1F}_{(16)} = 10\times16^2 + 1\times16^1 + 15\times16^0 = 2560 + 16 + 15 = 2591_{(10)}$$

Onluq say sistemində verilmiş tam ədədlərin digər say sistemlərindəki ekvivalent ədədlərini tapmaq üçün isə verilmiş ədədi bu say sisteminin əsasına bölərək, qalıq həddlərini yazmaq

lazımdır. *Məsələn:*

Ədəd	Bölən	Qalıq
54	2	0
27	2	1
13	2	1
6	2	0
3	2	1
1	–	1

$54_{(10)} = 110110_{(2)}$

Ədəd	Bölən	Qalıq
348	8	4
43	8	3
5	–	5

$348_{(10)}=534_{(8)}$

Ədəd	Bölən	Qalıq
875	16	11
54	16	5
3	–	3

$875_{(10)}=35B_{(16)}$

Nəticə alınan qalıq hədlərini əks istiqamətdə, yəni aşağıdan yuxarıya yazdıqda alınır.

Ədədlərin müxtəlif say sistemlərindəki ifadəsini aşağıdakı cədvəllə verək:

Onluq	İkilik	Səkkizlik	Onaltılıq
00	00000	00	00
01	00001	01	01
02	00010	02	02
03	00011	03	03
04	00100	04	04
05	00101	05	05
06	00110	06	06
07	00111	07	07
08	01000	10	08
09	01001	11	09
10	01010	12	A
11	01011	13	B
12	01100	14	C
13	01101	15	D
14	01110	16	E
15	01111	17	F
16	10000	20	10
17	10001	21	11
18	10010	22	12
19	10011	23	13
20	10100	24	14
21	10101	25	15
22	10110	26	16
23	10111	27	17
24	11000	30	18

Onluq say sistemində verilmiş düzgün kəsrləri digər say sistemlərinə keçirmək üçün isə verilmiş kəsri ardıcıl olaraq keçirilən say sisteminin əsasına vururlar. Vurma nəticəsində alınan ədədlərin tam hissələri həmin say sistemində verilmiş kəsri ifadə edəcəkdir. *Məsələn:*

0,	725×2
1,	450×2
0,	900×2
1,	800
	və s.

0,	873×8
6,	984×8
7,	872×8
6,	976
	və s.

0,	27×16
4,	32×16
5,	12×16
1,	92×16
14,	72

$0{,}725_{(10)}=0{,}101_{(2)}$ $0{,}873_{(10)}=0{,}676_{(8)}$ $0{,}27_{(10)}=0{,}451E_{(16)}$

Nəticəni almaq üçün hər dəfə vurma nəticəsində alınan ədədlərin tam hissələrini yuxarıdan aşağıya doğru yazmaq lazımdır. Qeyd edək ki, hər addımda say sisteminin əsasını, alınan yeni ədədin yalnız kəsr hissəsinə vururuq. Bir də qeyd edək ki, onluq say sistemində verilən düzgün kəsri, digər say sistemlərinə həmişə dəqiq çevirmək olmur və onların təqribi ifadəsindən istifadə olunur.

Səkkizlik say sistemində verilmiş ədədi ikilik say sisteminə keçirmək üçün hər bir səkkizlik rəqəmin yerinə, bu rəqəmə uyğun gələn üç ikilik rəqəmi (triada) qoymaq lazımdır. *Məsələn:*

6	7	5	3	2	1	0	7
110	111	101	011	010	001	000	11

yəni $67532{,}107_{(8)}=110111101011010{,}001000111_{(2)}$

Əksinə, ikilik say sistemindən səkkizlik say sisteminə keçdikdə, hər bir ikilik triadaların yerinə onlara uyğun səkkizlik rəqəmlər qoyulur. Bu zaman əgər rəqəmin əvvəli və sonunda tam üçlük (triada) alınmazsa, onlar soldan və sağdan sıfırlarla tamamlanır. *Məsələn:*

010	111	011	101	110	100
2	7	3	5	6	4

yəni $10111011101{,}1101_{(2)}=2735{,}64_{(8)}$

Onaltılıq say sistemində verilmiş ədədi ikilik say sisteminə keçirmək üçün hər bir onaltılıq rəqəmin yerinə, bu rəqəmə uyğun gələn dörd ikilik rəqəmi (tetrada) qoymaq lazımdır. *Məsələn:*

3	5	B	4	5	1
0011	0101	1011	0100	0101	0001

yəni $35B451_{(16)}=1101011011,010001010001_{(2)}$

İkilik say sistemində verilmiş ədədi onaltılıq say sisteminə keçirdikdə isə hər bir ikilik tetradaların yerinə onlara uyğun onaltılıq rəqəmlər qoyulur və uclarda tam tetradalar alınmadıqda, bura soldan və sağdan sıfırlar əlavə olunur.

İkilik say sistemində cəbri əməliyyatlar aşağıdakı qaydalar üzrə aparılır:

		Toplama	*Vurma*
0	0	0	0
0	1	1	0
1	0	1	0
1	1	10	1

		Çıxma
0	0	0
1	0	1
1	1	0
10	1	1

Misal:

$$\begin{array}{r} 6 \to 110 \\ + \\ \hline 9 \to 1001 \\ \hline 15 \to 1111 \end{array} \qquad \begin{array}{r} 20 \to 10100 \\ - \\ \hline 15 \to 1111 \\ \hline 5 \to 0101 \end{array} \qquad \begin{array}{r} 6 \to 110 \\ \times \\ \hline 3 \to 011 \\ \hline 18 \to 110 \\ + \\ \hline 110 \\ \hline 10010 \end{array}$$

Bölmə əməli isə, bölənin, bölünəndən ardıcıl olaraq çıxılması ilə əvəz olunur. *Məsələn:*

$$\begin{array}{r|l} 101010 & 101 \\ \cline{2-2} 101 & 1000,011 \\ \cline{1-1} 00001000 & \\ 101 & \\ \cline{1-1} 0110 & \\ 101 & \\ \cline{1-1} 001 \text{ və s.} & \end{array}$$

Yoxlama:

$1000,011 \times 101 + 0,001 = 101001,111 + 0,001 = 101010$

1.4. EHM-də informasiyanın verilməsi. EHM-in iş prinsipi

İxtiyari informasiyanı kiçik elementar hissələrə bölmək olar. Məsələn, kitabdakı ixtiyari mətn hərf və digər simvollardan ibarətdir. Burada hərf – mətni informasiyanın elementar hissəsidir.

İnformasiyanın ixtiyari elementar hissəsi ədəd şəklində ifadə edilirsə, onda belə informasiyaya kodlaşdırılmış informasiya deyilir. Belə ədədlər isə kodlar adlanır. Əgər mətndəki hər bir hərfi kodlaşdırsaq, məsələn, bu hərfin əlifbadakı sıra nömrəsi ilə, onda bütün mətni kodlaşdırmaq olar.

EHM-lər isə informasiyanı yalnız kodlaşdırılmış şəkildə tədqiq edir. İnformasiyanın verilməsi üçün isə EHM-də yalnız iki simvoldan 0 və 1 rəqəmlərindən istifadə edilir. İnformasiyanın bu cür ifadəsi EHM-lərin texniki xüsusiyyətləri ilə izah olunur. Belə ki, kompüterlərdə ikilik say sistemindən ona görə istifadə olunur ki, bu sistemin rəqəmlərini elektrik siqnalları (cərəyanı) ilə çox asanlıqla ifadə etmək olur: 0 – siqnal yoxdur və 1 – siqnal var.

Minimal informasiya vahidi 1 bit (bit) adlanır. Bir bit informasiya - bu ikilik sistemin rəqəmlərindən biri ya 0, ya da 1 olur. Bu çox kiçik informasiya vahididir, buna görə də kompüterlərdə informasiyanın elementar hissələrinin tədqiqi üçün daha böyük vahiddən - bayt (byte) vahidindən istifadə olunur. Bir bayt səkkiz bitə bərabərdir. Bir baytda 256 simvoldan ($256 = 2^8$) birini kodlaşdırmaq olar.

EHM-in yaddaşının tutumu baytlarla ölçülür, lakin çox vaxt digər ölçü vahidlərindən istifadə olunur. Bunlar: kilobayt (1 *Kbayt*=1024 *bayt*); meqabayt (1 *Mbayt*=1024 *Kbayt*), qiqabayt (1 *Qbayt*=1024 *Mbayt*). Əgər bir səhifə mətndə təqribən 2500 işarə varsa, onda 1 Mbayt - təqribən 400 səhifə, 1 *Qbayt* isə 400 min səhifə mətn deməkdir.

Bir bayt - informasiya vahidi olmaqla yanaşı, həm də EHM yaddaşının elementar oyuğudur. EHM yaddaşı bu cür oyuqlar ardıcıllığından ibarətdir. Hər bir oyuğun (*bayt*) öz ünvanı - oyuğun nömrəsi var. EHM-in prosessoru informasiyanı tədqiq edərkən, o ünvana görə yaddaşda lazımi oyuğu tapır, oradakı informasiyanı oxuyub, lazımi əməliyyatları aparıb, alınan nəticəni digər bir oyuğa yazır.

Kompüterin işini ümumi şəkildə aşağıdakı kimi təsvir etmək olar. Əvvəlcə hər hansı bir xarici qurğudan istifadə etməklə, məsələn klaviatura vasitəsilə kompüterin yaddaşına proqram daxil edilir. Kompüterin idarəedici qurğusu, proqramın birinci əmri yerləşən yaddaşın oyuğundan bu əmri oxuyub, onun yerinə yetirilməsini təmin edir. Bu əmr isə cəbri və ya məntiqi əməliyyatların yerinə yetirilməsi, bu əməliyyatların yerinə yetirilməsi üçün verilənlərin yaddaşdan oxunması, bu əməliyyatların nəticələrinin yaddaşa yazılması, verilənlərin xarici qurğudan yaddaşa verilməsi və verilənlərin yaddaşdan xarici qurğulara verilməsi kimi işlərin görülməsini təmin edə bilər. Adətən, bir əmr yerinə yetirildikdən sonra idarəedici qurğu yaddaşın bu oyuqdan sonra gələn oyuqdakı əmri yerinə yetirməyə başlayır. Bu qayda yalnız idarəetmənin keçid əmrləri ilə dəyişdirilə bilər. Bu əmrlər idarəedici qurğuya, proqramın yerinə yetirilməsini, yaddaşın hər hansı başqa bir oyuğunda yerləşən əmrdən başlayaraq davam etdirilməsi haqqında göstəriş verə bilərlər.

Beləliklə, idarəedici qurğu proqramın göstərişlərini insanın köməyi olmadan avtomatik olaraq yerinə yetirir. O, kompüterin operativ yaddaşı və xarici qurğuları ilə əlaqə saxlayıb, onlardan informasiya alıb, informasiya verə bilir. Yerinə yetirilmiş proqramın bütün nəticələri idarəedici qurğu tərəfindən xarici qurğulara verilir və bundan sonra kompüter xarici qurğulardan yeni əmrlərin gözləmə rejiminə keçir.

1.5. EHM-in arxitekturası

Konstruksiya etibarilə fərdi EHM-lər aşağıdakı formalarda ola bilər: 1) stolüstü (Desktop); 2) Bloknot (Notebook) 3) Superbloknot (Sub-Notebook); 4) Cibə qoyulan ölçülü (Palmtop); 5) Elektron yazı kitabçası.

Fərdi EHM-lərin aşağıdakı imkanları var:

1) Məhsuldarlığı 1 saniyədə 1 milyondan çox əməliyyat (takt tezliyi 1000 MeqoHers)

2) Operativ yadda saxlama qurğusunun həcmi 128 *Mbayt*

3) Daimi yadda saxlama qurğusunun həcmi 47 *Qbayt*.

Fərdi EHM-lər aşağıdakı əsas (standart) qurğulardan ibarətdir:

1) Sistem bloku
2) Monitor
3) Klaviatura.

Bundan əlavə EHM-ə periferik adlanan aşağıdakı qrup qurğuları qoşmaq olar:

1) daxiletmə qurğuları: skaner, rəqəmsal fotokamera, qrafik planşet
2) xaricetmə qurğuları: printer, plotter
3) xarici yaddasaxlama qurğuları: maqnit və lazer disklərlə iş üçün diskovodlar, strimmer
4) idarəetmə qurğuları: siçan, trekbol, djoystik
5) həm daxiletmə, həm də xaricetmə funksiyalarını yerinə yetirən qurğular: modem, şəbəkə platası, səs platası.

Sistem blokunun tərkibinə sistem platası, elastik maqnit disklər üçün diskovodlar, bərk maqnit disk (vinçestr), daxiletmə və xaricetmə üçün giriş-çıxış oyuqları (portlar), elektrik təminatı bloku, səs dinamiki aiddir. Sistem platasında isə mikroprosessor, soprosessor (olmaya da bilər), operativ yaddaş modulları, tez yadda saxlama mikrosxemləri (KEŞ-yaddaş), giriş-çıxış sistemlərinin mikrosxemləri (BİOS), diskovodun, monitorun və digər qurğuların işini idarə edən adapterlər və kontrollerlər, sistem platasının müxtəlif elementlərini əlaqələndirən rabitə kanallarının kompleksi olur.

Mikroprosessor kompüterin əsas elementidir, bütün hesabi və məntiqi əməliyyatları yerinə yetirən, verilmiş proqram üzrə qoyulan məsələnin bütün həll prosesini idarə edən mikrosxemdir. Mikroprosessorun əsas xarakteristikaları: onun emal edə biləcəyi ədədlərdəki rəqəmlərin sayını müəyyən edən qiymət (bu qiymət 8, 16, 32 (köhnə model EHM-lər üçün) və ya 64 ola bilər) və takt tezliyidir – bu prosessorun bir saniyədə yerinə yetirə biləcəyi taktların sayıdır (takt – prosessorun elementar əməliyyatın yerinə yetirilməsinə sərf etdiyi zamandır). Müasir EHM-də bu tezlik 1000 MHersdir. Yadda saxlama qurğuları proqramların və verilənlərin qorunub saxlanmasını təmin edir və aşağıdakı tiplərə bölünür: operativ yadda saxlama qurğusu və ya operativ yaddaş, KEŞ-yaddaş, daimi yadda saxlama qurğusu və ya daimi yaddaş, xarici yadda saxlama qurğuları.

Operativ yaddaşda EHM-də aparılan cari əməliyyatlar

yerinə yetirilir. Bu yaddaşdakı informasiya yalnız EHM-in işlədiyi müddət ərzində saxlanılır. Müasir EHM-də operativ yaddaşın həcmi 2 Qbayta çata bilər, lakin praktik iş üçün 128 Mbayt yaddaş həcmi də kifayətdir.

KEŞ-yaddaş EHM-min yaddaşında aparılan əməliyyatların sürətləndirilməsini təmin edir. KEŞ-yaddaş 2 *Mbayta* qədər həcmə malik ola bilər.

Daimi yaddaş bura yazılmış verilənlərin və proqramların oxunması üçün nəzərdə tutulub. IBM tipli kompüterlərdə daimi yadda saxlama qurğusu ayrıca mikrosxem kimi verilir və burada əməliyyat hissəsinin bir bölməsi – daxiletmə və xaricetmənin baza sistemi (BIOS) saxlanılır.

Xarici yadda saxlama qurğusu böyük həcmli informasiyanın əsas yaddaşdan xaricdə uzun müddətə saxlanılmasını təmin edir. Yaddaş «xarici» olsa da onun çox hissəsi sistem blokunun korpusunda yerləşir. Xarici yaddaşa aşağıdakılar aiddir:

1) elastik maqnit disklərdə toplanan informasiya
2) bərk maqnit diskdə (vinçestrdə) toplanan informasiya
3) dəyişdirilə bilən bərk maqnit disklərdə toplanan informasiya
4) lazer kompakt disklərlə iş üçün diskovodlar
5) strimerlər

(Diskovod - elastik, lazer maqnit disklərdəki informasiya ilə işləməyə imkan verən qurğulardır).

Elastik maqnit disklərdə toplanan informasiya daşıyıcıları dəyişdirilə bilən maqnit disklər – disketlərdir. Disketlərdən EHM-lər arasında informasiya mübadiləsi üçün, informasiyanın EHM-dən kənarda saxlamaq üçün istifadə edilir.

Hal-hazırda adi disketlərdən və floppi disketlərdən istifadə olunur. Adi disket 3,5 *dyum* (89 *mm*) ölçüsünə və 1440 *Kbayt* informasiya daşımaq imkanına malikdir, disket bərk plastik korpusdan ibarətdir. Floptik disketlər də adi disketlərə oxşardır, lakin 21 *Mbayta* qədər informasiya daşıya bilir. Son zamanlar HIFT (Sony firması) tipli disket və diskovodlar tətbiq olunur. Bu disketlər də 3,5 *dyum* ölçülüdür, lakin 200 *Mbayt* həcmdə informasiya daşıya bilir. HIFT diskovodu 1,44 *Mbaytlı* disketlər üçün də uyğunlaşdırılıb.

Dəyişdirilməyən bərk maqnit disk (vinçestr) EHM-lə iş

zamanı daim istifadə olunan informasiyanın – əməliyyat sisteminin, proqram örtüyü proqramlarının və s. uzun müddətə saxlanmasını təmin edir. Vinçestr sistem blokunun korpusunda yerləşdirilir. Müasir IBM tipli fərdi EHM-də vinçestrin yaddaş həcmi ən azı 20 *Qbayt* ən çoxu 120 *Qbayt* və daha artıq olur. Dəyişdirilə bilən bərk maqnit disklər disketlər kimi informasiyanın saxlanması və daşınmasını təmin edir. Bu cür qurğuların bir neçə tipi mövcuddur, ən çox yayılmış qurğular dəyişdirilə bilən diskinin həcmi uyğun olaraq 200 *Mbayt* və 2 *Qbayt* olan Zip və JAZ adlanan diskovodlardır.

Lazer kompakt disklərlə iş üçün diskovodlar müxtəlif tip kompakt disklərdən informasiyanın oxunmasını təmin edir. Halhazırda istifadə olunan disklərin əsas tipləri aşağıdakılardır:

1) CD-ROM (Compact Disk – Read Only Memory) – kompakt disklərin yalnız oxunmasını təmin edir. Kompakt disklərə informasiya istehsalçı tərəfindən yalnız bir dəfə yazıla bilir, disk isə çoxlu sayda oxuna bilər. CD-ROM-un ölçüsü 120 *mm*, yaddaş həcmi ən azı 680 *Mbayt* olur.

2) CD-R disklər. Bu cür disklərə informasiya istifadəçi tərəfindən xüsusi yazı qurğusu (CD-WRITER) ilə bir dəfə yazıla bilər. Diski ixtiyari CD-ROM ilə çoxlu sayda oxumaq olur.

3) CD-RW disklər. Bu cür disklərə informasiya yazan diskovodla istifadəçi tərəfindən çoxlu sayda yazılıb, silinə bilər.

4) DVD disklər. Bu disklərin digər lazer disklərdən fərqi informasiyanın daha sıx yerləşdirilə bilməsindədir. Ən çox istifadə edilən DVD diskin ölçüləri 120 *mm*, informasiya tutumu 4,38 Qbaytdır. Bu disklərin tutumu 14 *Qbayta* qədər ola bilər. İki tip disklər mövcuddur: DVD-ROM – yalnız informasiyanı oxumaq üçün, DVD-RAM isə informasiyanı həm oxumaq, həm də yazmaq üçün istifadə olunur. DVD disklərin oxunması üçün xüsusi qurğudan istifadə olunur, bu qurğu adətən CD-ROM-un da oxunmasını təmin edir.

Strimerlər informasiyanın maqnit lentə köçürülməsini təmin edən qurğudur. Bu qurğu informasiyanı qabaqcadan sıxmaqla maqnit lentə köçürür. Bu qurğudan bərk diskdə olan vacib informasiyanın maqnit lentə köçürüb saxlamaq üçün istifadə olunur. Strimerin kasetlərində bir neçə qiqabayt həcmində informasiya saxlamaq olur.

Monitor EHM-ə daxil edilən və ondan alınan həm mətni, həm də qrafik informasiyanın zahiri ifadəsini təmin edən elektron qurğudur. Monitorlar maye kristallar əsasında (LCD displey) və ya elektron-şüa əsaslı olur. Monitor ekrandan və onun idarəetmə sistemindən ibarətdir. Bu sistemin ən vacib hissəsi videoadapter – sistem blokunda videoplatada yerləşdirilir. Bu qurğu monitorun imkanlarını müəyyən edir. Monitorun ekranı piksel adlanan elementar hissələrdən ibarətdir. Monitorun iki iş rejimi var: qrafik iş rejimi və mətn iş rejimi.

Qrafik iş rejimi ekrana qrafiklərin, şəkillərin və s. çıxarılmasını təmin edir. Bu rejimdə ekranın hər bir nöqtəsini idarə etmək, onlara ixtiyari rəng vermək, onlardan müxtəlif təsvirlər qurmaq olur.

Mətn iş rejimi mətnlərin ekrana verilməsi üçün nəzərdə tutulub.

Monitorların əsas xarakteristikaları aşağıdakılardır:

1) Monitorların tipləri: CGA, EGA, VGA, SVGA və s.

2) Videoplataların tipləri: CGA, EGA, VGA, SVGA və s.

3) Rəngin dərinliyi – ekrana eyni zamanda çıxarıla bilən rənglərin sayı ilə müəyyən edilir və videoyaddaşın ölçüsü ilə təyin edilir. SVGA adapteri 128 Mbayt ölçülü videoyaddaşa malikdir ki, bu da High Color iş rejimini (25536 rəng verilə bilir) və True Color iş rejimini (16,8 milyon rəng verilə bilir) təmin edir. Qeyd edək ki, ixtiyari rəngli monitordan monoxrom monitor kimi də istifadə etmək olur.

4) Ekranın ölçüləri – 14, 15, 17, 19, 21, 29 dyum (diaqonal üzrə) ola bilir.

5) Ekrana çıxarılan nöqtələrin sayı məsələn, 640×200 ola bilər. Eyni bir monitorda müxtəlif iş rejimlərində ekrana müxtəlif sayda nöqtələr çıxarıla bilir. *Məsələn*, VGA tipli monitorda ekrana 640×480 və 800×600 sayda nöqtə, SVGA tipli monitorda ekrana 1024×768 və 1280×1024 sayda nöqtə çıxarıla bilir.

6) Nöqtənin (pikselin) ölçüləri. Bu kəmiyyət kiçik olduqca ekranda təsvir bir o qədər keyfiyyətli alınır. Normal ölçü 0,25 *mm*-dir.

7) Kadrların dəyişmə tezliyi. Bu tezlik böyük olduqca istifadəçinin gözləri bir o qədər az yorulur. Normal tezlik 70 Hersdir.

Klaviatura informasiyanın EHM-ə daxil edilməsi və onun

işinin idarə edilməsi üçün nəzərdə tutulub. Windows sistemində iş üçün nəzərdə tutulmuş ən geniş yayılmış klaviatura 104 düymə-lidir (bura o cümlədən Windows sisteminin pəncərələrinin təsviri olan Win kimi işarə edəcəyimiz 2 düymə də aiddir). Hərf, rəqəm və digər işarələr klaviaturanın əsas hissəsini təşkil edən hərf-rəqəm düymələri ilə daxil edilir. Bundan əlavə klaviaturada aşağı-dakı xüsusi düymələr də var:

Caps Lock – bu düymə əlifbanın böyük hərflərinin daxil edilməsi rejimini verir. Bu düymənin təkrar sıxılması ilə böyük hərflərin daxil edillməsi rejimi ləğv olunur.

Enter düyməsi sətrin daxil edilməsinin sona çatdırılması üçün, yaxud hər hansı proqramın bu və ya digər sualına təsdiq edici cavab vermək üçün istifadə olunur.

Back Space – (*Enter* düyməsindən yuxarıda yerləşən və üzərində soldan sağa istiqamətləndirilmiş ox işarəsi olan düymə) kursordan (kursor – ekranda cari anda klaviaturadan daxil ediləcək simvolun yerləşəcəyi yeri bildirən göstəricidir) solda yerləşən simvolu ləğv edir.

Del (Delete) – kursordan sağda duran simvolu ləğv edir.

Ins (Insert) – simvolların daxil edilməsi rejimlərinin dəyişdirilməsi üçün nəzərdə tutulub.

" ↑ ", " ↓ ", " → ", " ← " düymələri kursoru ekranda oxların göstərdiyi istiqamətlərdə hərəkət etdirmək üçün istifadə olunur.

Home (End) – kursoru mətndəki sətrin əvvəlinə (sonuna) keçirir.

Pg Up (Pg Dn) – kursoru mətn üzrə ekranda yerləşə biləcək sətrlər həcmində yuxarı (aşağı) istiqamətlərdə hərəkət etdirir.

Num Lock düyməsi klaviaturanın sağ kənarında yerləşən düymələr blokundan rəqəmlərin daxil edilməsi üçün istifadə rejimini daxil edib və ləğv etmək üçün nəzərdə tutulub. Bu düy-mələrdən eləcə də kursorun işini idarə edən yuxarıda qeyd etdiyi-miz düymələr kimi də istifadə etmək olar.

Esc (Escape) düyməsindən adətən hər hansı bir əməliyyatı ləğv etmək, cari iş rejimindən, proqramdan çıxmaq üçün istifadə olunur.

F1–F12 – funksional düymələri, cari proqramın təyin etdiyi müxtəlif xüsusi əməliyyatların yerinə yetirilməsi üçün nəzər-

də tutulub.

Ctrl, Alt və Shift düymələri klaviaturanın digər düymələri ilə kombinasiyada işlədilir və bu düymələrin təyinatını dəyişdirmək üçün nəzərdə tutulub.

Space bar (probel) – boş yer daxil etmək üçün istifadə olunan və klaviaturanın aşağı hissəsində yerləşən, üzərində heç bir yazı olmmayan düymədir.

Pause – proqramın yerinə yetirilməsini müvəqqəti dayandıran düymədir.

Bəzi düymə kombinasiyalarını da qeyd edək:

Ctrl+Break kombinasiyası (MS DOS-da) yerinə yetirilən proqram və ya əmrin sona çatdırılmasını təmin edir.

Ctrl+Alt+Del kombinasiyası EHM-in əməliyyat sisteminin yenidən yüklənməsinə gətirir.

Shift+Print Screen ekrandakı informasiyanın çap qurğusu ilə çap olunmasını təmin edən kombinasiyadır.

Ctrl+Num Lock kombinasiyası proqramın yerinə yetirilməsini dayandırır. Proqramın işini davam etdirmək üçün ixtiyari düyməni basmaq kifayət edir.

Klaviatura ilə yanaşı EHM-in işini idarə edən qurğulara siçanı, trekbolu, kontakt panelini, coystiki də aid etmək olar.

Siçan – portativ manipulyator olub, hamar səth üzrə hərəkət etdirildikdə, kursorun da ekranda həmin istiqamətdə hərəkətini təmin edir. Bu qurğu 2 və ya 3 düyməsi olan bir qutu formasında olub, adətən EHM-lə xüsusi kabel ilə əlaqələnir. *Trekbol* – əksinə çevrilmiş siçan qurğusuna bənzəyir. Onu hərəkət etdirmirlər, bunun əvəzinə ondakı kürəciyi hərəkət etdirirlər. Kontakt paneli – EHM-lə kabellə əlaqələndirilmiş, adətən 76 x70 mm ölçülü qutudur. Qutunun qapağı ekrandan ibarətdir. Kursoru ekran boyunca hərəkət etdirmək üçün istifadəçi barmağını həmin ekran üzrə hərəkət etdirir. Barmağın bu ekrana vurulması, siçanın düyməsinin sıxılması effektini verir. Coystik – oyun proqramlarında istifadə olunur və ekranda kursorun idarə olunması üçün tətbiq olunur.

İndi isə xaricetmə qurğularına baxaq.

Printer – informasiyanın kağız üzərinə köçürülməsini təmin edən qurğudur. Bütün müasir printerlər, kağız üzərinə həm mətn, həm şəkil, həm də qrafiklər çıxara bilir. Hal-hazırda IBM PC tipli

fərdi EHM-lə uyğunlaşdırılmış kompütelərdə istifadə edilə bilən bir neçə min sayda müxtəlif model printer mövcuddur. Bunlar əsasən matris, şırnaq və lazer tipli printerlərdir. Matrisli printerlərdə çap edən başlıqda metal iynələr ardıcıllığı yerləşir, başlıq çap olunan sətr boyunca hərəkət edir və iynələr lazım olan anda rəngləyici lent üzərindən kağıza zərbələr endirir. Bununla da kağız üzərində simvol və təsvirlər formalaşdırılır. Bu tipli printerlərdə çap sürəti böyük olmur (dəqiqədə 0,5 – 2 səhifə), çapın keyfiyyəti də böyük deyil (360 *dpi (dpi* – 1 dyum kağız səthinə düşən nötələrin sayıdır)), lakin bu printerlər ucuz başa gəlir və istifadədə sadədir. Şırnaqlı printerlərdə çap edən başlıqda iynələr mikroskopik borucuqlarla, rəngləyici lent isə rəng konteynerləri ilə əvəz olunub. Rəngli printerlərdə əsas rənglərin sayına uyğun olaraq 4 belə konteyner («kartridj») ola bilər. Buradakı borucuqlardan rəng mikro damlalarla kağız üzərinə püskürdülür və beləliklə simvol və təsvirlər formalaşdırılır. Bu cür printerlərin təsvir keyfiyyəti 600 *dpi* qədər olur. Lazer tipli printerlərdə rəngləyici tozdan (toner) istifadə olunur. Çap edən başlıq rolunu xüsusi lazer sistemi oynayır. Bu sistem təsvirin proyeksiyasını fırlanan işığa həssas baraban üzərinə verir, buradan isə təsvir toner vasitəsilə kağız üzərinə köçürülür. Burada təsvir keyfiyyəti 1200 *dpi* qədər olur.

Plotter – mürəkkəb sxemlərin, qrafiklərin, keyfiyyətli rəngli təsvirlərin kağız üzərinə çıxarılması üçün nəzərdə tutlub. Ondan adətən proyekt və konstruktor bürolarında, reklamla məşğul olan təşkilatlarda istifadə olunur.

İndi isə daxiletmə qurğularını qeyd edək.

Skaner – kağız üzərindəki ixtiyari informasiyanın oxunub, EHM-ə daxil edilməsini təmin edir. Ağ-qara və rəngli simvollarla iş üçün nəzərdə tutulan skanerlər mövcuddur. Skaner təsviri EHM-ə koordinatları və rəngləri qeyd olunmuş nöqtələr çoxluğu kimi daxil edir, sonra bu verilənlər əsasında ekrana təsvir çıxardılır. Skanerlə mətnləri oxuyub, onlarla mətn kimi işləmək tələb olunduqda, təsviri mətn kimi qəbul etməyə imkan verən xüsusi proqramlardan istifadə olunur. İngilis, rus və digər dillərdə olan mətnlər üçün uyğun proqramlar mövcuddur. Onlar mətnləri, məsələn, MS Word mətn redaktoruna keçirməyə imkan verir. Qeyd edək ki, hətta əlyazma mətnlərini oxumağa qadir olan

proqramlar mövcuddur.

Rəqəmsal fotokamera ilə çəkiliş adi qaydalarla aparılır, kamera EHM-ə qoşulduqda çəkilən kadr ekrana çıxarılır və printerlə çap oluna bilər.

Qrafik planşet xüsusi örtüyü olan qurğudur, onun üzərində xüsusi qrafik qələmlə yazmaq olur. Bütün yazılanlar EHM-ə təsvir şəklində daxil edilir.

Həm daxiletmə, həm də xaricetmə funksiyaları olan qurğulara baxaq.

Modem EHM-i qlobal kompüter şəbəkələrinə qoşulmasını təmin edən qurğudur.

Şəbəkə adapteri – EHM-in lokal kompüter şəbəkəsinin tərkibində işləməsini təmin edir.

Səs platası səs informasiyasının rəqəmsal formaya çevirməyə imkan yaradır.

1.6. Kompüter şəbəkələri

Kompüter şəbəkələrini aşağıdakı iki qrupa bölmək olar:

1. Lokal şəbəkələr (LAN – Local Area Network). Bir binada və ya bir-birinə yaxın yerləşən binalar kompleksində (məsələn, universitetin ayrı-ayrı fakültələrinin yerləşdiyi binalar) olan kompüterlər, hər hansı məsələlər birgə həll edirsə, informasiya mübadiləsi edir, eyni bir verilənlər bazasından istifadə edirsə, proqramlardan, periferik qurğulardan birgə istifadə edirsə, onların lokal şəbəkələrdə birləşdirilməsi əlverişli olur.

2. Qlobal şəbəkələr (WAN – World Area Network). Bu cür şəbəkələrdən isə bir-birindən çox böyük məsafələrdə yerləşən kompüterlərin əlaqələndirilməsi üçün istifadə edilir.

Əvvəlcə lokal şəbəkələrə baxaq. Bu cür şəbəkələrdə EHM-in müxtəlif əlaqələndirmə sxemlərindən (şəbəkə topologiyası) istifadə olunur. Belə ki, «ulduz», «halqa» və «ağac» topologiyaları mövcuddur.

«Ulduz» birləşmə sxemində şəbəkənin bütün EHM-i şəbəkənin mərkəzində yerləşən və server adlanan baş EHM-ə qoşulurlar. Server bütün şəbəkənin mərkəzləşdirilmiş idarəçiliyini təmin edir, məlumatların verilmə marşrutlarını müəyyən edir, periferik qurğuları ehtiyac olduqca şəbəkəyə qoşur, bütün şəbəkə üçün verilənlər bazası rolunu oynayır.

«Halqa» topologiyasında şəbəkədəki bütün EHM-lər ardıcıl olaraq bir halqada birləşir və serverin funksiyaları bütün EHM-lər arasında paylanır. Bu birləşmə sxemləri arasında ən əlverişli və deməli, ən geniş yayılmışı «ağac» birləşmə strukturudur. Bu cür şəbəkəyə qoşulmuş ixtiyari EHM server ola bilər. Şəbəkədə əlaqə kanalının əsas xarakteristikası, onun keçiricilik qabiliyyətidir, yəni informasiyanın maksimal ötürmə sürətidir. Bu sürət saniyədə bit, kilobit və ya meqabitlə ölçülür. Şəbəkədə aşağıdakı əlaqə kanallarından istifadə edilə bilir:

1) Bir-birinə hörülmüş keçiricilər cütü – bu əlaqə kanalı bir-birinə hörülmüş iki cüt keçiricidən ibarətdir. Bu əlaqə kanalında keçiricilik imkanları 1 *Mbit/san*-dir, lakin xüsusi keçricilər tətbiq etməklə keçiricilik sürətini 10 *Mbit/san* və hətta 100 *Mbit/san* çatdırmaq olur. 150 *metr* məsafəyə qədər keçiricilik sürəti 10 *Mbit/san*, 80-90 *metr* məsafədə isə sürət daha böyük olur.

2) Koaksal kabel (BNC) – orta keçiricilik qabiliyyətinə malikdir, lakin bu əlaqə kanalı hörülmüş keçiricilər cütünə nisbətən 1,5-2 dəfə daha böyük məsafədə normal keçiricilik sürətini təmin edir. Əlavə gücləndiricilər olmadan bu məsafə 180-200 *metr*, bəzi hallarda daha çox ola bilər.

3) Optik-lifli kabel – ən böyük keçiricilik qabiliyyətinə malikdir. Bu əlaqə kanallarında sürət 60 *Qbit/san* və daha artıqdır.

4) Naqilsiz əlaqə – burada informasiya EHM-lər arasında infraqırmızı şüalarla ötürülür. Bu zaman ötürücü EHM-lə qəbuledici EHM bir-birinin görünmə sərhədlərində olmalıdır. Bu cür əlaqədən, məsələn, Notebook tipli portativ EHM-i şəbəkədə birləşdirərkən istifadə etmək əlverişli olur. Bu tip şəbəkələrə misal olaraq AirLAN, Altair Plus şəbəkələrini göstərmək olar. Qeyd edək ki, əlaqə kanalı rolunu adi elektrik şəbəkəsi oynayan lokal şəbəkələr də var. *Məsələn*, Carriernet şəbəkəsi buna misal ola bilər. Həmçinin əlaqə kanalları kimi radio şəbəkəsindən də istifadə edilə bilir.

Lokal şəbəkənin qurulması üçün tələb olunan əsas avadanlığı qeyd edək:

1) «Xab» (hub - mərkəz) – siqnalın gücləndirilməsi funksiyaları olmayan paylayıcı qurğu

2) «Ripiter» (repeater - təkrarlayıcı) – siqnalı gücləndirə bilən paylayıcı qurğu

3) «Bridj» (bridge - körpü) – şəbəkənin iki müxtəlif seqmenti arasında əlaqə yaratmaq üçün istifadə olunur

4) «Sviç» (switch – dəyişdirici açar) – şəbəkənin bir neçə müxtəlif seqmenti arasında əlaqə yaradır.

Lokal şəbəkənin idarə edilməsini təmin edən proqramlar kompleksinə şəbəkənin proqram təminatı deyilir. Bu proqramların əsas hissəsi serverdə, digər hissəsi isə şəbəkəyə qoşulan EHM-lərdə yerləşdirilir. Bu cür şəbəkə sistemlərindən ən çox istifadə ediləni NOVELL firmasının Netware sistemidir. Eləcə də MicroSoft firmasının MS Network şəbəkə proqram təminatından da geniş istifadə olunur. İxtiyari lokal şəbəkənin işi aşağıdakı prinsipə əsaslanır. Şəbəkəyə qoşulmuş ixtiyari EHM öz məxsusi nömrəsinə (identifikatora) malikdir, konkret EHM-dən informasiya şəbəkəyə ayrı-ayrı informasiya paketləri şəklində daxil olur, hər bir paketdə onun şəbəkədəki hansı EHM üçün nəzərdə tutulduğu haqqında məlumat olur və paket şəbəkə üzrə azad hərəkət edə bilir. Paketdəki ünvan hissəsi şəbəkədəki hər bir EHM-in nömrəsi ilə müqayisə olunur və bu ünvanlar üst-üstə düşdükdə informasiya nəzərdə tutulmuş EHM tərəfindən qəbul edilir. Paket lazımi ünvanı şəbəkədə tapmadıqda məhv edilir. Qeyd edək ki, informasiya şəbəkə üzrə ona qoşulmuş bütün EHM-ə eyni zamanda göndərilə bilər. Şəbəkənin bu funksiyasına «broadkastinq» deyilir.

İndi isə qlobal şəbəkələrə baxaq. Əvvəlcə bu cür şəbəkələrin yaranma tarixi ilə tanış olaq. 1968-ci ildə ABŞ hökuməti özünün Sovet İttifaqından xüsusi rabitə və kosmik proqramlar sahəsindəki geriliyindən narahat olaraq müdafiə nazirliyinin kompüterləri arasında rabitə sistemlərinin yaradılması üçün maliyyə vəsaitləri ayırır və bu proyektin yerinə yetirilməsini «ARPA» – Advanced Research Projects Agency (ABŞ-ın Müdafiə Nazirliyinin perspektiv araşdırmalar İdarəsi)-nə tapşırır. Beş ildən sonra informasiya paketlərinin kommutasiyasına əsaslanan şəbəkə yaradıldı və bu şəbəkə yaradıcının adı ilə ARPAnet adlandırıldı. Şəbəkədə informasiya mübadiləsi IP (Internet Protocol - tərcümədə «şəbəkə arası protokol») əsasında qurulmuşdur. Onun köməyilə şəbəkədə rabitə əlaqəsi qurulur və təmin edilir. 1982-ci ildə o

zaman ARPAnet protokolları adlanan IP-protokolları TCP/IP ailəsini yaratdı.

Qlobal şəbəkələr çox böyük əraziləri əhatə etməklə yanaşı, lokal şəbəkəyə nəzərən aşağıdakı məxsusiyyətlərə malikdir:

1. Qlobal şəbəkələrdə tarixi olaraq çox vaxt rabitə kanalları kimi telefon xətlərindən istifadə olunur, bu isə əlaqəyə kifayət qədər böyük xətalar gətirir və informasiyanın ötürülməsi sürəti nöqtəyi nəzərdən nisbətən daha zəifdirlər. Lakin son zamanlar çox vaxt rabitə kanalı kimi informasiya mübadiləsinin yüksək sürətini təmin edən peyk radiokanallarından və optik-lifli rabitə kanallarından istifadə olunur.

2. EHM-lər rabitə kanallarına modemlər adlanan xüsusi qurğuların köməyilə bağlanır.

3. Qlobal şəbəkələrin konfiqurasiyası müxtəlif olur və lokal şəbəkədən fərqli olaraq requlyar deyil.

TCP/IP tipli şəbəkələrdə müəyyən məlumat göndərmək üçün EHM göndərilən məlumatı «zərf» deyəcəyimiz bir informasiya daşıyıcısına yerləşdirib, zərfin üzərində isə məlumatın çatdırılacağı son şəbəkə ünvanını göstərməlidir. Rabitə seansı zamanı zərfin daxilindəki məlumat, adətən ölçüsü 128 *bayt* olan və «paketər» adlanan bərabər hissələrə bölünür. Hər bir paketin üzərində son şəbəkə ünvanı göstərilir, sonra paketlər kompüter şəbəkəsinə buraxılır. Hər bir paket ünvana öz yolu ilə gəlir və son nöqtədə bu paketlərin hamısının ünvana çatdığı, zədələnmədiyi yoxlanılır. Zədələnən və ya ünvana çatmayan paketlər olduqda məlumatın göndərildiyi şəbəkə nöqtəsindən onları əvəz edəcək paketlər təkrarən göndərilir. Əgər paket öz yolunu azıb, sonuncu məntəqəni tapmazsa, müəyyən olunmuş zamandan – TTL (time to live – «həyat zamanı») sonra o, məhv edilir. Belə olmazsa şəbəkənin düyünlərindəki – IP-rounterlərdəki yaddaş buferlərinin dolub daşması baş verər. TTL-paketin həyat zamanı, paketin yolu boyunca rast gəlinən düyünlərin sayından düz mütənasib asılıdır. Lakin düyünlər arası «sıçramaların» və ya hopların (hop - ingiliscədən tərcümədə sıçrayışdır) sayı 15-30 dəfə ilə məhdudlaşır.

Son illər qlobal şəbəkə istifadəçilərinin sürətli artımı üzündən verilənlərin ötürülməsi üçün istifadə olunan telefon xətləri şəbəkəsi ötürülən informasiya həcmini təmin edə bilmir.

Köməyə peyk rabitəsi gəlir. Bu rabitə elə təşkil olunur ki, istifadəçilər kiçik peyk antenlərin olduğu stansiyaların yaxınlığında yerləşmiş olsun. Bu cür kiçik stansiyalar öz aralarında və şəbəkə ilə peyk vasitəsilə rabitə yaradırlar və VISAT adlanırlar. Lakin telefon xətt şəbəkələri hələlik əsas rabitə xətləri olaraq qalır. Telefon xətləri EHM-in işlədiyi rəqəm siqnalların ötürülməsi üçün nəzərdə tutulmayıblar. Bu xətlərdən qlobal şəbəkələrdə istifadə edilməsi xüsusi qurğuların – modemlərin köməyilə mümkün olmuşdur. Beləliklə, modem – kompüterlərdən siqnalları alaraq onları telefon xətləri şəbəkəsinə uyğun olan formaya və əksinə çevirən qurğudur. Konstruksiya etibarilə istifadəçi modemləri 2 formalı: xarici və daxili formalı olur. Daxili modem fərdi EHM-in sistem blokunda ana plataya birləşdirilən plata formasındadır. Xarici modem isə bir tərəfdən EHM-ə, digər tərəfdən isə telefon şəbəkəsinə qoşulur.

Müasir qlobal şəbəkələrin istifadəçilərə verdikləri imkanlara baxaq:

1. Elektron poçt (e-mail). Elektron poçt qlobal şəbəkələrin təklif etdiyi ən geniş yayılmış və sadə servis növüdür. Şəbəkə ilə göndərilən məlumatda adi məktubda olduğu kimi göndərənin və qəbul edənin ünvanları olur. Məlumat mətnində verilənlər faylı da ola bilər. Elektron poçtla göndərilən məlumat müəyyən zaman müddətindən sonra onun göndərildiyi EHM-ə çatır və istifadəçi onu istədiyi vaxt oxuya bilir. Burada məlumatın çatdırılması haqqında bildiriş və digər əlavə imkanlar var.

2. Elanların elektron lövhəsi (Bulletin Beard Systems və ya qısa BBS). Bu modemi olan baş (düyün) maşında qoyulmuş tətbiqi proqramdır. İstifadəçilər adi telefon xətlərinin köməyilə bu maşina qoşulurlar. Bu EHM-in diskində hər bir istifadəçinin daxil ola biləcəyi yaddaş sahəsi ayrılıb. Hər bir istifadəçi bu sahəyə müraciət edib, oraya öz informasiyasını daxil edə yaxud oradan informasiyanı öz EHM-nə köçürə bilər. Beləliklə, hər bir istifadəçi BBS-in köməyilə öz informasiyasını şəbəkədə yayır və buradakı informasiyadan bəhrələnə bilir.

3. Telekonferensiya (və ya NEWS) Qlobal şəbəkə istifadəçilərinin maraqları üzrə qruplara bölünməsi üsuluna telekonferensiya üsulu deyilir. İstifadəçiyə maraqlı olan istiqamət üzrə telekonferensiyaya yazılmaq kifayət olur. Nəticədə bu kompüterə

uyğun istiqamət üzrə məlumatlar axını gəlir. İnsan maraqları o qədər genişdir ki, hər bir konferensdə bir çox səviyyə və alt səviyyələr olur, məsələn: «kitablar/dərsliklər/riyaziyyat» və s.

Qlobal şəbəkəyə ən bariz misal, praktik olaraq bütün dünya üzrə yayılmış İnternet şəbəkəsi ola bilər. Əslində bu şəbəkə deyil, elmi, tədris, dövlət, hərbi şəbəkələrin, verilənlərin ötürülməsinin ümumi protokolu ilə birləşdirilmiş şəbəkələr külliyyatıdır. İnternet şəbəkəsinin imkanları çox genişdir. Onlardan bəzilərini qeyd edək.

1. Elektron poçt bir və ya bir neçə şəxsə məlumat göndərməyə, eləcə də ixtiyari şəxsdən və kompüter proqramlarından informasiya almağa imkan verir.

2. Faylların ötürülməsi. Bu prosedur FTP fayllarının ötürülməsi protokolu ilə təyin edilir. O, uzaqdakı EHM-ə daxil olub, onların ümumi istifadə üçün açıq olan kataloqlarına müraciət etməyə imkan verir. Sonra lazımi faylları arxiv EHM-dən düyün EHM-in köməyi ilə istifadəçinin maşınına köçürmək olur.

3. Uzaqdakı mənbələrə daxil olmaq. Başqa cür Telnet adlanır, uzaqda yerləşən EHM-ə qoşulub onlarla dialoq rejimində işləməyə imkan verir. Telnet prosedurunun köməyi ilə, İnternetin tərkibinə daxil edilmiş ixtiyari açıq EHM-ə daxil olaraq bu uzaqda olan EHM-lə öz məxsusi maşını kimi davranmaq, ixtiyari informasiyanı aramaq olur.

Bu qeyd etdiyimiz əsas üç funksiya (xidmətlərin baza tipləri) İnternetdə istifadəçilər üçün çox saylı şəbəkə xidmətlərini təşkil etməyə imkan verir.

WWW xidməti. İnternet şəbəkəsinin təklif etdiyi xidmətlər arasında hal-hazırda ən çox istifadə ediləni – WWW (World Wide Web – ümumdünya hörümçək toru). WWW xidməti İnternet şəbəkəsində olan informasiyaya yiyələnmək üçün ən rahat vasitədir. WWW xidməti multimediali sənədlə, yəni mətn, səs, üçölçülü təsvir və s. imkanları olan sənədlə işləyir. Belə sənəd «səhifə» (Web-səhifə) adlanır və qeyd olunmuş struktura malikdir. WWW xidmətinin emal etdiyi bütün informasiya Web-səhifələr şəklində verilir. Hər bir səhifə şəbəkədə özünəməxsus ünvana (URL) malikdir. Hər hansı bir şəxs, firma və ya təşkilat öz informasiyasının WWW vasitəsilə hamıya çatdırmaq istəyirsə, o bu infor-

masiyanı Web-səhifə şəklində ifadə etməlidir. Bu səhifələr İnternetin bir hissəsi olan, Web-serverlər adlanan EHM-də saxlanılır. Web-serverin sahibi ixtiyari firma ola bilər. Əslində, burada WWW xidməti əlaqələndirilmiş Web-serverlər külliyatıdır. WWW xidməti «müştəri-serverlər» prinsipi əsasında işləyir. Müştərinin sorğusu ilə onun Web-serverlərdən biri ilə əlaqəsi yaradılır və o, müştəriyə uyğun informasiyalı Web-səhifə göndərir. Bundan sonra növbəti sorğuya qədər müştəri ilə əlaqə dayandırılır. İnformasiyanın dəyişdirilməsi və ya yeni informasiyanın daxil olması haqqında Web-server müştərini xəbərdar etmir. Web-server və ya Web-saytla əlaqə xüsusi proqram – Web-brouzerin (ingiliscədən browse – nəzərdən keçirmək) köməyilə həyata keçirilir. Bu cür iki əsas proqram – Netscape Navigator və Microsoft Explorer proqramlarıdır.

WWW xidməti yalnız Web-səhifələrin sadəcə oxunması ilə bitmir. Adətən, səhifədə izaha ehtiyacı olan terminlər, sözlər, şəkillər olur. Bu izahlar başqa Web-səhifələrdə verilə bilir. Səhifənin bu cür elementini (söz, termin, şəkil) izah edən Web-səhifəni ekrana çıxarmaq üçün siçanın kursorunu bu element üzərinə yerləşdirib düyməni sıxmaq kifayətdir. Səhifənin izahlarla müşayət olunan elementləri mətndə başqa rənglə və ya qeyd edilmə ilə seçilir və hiper-mətnli müraciətlər adlanır. Onları həm də onunla fərqləndirmək olar ki, kursor bu elementlər üzərinə yerləşdirilərkən «ovuc» şəklinə düşür. İkinci səhifədəki müraciətlə üçüncünü, üçüncüdən dördücünü və s. sona çatana qədər çağırmaq olar. Beləliklə, başlanğıc Web-səhifə ilə izahedici səhifələrin bütöv bir çoxluğu bağlı ola bilər. Onların da hər birinin öz ünvanı var, lakin lazımi informasiyanı almaq üçün birinci səhifənin ünvanını bilmək kifayətdir.

WWW xidmətində lazımi informasiyanın tapılması üçün arayış sistemləri mövcuddur. Periodik olaraq jurnallarda Web-səhifələrin kataloqları dərc olunur. Bütün bilik sahələri üzrə belə səhifələrin sayı 23000-dən çoxdur və onlar gündəlik olaraq artır. Lakin İnternetdəki axtarış sistemlərindən istifadə etmək daha sadə olur. Belə sistemlərə misal olaraq «Rambler», «Яndex», «Yahoo», «Likos» və s. başqalarını göstərmək olar.

İnternet şəbəkəsindən telefon əlaqələri yaratmaq üçün də istifadə etmək olar. Səs kartı və İnternetə girişi olan kompüterdə

«NetMeeting» tipli proqramı işə salıb, çox uzaq məsafədə olan digər şəxslə həmsöhbət olmaq olar. Bu zaman klaviaturanın köməyi ilə həmin şəxsin terminalına informasiya ötürməklə yanaşı mikrofon və audiokolonkanı işə salıb onunla söhbət etmək də mümkündür. Burada əlaqə kanalında sürət ən azı 14400 *bit/saniyə* olmalı və böyük gecikmələr olmamalıdır.

İnternet vasitəsilə həmçinin videokonfranslar da təşkil etmək mümkündür. Lakin bunun üçün telefon əlaqəsinə nisbətən 5 dəfə artıq sürət tələb olunur. Təsvirin ötürülməsindəki gecikmələrə, səsin ötürülməsinə nisbətən daha az tələblər qoyulur. Adətən yalnız delta-siqnal, yəni təsvirdəki dəyişmələr ötürülür. Bununla ötürülən informasiyanın artımı və gecikmələrə tələblər azaldılır.

Şəbəkədə ikilikdə deyil, bütöv bir dəstə ilə söhbət etmək olur. Bunun üçün çatı dəstəkləyən serverə girmək lazımdır. Çatda isə istifadəçinin həmin qayda ilə artıq bura girmiş həmsöhbətlər dəstəsi gözləyir.

Son zamanlar İnternetdə ICQ adlanın ünsiyyət vasitəsi geniş yayılmağa başlamışdır. Bu əlaqə vasitəsi əsasən kompüter qarşısında uzun müddət işləyən şəxslər üçün nəzərdə tutulub. Rahat səs və qrafik interfeysli bu əlaqə vasitəsi həm çat, həm də elektron poçtun üsünlüklərini özündə birləşdirir. ICQ-dən istifadə etmək üçün uyğun proqram təminatını yükləyib, «Mirabilis» serverində registrasiyadan keçmək kifayətdir.

II FƏSİL

EHM-LƏRİN PROQRAM TƏMİNATI

2.1. EHM-lər üçün proqramlar

EHM-də ixtiyari informasiya proqramlar vasitəsilə tədqiq olunur. Bunun üçün kompüterin qəbul edəcəyi dildə, informasiyanı emal etmək üçün lazım olan dəqiq və ətraflı göstərişlər ardıcıllığını (proqramı) tərtib etmək lazımdır. Öz-özlüyündə heç bir kompüter onun tətbiq edildiyi sahələr üzrə heç bir biliklərə malik deyildir. Bütün bu biliklər kompüterlərdə yerinə yetirilən proqramlarda cəmlənmişdir. Kompüter üçün müxtəlif proqramlar tətbiq etməklə, onu mühəndis və ya mühasibin, aqronom və ya riyaziyyatçının iş yerinə çevirmək olar. Buna görə də kompüterdən effektiv istifadə etmək üçün, onunla iş üçün lazım olan proqramların təyinat və xassələrini bilmək lazımdır.

Hal-hazırda EHM-in hər bir tipi üçün çoxlu sayda proqramlar işlənib hazırlanmışdır. EHM-in işlədilməsi üçün zəruri olan prqramlar külliyatına EHM-in poqram təminatı deyilir. EHM-də istifadə olunan əsas tip proqramlar aşağıdakılardır:

1) *Sistem proqramları.* Bu proqramlar EHM-in işlədilməsi və onlara texniki qulluq, EHM-də ixtiyari konkret məsələnin həlli zamanı hesablama prosesinin təşkili və idarə edilməsi və s. üçün nəzərdə tutulub. Onlara əməliyyat sistemləri, əməliyyat sistemlərini örtükləri, utilit-proqramlar, antivirus proqramları və s. aiddir.

2) *Proqramlaşdırma sistemləri.*

3) *Alət proqramlar.* Bu proqramlardan gündəlik fəaliyyətdə sənədlərin hazırlanması üçün bir alət kimi istifadə olunur. Onlara mətinlərlə iş üçün nəzərdə tutulan proqramlar (mətn redaktorları), qrafik redaktorlar, elektron cədvəllər (cədvəl prosessorları), verilənlər bazasını idarəetmə sistemləri, özündə bir neçə alət proqramı birləşdirən mühitlər və s.

4) *Mətnlərin avtomatik tərcümə proqramları*

5) *Tədris proqramları*

6) *Tətbiqi proqramlar*

7) *Oyun proqramları*
8) *Multimedia proqramları.*

Əvvəlcə sistem proqramına baxaq. Əməliyyat sistemləri və əməliyyat sistemlərinin proqram örtükləri sistem proqramlarının əsas növləridir. Bu vacib proqramlarsız müasir EHM-in işi mümkün deyildir. Birinci növbədə bu əməliyyat sistemlərinə aiddir.

Əməliyyat sistemi – məsələlərin həlli prosesində və istifadəçi ilə EHM arasında qarşılıqlı əlaqənin təşkili, EHM-in bütün vasitələrindən effektiv istifadə üçün nəzərdə tutulmuş proqramlar kompleksidir. Fərdi EHM-də əməliyyat sistemi xüsusilə vacib rol oynayır, çünki məhz o, fərdi EHM-lə işi sadə və əlverişli edir. IBM PC tipli EHM-lər üçün də bir çox tip əməliyyat sistemi mövcuddur. Onlardan hər biri öz tətbiq sahəsinə malikdir.

IBM PC tipli kompüterlər üçün yaradılan ilk əməliyyat sistemi Microsoft firmasının MS DOS sistemi idi. Bu sistem ən etibarlı və sadə sistemdir. Sonrakı illərdə firma tərəfindən bu tipli kompüterlər üçün MS DOS-un əsasında Windows 3.1 (3.11) sistemi yaradıldı. Bu əməliyyat sistemi üzərində iş aparılaraq onun daha mükəmməl variantları Windows 95, Windows 98, Windows 2000, Windows NT, Windows XP, Windows Vista meydana gəldi.

Beləliklə, əməliyyat sistemi istifadəçilər ilə EHM arasında əlaqə yaradıb, istifadəçiyə EHM-i idarə etmək imkanını verir. Lakin sistem əmrlərin köməyi ilə EHM-i idarə etmək çətinliklər yaradır, bu çətinlikləri aradan qaldırmaq üçün proqram örtüklərindən istifadə olunur. Praktiki olaraq hər bir əməliyyat sistemi üçün proqram örtükləri mövcuddur. Təkcə MS DOS üçün bu cür proqramlardan bir neçə tip hazırlanmışdır. Onlardan ən geniş yayılanları Norton Commander, Norton Navigator, XTree, Norton Desktop, PC ToolsDesktop, MS DOS Shell və başqalarıdır.

Sistem proqramlarına utilit proqramları adlanan proqramlar da aiddir. Bu proqramlar istifadəçi üçün xeyirli olan müxtəlif əməliyyatlar yerinə yetirir. Bunlar, məsələn disklərdəki verilənləri sıxlaşdıraraq onların tutduğu həcmi kiçildən arxivator proqramlar, EHM-dəki verilənləri qoruyan və bərpa edən proqramlar, disklərdəki informasiyanı optimal yerləşdirən proqramlar və s. Utilitləri adətən müəyyən proqram paketlərində birləşdirirlər. Belə paketlərdən ən çox istifadə olunanı və güclüsü Norton Utilities adlanır. Bu paketdən olan və MS DOS üçün nəzərdə tutu-

lmuş Ndd (Norton Disk Doctor) proqramı, eləcədə ona uyğun Windows 95 tərkibində olan Scandisk proqramı diskləri (disketləri) onların məntiqi sisteminin düzgünlüyü nöqteyi-nəzərdən testləşdirməyə, disklərin səthindəki zədəli sektorların təyin edilməsinə və bir çox əməliyyatların aparılmasına imkan verirlər. Antivirus proqramlar da utilit proqramlardandır. Virus proqram mətni təhrif edərək, onu tamamilə işə yaramaz hala gətirə bilir. Virus duşmüş proqram, başqa proqramları da bu virusa yoluxdura bilir. Yoluxmaya əsasən *com, exe* ad genişlənmələri olan proqram faylları məruz qalır. Mətni fayllar, proqramlaşdırma dillərində tutulmuş proqram mətnləri, sənəd mətinləri və s. fayllar virusa yoluxmur, onlarda yalnız mətn təhrif oluna bilir. Virusa yoluxmuş EHM-də bir çox lazımlı proqram paketləri sıradan çıxa bilir, lazımı fayllar silinə bilir və viruslarla mübarizə çox zaman tələb edir. Çox saylı viruslarla effektli mübarizə üçün antivirus proqramları yaradılır. Belə proqramların bəzi tiplərini qeyd edək:

1) Həkim – proqramlar, proqramları «müalicə» edir, onları əvvəlki formaya qaytarır və viruslardan təmizləyir;

2) Filtr – proqramlar, əməliyyat sistemlərinə daxil olmağa çalışan virusların qabağını alaraq, onlar haqqında istifadəçiyə məlumat verirlər.

Ən geniş yayılmış antivirus proqramlarına periodik olaraq yeniləşən və tamamlanan AIDSTEST, DRWEB (DOKTOR WEB) proqramlarını göstərmək olar. İnternet şəbəkəsi ilə ötürülən və yeniləşən güclü antivirus proqramlarına misal olaraq AVP (Anti Viral ToolKit Pro), Kasperski və s. göstərmək olar.

Texniki qulluq proqramlarının köməyi ilə kompüter sistemləri testləşmədən keçirilir, tapılmış çatışmamazlıqlar aradan qaldırılır, EHM-in bəzi qurğularının işi optimallaşdırılır.

Konkret tip EHM üçün konkret alqoritmik proqramların yaradılıb, işlədilməsi üçün nəzərdə tutulmuş proqram və digər vasitələr kompleksinə proqramlaşdırma sistemləri deyilir. Proqramlaşdırma sistemi adətən proqramlaşdırma dilinin müəyyən versiyasından, bu dildə verilən proqramların translyatorundan və s. ibarət olur. Hər bir proqramlaşdırma sistemi ilə müəyyən proqramlaşdırma dili bağlı olduğundan, dillər haqqında məlumatlar verək.

Proqramlaşdırma dili – kompüter proqramlarının yaradıl-

proqramlaşdırma sistemləri (verilənlər bazasının idarəetmə sistemi-VBİS) nəzərdə tutulmuşdur. Belə sistemlər adətən proqramlaşdırma dilindən, translyatordan və proqramlaşdırma mühitindən ibarətdir. VBİS bazada olan verilənlər əsasında sənədləri, cədvəlləri tez və dəqiq hazırlamağa imkan verir. Ən çox istifadə edilən sistemlər Foxpro, MS Access, Lotus1-2-3 və s.

Prenzentasiya proqramları geniş kütlə qarşısında çıxışlar üçün nəzərdə tutulmuş slaydların və digər nümayiş materiallarının yaradılmasında istifadə olunur (məsələn, MS PowerPoint). Riyazi proqramlar proqramlaşdırmadan istifadə etmədən riyazi məsələlərin çox geniş sinfini həll etməyə imkan verir (məsələn, Maple, MathCad).

Multimedia proqramlarına səsyazma proqramları, səs və video faylların redaktorları, musiqi sintezatorlarının proqramları və s. aiddir. Nitq proqramları mətnin EHM-ə nitq vasitəsilə daxil edilməsini, sənədlərin məzmununa qulaq asılmasını və s. təmin edir.

Özündə bir neçə alət proqramını birləşdirən mühitə misal olaraq Microsoft Works 3.0 (4.0) mühitini göstərmək olar. Bu mühit mətn prosessorunu, elektron cədvəlini, verilənlər bazasının daxil edilməsi və yaradılması proqramını, qrafik redaktoru özündə birləşdirir. Bir mühitdə müxtəlif növ proqramların birləşdirilməsi mühitdən kənara çıxmadan praktik olaraq ixtiyari sənədlərin hazırlanmasına, ehtiyac olduqda bir proqramdan digərinə keçməyə şərait yaradır. Bundan əlavə verilənləri bir proqramdan digərinə keçirmək olar. Bu cür mühitlərə misal olaraq MS Office, Lotus SmartSuite və s. göstərmək olar.

Mətnlərin avtomatik tərcümə etmə proqramları müəyyən bir dildə verilən mətnlərin başqa bir dilə tərcümə edilməsi üçün nəzərdə tutulmuşdur. Bu proqramlar ayrı-ayrı söz və söz birləşmələrinin tərcüməsini ani olaraq və səhvsiz yerinə yetirir, lakin bütöv bir abzas və bir neçə abzasdan ibarət mətnin düzgün tərcüməsini bu proqramlarla praktik olaraq yerinə yetirmək mümkün olmur. Lakin bununla yanaşı bu proqramlar xarici dili heç bilməyən şəxslər üçün, xarici dildə verilmiş mətn haqqında hər hansı bir anlayış almaq üçün, elektron poçtun qısa məlumatlarının tərcüməsi üçün əlverişli olur. Bu proqramlara misal olaraq Sokrat, Stylus,Promt 98 sistemlərinin göstərmək olar. Sistemlər elmin

müxtəlif sahələri üzrə böyük lüğətlərə, mətnlərin avtomatik tərcümə etmə proqramlarına, Web-səhifələrin sinxron tərcümə proqramlarına və s. malikdir.

Tətbiqi proqramlar elm və texnikanın bu və ya digər sahələrinin tətbiqi məsələlərinin həlli üçün nəzərdə tutulub. Bu proqramlara misal olaraq mühasibat proqramlarını, nəşretmə sistemlərini, maliyyə təhlili proqramlarını, avtomatik proyektləşdirmə sistemlərini, verilənlərin statistik təhlili proqramlarını və s. göstərmək olar.

Tədris proqramları müxtəlif fənlərin öyrədilməsində geniş istifadə olunur. EHM-dəki proqramlarla iş qaydalarını öyrədən tədris proqramları da mövcuddur. Tədris proqramları arasında uşaqların yaradıcılıq qabiliyyətlərini üzə çıxarmağa imkan verən inkişafetdirici proqramları xüsusi qeyd etmək lazımdır.

Son illərdə kompüter oyun proqramları sahəsində çox böyük nəticələr alınmışdır. Bu proqramlarda kompüterin qrafik imkanlarından, üçölçülü təsvirlərdən geniş istifadə olunur.

Multimedia, EHM-dən mətnin, stereosəsin, səs müşayətinin, yüksək keyfiyyətli qrafikanın, videokliplərin, animasiyanın, hətta virtual reallığın tətbiqi ilə istifadə qaydalarına deyilir. Başqa sözlə multimedia EHM-in rəqamsal və mətni informasiyanın səs siqnalları və video siqnallarla birləşdirilməsi vasitəsidir. Mutimedialı kompüterdə səs stereoplatası, videomaqnitafon, videokamera, rəqəmsal fotokamera, televizorla iş üçün video daxiletmə platası, CD-ROM-la iş üçün diskovod, səs stereokolonkaları, mikrofon və tələb olunan proqram təminatı olmalıdır. Hal-hazırda multimedia üçün zəruri olan bütün proqramları özündə birləşdirən proqram paketləri (məsələn, Multimedia Kit) hazırlanır.

2.2. Fayl və kataloq anlayışları

Maqnit disklərdəki ixtiyari informasiya fayllar şəklində saxlanılır. Hər bir proqram da fayl (File) təşkil edir. Proqramda istifadə olunan ədədi və digər verilənlər, fayllar şəklində saxlanıla bilər.

Faylları çox vaxt iki qrupa – mətni və ikilik fayllarına bölürlər. Mətn faylları insan tərəfindən oxunması üçün nəzərdə

tutulub. Məsələn, bu səhifədə yazılan mətn də fayl şəklində maqnit diskə yazıla bilər və bu fayl mətni fayl olacaqdır. Mətn fayllarında eləcədə proqramların mətnləri və s. yerləşdirilə bilər. Mətn faylı olmayan yerdə qalan bütün mümkün fayllar adətən ikilik faylları adlanır.

Hər bir fayl - faylın adı (file name) ilə işarələnir. Faylın adı bir-birindən nöqtə ilə ayrılan iki hissədən ibarət olur. Birinci hissə - faylın əsas adı, birdən səkkizə qədər simvola malik ola bilər. İkinci hissə isə - faylın adının genişlənməsi (file name extention) üçə qədər simvola malik ola bilər.

Məsələn, DOS sisteminin əsas proqramının faylının adı COMMAND. COM, (COMMAND – əsas ad, COM isə faylın adının genişləndirilməsidir).

Faylın adında latın hərflərindən və kompüterdəki digər simvollardan, rəqəmlərdən istifadə etmək olar. Lakin faylın adını bir-birindən probellə ayrılmış simvollarla vermək olmaz. *Məsələn, paper.doc, autoexec.bat, FILE10.BAS* və s. fayl adları düzgün, *FILE10.TEXT, 2< >1.TXT* və s isə düzgün deyildir.

Fayl adının genişləndirilməsi fayldakı informasiyanın xarakteri haqqında məlumat verir və buna görə də ondan istifadə əlverişli olur. Bundan əlavə, bir çox proqramlar (məsələn, Norton Commander) faylın adının genişləndirilməsinə əsaslanaraq, uyğun proqramı çağırıb, verilən faylı ora yükləməyə imkan verir, bu isə vaxta xeyli qənaət etmək deməkdir. Bəzi fayl adının genişləndirilmələrini qeyd edək:

.com, . exe - yerinə yetirilən fayllar;
.bat - komanda (Batsh) faylları;
.bas - BASIC dilində olan proqramlar;
.pas - PASCAL dilində olan proqramlar;
.cpp – C++ dilində olan proqramlar;
.bak - dəyişdirilməzdən əvvəl faylın yaradılan təkrarı;
.txt - mətn fayllar və s.

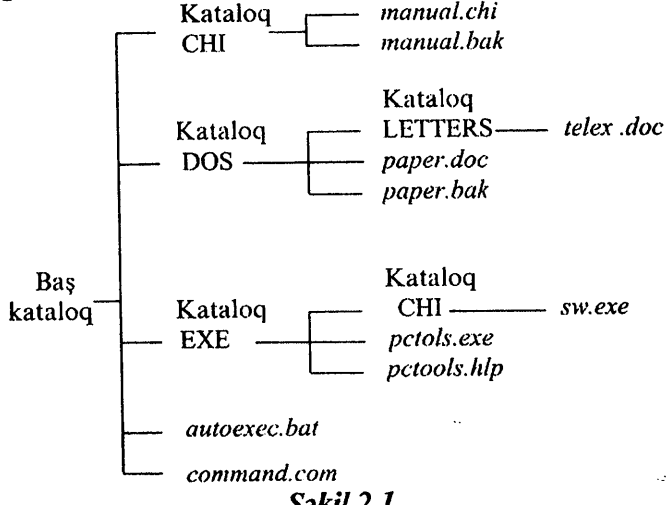
Qeyd edək ki, fayllar üzərində iş zamanı faylın *.bak* genişləndirilməsindən istifadə çox əlverişli olur. Faylın belə bir təkrarının olması, fayl üzərində işləyərkən, əgər müəyyən informasiya səhvən dəyişdirilərsə və ya pozularsa, onu yenidən bərpa etməyə imkan verir. Fayl üzərindəki işi bitirdikdən sonra isə, faylın təkrarını ləğv etmək olar.

Kompüterin işə salınması ilə verilən əməliyyat sistemlərindən başqa, ixtiyari proqram öz tərkibində, bu proqramı işə salan fayla malikdir. Belə fayl yerinə yetirilən fayl adlanır. Başqa sözlə yerinə yetirilən fayl, proqramın baş faylı olub, onun yerinə yetirilməsini təmin edir. Əgər proqram yeganə bir fayldan ibarətdirsə, onda elə bu fayl yerinə yetirilən fayldır. Bu cür fayllar adətən *.com* və *.exe* genişləndirilmələrinə malik olurlar.

Diskdəki bütün faylların siyahısı diskin kataloqu adlanır. Kataloq - diskdə, faylların adları, onların ölçüləri, axırıncı dəfə təzələnməsi vaxtı, xassələri və s. haqqında məlumat olan bir sahədir. Əgər kataloqda hər hansı bir faylın adı varsa, onda deyirlər ki, bu fayl həmin kataloqda yerləşir. Hər bir diskdə bir neçə kataloq ola bilər. Hər bir kataloqda isə çoxlu sayda fayl ola bilər, lakin hər bir fayl yalnız bir kataloqda qeyddən keçir. Hər bir kataloqun öz adı vardır və hər bir kataloq digər bir kataloqun daxilində qeyddən keçə bilər. Əgər X kataloqu Y kataloqunda qeyddən keçirsə, onda X kataloqu Y kataloqunun alt kataloqudur.

Kataloqların adlarına olan tələblər elə fayllarda olduğu kimidir. Lakin kataloqlarda adətən ad genişləndirilməsindən istifadə olunmur.

Hər bir diskdə bir baş kataloq olur. Bu kataloqda fayllar və alt kataloqlar və ya birinci səviyyə kataloqları qeyd olunur. Birinci səviyyə kataloqlarda isə öz növbəsində fayllar və ikinci səviyyə kataloqları qeyd olunur və s. Nəticədə diskdəki kataloqların ağaca bənzər, budaqlanan strukturu alınır. *Məsələn:*



Şəkil 2.1

Kompüterdə işləyən şəxsin hal-hazırki anda işlədiyi kataloqa, cari kataloq deyilir. Məsələn, Windows və ya Norton Commander-də işlərkən, ekranda cari kataloqun tərkibi haqqında məlumat verilir.

DOS sistemində cari kataloqun başlığını vermək üçün Dir komandasını vermək lazımdır, bu kataloqdan digərinə keçmək üçün isə Cd əmrindən istifadə olunur. Norton Commander, Windows sisteminin fayllar dispetçerində, Windows 95-in bələdçisində və s. isə cari kataloqun dəyişdirilməsi, bir kataloqdan digərinə keçərkən avtomatik yerinə yetirilir.

Ümumiyyətlə, fayllar sistemini kitabxana ilə müqayisə etmək olar:

maqnit diski - kitablar olan rəf;
fayl - kitab;
faylın adı - kitabın adı;
diskin kataloqu - kitabların siyahısı, kataloqu.

Cari kataloqdan olmayan faylla işləyərkən, bu faylın hansı kataloqdan olduğunu göstərmək lazımdır. Bunu isə fayla keçid yolunu göstərməklə etmək olar. Fayla keçid yolu - kataloqların adları və ya « \ » simvolu ilə ayrılan «..» simvolları ardığıllığından ibarətdir. Bu yol - cari kataloqdan və ya diskin baş kataloqundan, axtarılan fayl yerləşən kataloqa olan istiqaməti göstərir. Əgər yol « \ » simvolundan başlayırsa, istiqamət diskin baş kataloqundan götürülür. Məsələn, tutaq ki, cari kataloq DOC götürülüb (şəkil 2.1), onda

Baş kataloqdan başlayan yol	Cari kataloqdan başlayan yol	
\ CHI	..\ CHI	(1)
\ DOC \ LETTERS	LETTERS	(2)
\ EXE \ CHI	..\ EXE \ CHI	(3)

Burada (1) birinci səviyyə CHI kataloquna olan yolu, (2) DOC kataloqunun alt kataloqu olan LETTERS kataloquna olan yolu, və (3) EXE kataloqunun alt kataloqu olan CHI kataloquna olan yolu göstərir.

Kompüterdə adətən bir neçə, diskovod adlanan və bərk

disklərdə, disketlərdə, kompakt-disklərdə və s. olan informasiyanı yığan və özündə saxlayan disklər olur. Onların hər birində kataloq və fayllar yerləşdirilə bilir. Bizə lazım olan diski çağırmaq üçün, əvvəlcə uyğun diskovoda onun adı ilə müraciət etmək lazımdır. Adətən, diskovodlar A, B, C, D, E və s. kimi adlandırılırlar. Məsələn, kompüterdə iki A və B diskovodları, maqnit elastik disklər, yəni disketlər üçün, bir C diskovodu isə bərk maqnit diski (vinçester) üçün nəzərdə tutula bilər. A və B diskovodları xarici yaddaşda, yəni disketlərdə olan informasiyanı oxumaq üçün, C diskovodu isə adətən əməliyyat sistemlərini kompüterə yükləmək üçün istifadə olunur.

Hal-hazırda işlənən diskovoda - cari diskovod deyilir. Məsələn, Windows və ya Norton Commander sistemində işlərkən ekranda cari diskovoddakı, kataloqun tərkibi verilir.

Beləliklə, bunları nəzərə almaqla, faylın tam adını aşağıdakı şəkildə ifadə etmək olar:

(diskovod:) (yol \) faylın adı; (harada mötərizə daxilində verilməsi vacib olmayan elementlər köstərilir).

Faylın adı, faylın yerləşdiyi kataloqa olan yol, bundan « \ » simvolu ilə ayrılan faylın adı və bunların hamısının qarşısında diskovodun adından ibarətdir. Əgər diskovodun, adı verilmirsə, onda cari diskovodun, əgər kataloqun yolu göstərilmirsə, onda cari kataloqun olduğu başa düşülür.

Məsələn, tutaq ki, şəkil 2.1-də A diskovodunda olan fayl sistemi təsvir olunub və cari kataloq, burada - A:\ DOS şəklindədir. Onda *a: paper.doc* - A diskovodundakı, diskin cari kataloqunun *paper.doc* faylını göstərir, *a:\ paper.doc* isə A diskovodundakı, diskin baş kataloqunun *paper.doc* faylını göstərir və s.

Eyni bir kataloqdan olan fayllar qrupunu göstərmək üçün faylların adlarında «*» və «?» simvolarından istifadə etmək olar. Belə ki, «*» stmvolu faylın adında və ya onun adının genişdirilməsində ixtiyari sayda və ixtiyari simvolları göstərir. «?» simvolu isə faylın adı və adının genişləndirməsində ixtiyarı bir simvolun olduğunu və ya olmadığını göstərir.

Məsələn: **.bak*-cari kataloqdakı.bak genişdirilməsinə malik,bütün fayları , C*.d* - cari kataloqdakı adı C ilə, adının genişləndirilməsi d ilə başlayan bütün faylları göstərir.

2.3. MS DOS əməliyyat sistemi

MS DOS əməliyyat sistemi 1981-ci ildə Microsoft firması tərəfindən, IBM firmasının sifarişi ilə, o vaxtlar yaradılan IBM PC kompüterləri üçun yaradılmışdır. Hal-hazırda 1994-cü ildə hazırlanmış MS DOS 6.22 versiyasından istifadə olunur. MS DOS əməliyyat sistemi, kompüterə, kompüter elektrik şəbəkəsinə qoşulmaqla və ya EHM-nin korpusundakı Reset düyməsini basmaqla avtomatik yüklənir.

DOS sistemi işə hazır olduqda, ekrana məsələn: A> və ya C:\> şəklində dəvət verir. Bu dəvətdə adətən, cari diskovod və cari kataloq haqqında məlumat olur. Məsələn: A:\> yəni A diskovodu və baş kataloq, C:\EXE yəni C diskovodu və \EXE kataloqu. Əmrləri vermək üçün, bu əmri klaviaturadan əmr sətrinə yığıb *Enter* düyməsini basmaq lazımdır.

Mətn fayllarının yaradılması. Bunun üçün copy con *faylın adı* – əmrini vermək lazımdır. Bundan sonra faylın sətrlərini daxil etmək lazımdır. Hər bir sətrin sonunda *Enter* düyməsini, mətnin axırıncı sətrindən sonra isə əvvəlcə *F6*, sonra isə *Enter* düyməsini basmaq lazımdır. Nəticədə diskdə göstərdiyimiz adlı fayl yaranacaqdır.

Faylların ləğv edilməsi. Bunun üçün aşağıdakı əmr verilir:

del *faylın adı*

Burada faylın adında * və ? simvollarından da istifadə olunur. *Məsələn:* del **.bak* – cari kataloqdan bak fayl ad genişlənməsinə malik bütün fayllar çıxarılır.

Faylların adlarının dəyişdirilməsi. Bunun üçün aşağıdakı əmr verilir:

ren *faylın adı1 faylın adı2*

Burada *faylın adı1*, adı dəyişdiriləcək faylı, *faylın adı2* isə bu fayla veriləcək yeni adı göstərir. Məsələn: ren *fox.doc fox.txt* – cari kataloqdakı *fox.doc* faylının adı *fox.txt* adı ilə dəyişdirilir.

Faylların təkrarının alınması. Bunun üçün aşağıdakı əmr verilir:

copy *faylın adı1 faylın adı2*

və ya

copy *faylın adı1 (kataloqun adı2)*

Burada faylın adı 1 parametri təkrarı alınacaq faylı, faylın adı 2 isə bu təkrarı alınan faylın yeni adını göstərir. Faylın

göndərildiyi kataloqu, kataloqun adı 2 parametri ilə, ya da faylın adı 2 parametrində kataloqun da adını verməklə təyin etmək olar. Əgər faylın təkrarının göndərildiyi kataloq göstərilmirsə, onda həmin fayl təkrarı cari kataloqa göndərilir və nəhayət əgər faylın adı 1 parametrində kataloqun adı göstərilirsə, onda fayl bu kataloqdan götürülərək təkrarı alınır, əks halda bu fayl cari kataloqdan götürülür. Burada faylın adında * və ? simvollarından istifadə etmək olar.

Məsələn: copy *c1.doc c1.txt* -cari kataloqdakı *c1.doc* faylının təkrarı alınır və oradakı *c1.txt* adlı faylında yerləşdirilir; copy A: \ *.* - A diskinin baş kataloqundakı bütün faylların təkrarı alınaraq cari kataloqa köçürülür və s.

Faylın digər kataloqa göndərilməsi. Bunun üçün aşağıdakı əmrdən istifadə olunur:

move *faylın adı kataloqun adı*

Burada faylın adında * və ? simvollarından istifadə edilə bilər. Bu əmr yerinə yetirilərkən adı göstərilən fayl, adı göstərilən kataloqa göndərilir. Məsələn: move *.*doc* d: -cari kataloqdakı *.doc* ad genişlənməsinə malik fayllar, d diskindəki cari kataloqa göndərilir.

DOS-da kataloqlarla iş. Cari diskovodu dəyişdirmək üçün, cari olacaq diskovodun adını yığıb, iki nöqtə simvolunu qoymaq lazımdır. *Məsələn:* A: , B: və ya C: və s. Bu komandalar verildikdən sonra *Enter* düyməsi basılmalıdır.

Cari kataloqun adına dəyişdirmək üçün aşağıdakı komandadan istifadə etmək olar:

move *kataloqun adı kataloqun yeni adı*

Məsələn: move *vin vin1* - cari kataloqun vin kataloqun adı *vin1*-ə dəyişdirilir.

Yeni kataloq yaratmaq üçün aşağıdakı əmrdən istifadə olunur;

md *(diskovod): yol*

Məsələn: md S1 - cari kataloqda S1 adlı alt kataloq yaradılır; md a: \ S2 - A diskovodunun baş kataloqunda S2 adlı alt kataloq yaradılır və s.

Boş kataloqu ləğv etmək üçün aşağıdakı əmrdən istifadə olunur;

rd *(diskovod): yol*

Məsələn: rd *r1* - cari kataloqdakı *r1* alt kataloqu ləğv edilir; rd *a:* \ *r2* – A diskovodundakı baş kataloqda olan *r2* alt kataloqu ləğv edilir.

Cari kataloqu dəyişdirmək üçün aşağıdakı əmrdən istifadə olunur:

cd *(diskovodun adı): yol*

Əgər burada diskovodun adı verilirsə, onda cari kataloq bu diskovodda, əks halda isə cari diskovodda dəyişdirilir. *Məsələn:* cd \ – cari diskin baş kataloquna keçidi, cd\exe\doc - bu isə /exe/doc kataloquna keçidi göstərir və s.

Kataloqun tərkibini nəzərdən keçirmək üçün isə aşağıdakı əmrdən istifadə olunur: dir *diskovod: yol/faylın adı* parametrlər.

Burada faylın adında «*» və «?» simvollarından istifadə etmək olar. Əgər faylın adı verilmirsə, kataloqda olan bütün fayllar haqqında, əks halda isə adı göstərilmiş fayl və ya fayllar haqqında məlumat verilir. Əgər burada diskovod və yol göstərilmirsə, onda cari diskovoddakı, cari kataloq haqqında məlumat verilir.

dir əmri parametrlərsiz verildikdə, bu əmr hər bir fayl üçün onun adını, ad genişləndirilməsini, faylın baytlarla ölçüsünü, bu faylın yaradıldığı və ya axırıncı dəfə təzələndiyi tarixi göstərir. *dir* əmri eləcədə parametrlərlə də verilə bilər. Belə parametrlər çoxdur, bunlardan bəzilərini qeyd edək:

/p - bu parametr verildikdə, kataloqun tərkibi ekrana səhifə-səhifə verilir, yəni bu parametr verildikdə, DOS sistemi, ekran kataloqdakı məlumatla dolduqdan sonra gözləmə rejiminə keçir və klaviaturadakı ixtiyari bir düymə basıldıqda növbəti səhifəyə keçir.

/w - bu parametr verildikdə isə, yalnız faylların və alt kataloqların adları verilir (alt kataloqların adı kvadrat mötərizə daxilində verilir), buna verilənlərin geniş formatda verilməsi deyilir.

/ o - bu parametr verildikdə isə, əvvəlcə alt kataloqların adları əlifba sırası ilə, sonra isə faylların adları əlifba sırası ilə verilir.

Misallar: *dir* - cari kataloqun tərkibi verilir; *dir*.exe* - cari kataloqdakı *.exe* ad genişlənməsi olan fayllar haqqında məlumat verilir; dir a:\ - A diskovodundakı baş kataloqun tərkibi verilir;

dir /p - cari kataloqun tərkibi ekrana səhifə-səhifə verilir; *dir/w* - cari kataloqun tərkibi geniş formatda verilir və s.

Kompüterdə tarix və zaman haqqında informasiyanın verilməsi və dəyişdirilməsi. Tarix haqqında informasiyanın verilməsi və onun dəyişdirilməsi üçün aşağıdakı əmrdən istifadə olunur:

date

Bu əmr ilə ekrana il, gün və ay haqqında məlumat verilir. Əgər tarix dəyişdirilməyəcəksə, *Enter* düyməsini basmaq lazımdır. Yəni tarix vermək üçün ayın gününü (1 - 31), ilin ayını (1-12) və ili və ya ilin axırıncı iki rəqəmini vermək lazımdır. Burada MS DOS onların hansı ardıcılıqla veriləcəyini göstərir (DD - gün, MM - ay, YY - il), ədədlər arasında isə «–» işarə qoyulur. *Məsələn:*

enter new date (dd-mm-yy): 21-04-2008

Zaman haqqında məlumat almaq və onu dəyişdirmək üçün aşağıdakı əmr var:

time (saat, dəqiqə),

harada saat 0-dan 24-ə qədər ədədlər, dəqiqə isə 0-dan 59-a qədər ədədləri göstərir. Əgər əmr parametrsiz, yəni time kimi verilirsə, onda DOS cari zamanı verir və yeni zamanı qoymağı tələb edir. Əgər zamanı dəyişdirməyə ehtiyac yoxdursa, onda *Enter* düyməsini basmaq lazımdır.

Məsələn: time - cari zaman verilir, time 11:45 – 11:45 zamanı qoyulur.

DOS-da ekran və printerlə iş qaydaları. Mətn faylını ekrana çıxarmaq üçün

type *faylın adı*

əmrindən istifadə olunur. Məsələn: type *kl.doc* – cari kataloqdakı *kl.doc* faylı ekrana verilir.

Faylın ekrana çıxarılmasını dayandırmaq üçün *Ctrl* düyməsini basıb saxlamaqla *C* düyməsini basmaq lazımdır. Bu düymələri təkrarən basmaqla, faylın ekrana çıxarılmasını davam etdirmək olar. Faylın ekrana çıxarılmasını dayandırmaq üçün isə *Ctrl* və *C* düymələrini və ya *Ctrl* və *Break* düymələrini basmaq kifayətdir.

Monitorun ekranını informasiyadan təmizləmək üçün aşağıdakı əmrdən istifadə olunur:

cls

Bu əmr yerinə yetirilərkən monitorun ekranı təmizlənir və

ekranın birinci sətrində DOS sisteminə dəvət haqqında məlumat verilir.

Mətn faylını çap etmək üçün aşağıdakı əmrdən istifadə olunur:

copy / b *faylın adı* prn

Bu əmr verilməmişdən qabaq, çap qurğusu işə hazır vəziyyətdə olmalıdır.

2.4. Norton Commander (NC) proqram örtüyü

Norton Commander proqramını işə salmaq üçün NC əmrini yığmaq kifayətdir. Bu komanda verildikdən sonra ekranın yuxarı hissəsində ikiqat çərçivə ilə məhdudlaşdırılmış iki düzbucaqlı pəncərə (və ya panel) yaradılır. Bu pəncərələrdə cari diskovoddakı cari kataloq haqqında məlumatlar verilir. Pəncərələrdən aşağıda DOS əməliyyat sisteminə dəvət olur və burada bu sistemin adi əmrlərini vermək olar. Bu sətirdən də aşağıda isə NC proqramının funksional düymələrinin mənasını açıqlayan sətir yerləşir.

NC proqramından çıxmaq üçün *F10* düyməsini basmaq lazımdır. Bu zaman ekranın mərkəzində NC - dan çıxışı təsdiq edən sorğu aparılır, bu sorğunu təsdiqləmək, yəni NC-dan çıxmaq üçün *Enter* düyməsini, əks halda yəni NC-da qalmaq üçün isə *Esc* düyməsini basmaq lazımdır.

Köməyin alınması. NC-nın daxilində olan arayışlar siyahısını ekrana çıxarmaq üçün *F1* düyməsini basmaq lazımdır. Əgər bu zaman biz nəzərdən keçirmə və ya faylların redaktə edilməsi rejimində olarıqsa, onda ekrana funksional düymələrin təyinatı haqqında məlumat veriləcək. Digər hallarda isə cari iş rejimi və s. haqqında məlumatlar verilir. Belə ki, məsələn, *F5* düyməsini (faylın təkrarının alınması) basıb, sonra *F1* düyməsini basaq, onda ekrana faylın təkrarının alınması haqqında arayış veriləcək.

Əgər verilən arayışlar ekrana tam yerləşmirsə, onda klaviaturadakı ↑, ↓, *Home, End, PgUp, PgDn* düymələrinin köməyi ilə arayışları «vərəqləmək» olar. Arayışlar siyahısından çıxmaq üçün isə *Esc* düyməsini basmaq lazımdır.

Menyudan istifadə qaydaları. NC-nın əsas və ya idarəedici menyusunun köməyi ilə, informasiyanın ekranda ifadəsi üçün ən əlverişli

variantı vermək, NC-nın iş rejimlərini dəyişdirmək və bəzi digər əməliyyatları yerinə yetirmək olar.

Menyuya daxil olmaq üçün *F9* düyməsini basmaq lazımdır. Bu zaman ekranın yuxarı sətrində mənyunun Left, Files, Disk, Commands, Options və Right bəndləri olan sətir verilir. Menyunun bu bəndlərindən biri qeyd edilmiş olur. Menyunun bəndlərini dəyişdirmək üçün → və ← düymələrindən istifadə olunur. Menyunun lazımi bəndini seçib, *Enter* və ya ↓ düyməsini basdıqda, bu bəndin altında həmin bəndə aid alt menyu açılır. Alt menyuda lazımi bəndi seçmək üçün ↑ və ↓ düymələrindən istifadə olunur. Alt menyudakı lazımi bəndi seçib *Enter* düyməsini basdıqda həmin bənd yerinə yetirilir.

NC-nın menyusundan çıxmaq üçün *Esc* düyməsini basmaq lazımdır.

NC-nın panelləri və funksional düymələri. NC-nın panellərində aşağıdakılar verilə bilər:

1) Diskdəki kataloqun mündəricatı. Panelin yuxarı hissəsində həmin kataloqun adı verilir.

2) Diskdəki bütün kataloqların ağacı. Panelin yuxarı hissəsində *«Tree»* («kataloqlar ağacı») sözü verilir.

3) Yanaşı paneldə verilmiş disk və kataloq haqqındakı informasiya. Panelin yuxarısında *«Info»* (İnformasiya) sözü verilir.

4) Yanaşı paneldə verilmiş faylın tərkibi. Panelin yuxarısında adətən *«View»* («Nəzərdən keçirmək») sözü verilir.

5) Verilmiş kompüterlə, birləşdirilmiş digər kompüterin diskindəki kataloqun mündəricatı. Panelin yuxarısında *«Link:»* («Əlaqə:») sözü verilir.

6) Arxiv faylının mündəricatı. Panelin yuxarısında arxivin tipi və adı verilir.

7) Yanaşı paneldə verilmiş kataloqun qeyd edilmiş alt kataloq və faylların sayı və həcmi haqqında məlumatlar Panelin yuxarısında *«Directory Information»* («Kataloqun pasportu») sözü verilir.

8) Faylın axtarılmasının nəticələri. Panelin yuxarısında *«Find File»* («Faylın axtarışı») sözü verilir.

NC-nın panellərinin idarə edilməsi. Əgər ekranda NC-nın iki paneli verilmişsə, onda bu panellərdən biri aktiv və ya cari panel,

digəri isə qeyri-aktiv panel adlanır. Aktiv panelin sərlövhəsi, yəni panelin yuxarı hissəsində verilən mətn, panelin digər sətirlərinə nisbətən daha çox işıqlandırılmaqla qeyd edilir. NC-nın praktiki olaraq bütün əməliyatları aktiv paneldə yerinə yetirilir. Məsələn, kursorun yerini dəyişdirmək düymələrinin işi yalnız aktiv panelə təsir edir, yalnız aktiv paneldən olan faylları funksional düymələri basmaqla, nəzərdən keçirə, redaktə edə, təkrarını ala bilər və s. etmək olar.

NC-nın panellərini idarə etmək üçün düymələrin təyinatı:

Tab – yanaşı paneli aktiv panel etmək üçün;

Ctrl və O – panelləri ekrandan çıxarmaq və ya panelləri ekrana qaytarmaq üçün;

Ctrl və R – qeyri-aktiv paneli ekrandan çıxarmaq və ya həmin paneli ekrana qaytarmaq üçün;

Ctrl və U – panellərin yerlərini dəyişdirmək üçün;

Ctrl və F1 – sol tərəfdəki paneli ekrandan çıxarmaq və ya onu ekrana çıxarmaq üçün;

Ctrl və F2 – sağ tərəfdəki paneli ekrandan çıxarmaq və ya onu ekrana çıxarmaq üçün;

Alt və F1 – sol tərəfdəki paneldə digər diskin mündəricatını çıxarmaq üçün;

Alt və F2 – sağ tərəfdəki paneldə digər diskin mündəricatını çıxarmaq üçün.

Əgər aktiv paneldə verilmiş informasiya, paneldə tam yerləşmirsə, onda kursorun yerdəyişmə düymələri vasitəsilə, bu informasiyanı nəzərdən keçirmək olar. Belə ki, *PgUp* və *PgDn* düymələri, uyğun olaraq mətni bir səhifə yuxarı və bir səhifə aşağı endirir, ↑ və ↓ düymələri isə uyğun olaraq mətni kursor üzrə bir sətir yuxarı və bir sətir aşağı salmağa imkan verir.

Adətən NC ilə iş vaxtı onun panellərində kataloqların mündəricatı verilir. Lakin bəzən digər tip panellərdən – faylın axtarışının nəticələri olan, kataloqların ağacı verilən və s. panellərdən istifadə etmək lazım olur. Panelin tipini dəyişdirmək üçün menyunun, sol panel üçün Left (sol) və sağ panel üçün Right (sağ) əmrlərindən istifadə etmək olar.

Fayl və kataloqlar haqqında məlumatın verilməsi. NC fayl və alt kataloqlar haqqındakı informasiyanı panellərdə müxtəlif qaydalarda verə bilir. İnformasiyanın cari paneldəki verilmə qayda-

larını dəyişdirmək üçün *Ctrl və F3 – Ctrl və F7* düymələrinin kombinasiyalarından aşağıdakı kimi istifadə olunur:

Ctrl və F3 – əlifba sırası ilə fayl adları üzrə çeşidlənməsi.

Ctrl və F4 – əlifba sırası ilə fayl ad genişləndirilmələri üzrə çeşidlənməsi.

Ctrl və F5 – fayl və kataloqların yaradıldığı tarix və zamanın azalma sırası üzrə çeşidlənməsi.

Ctrl və F6 – faylların ölçülərinin azalma sırası üzrə çeşidlənməsi.

Ctrl və F7 – fayl və alt kataloqlarla diskdəki kataloqda yazıldığı qayda üzrə verilməsi.

Axırıncı haldan başqa, bütün yuxarıda verilən hallarda, paneldə əvvəlcə alt kataloqlar, sonra isə fayllar haqqında məlumatlar verilir. Əgər cari paneldə disketdəki kataloqun mündəricatı verilibsə və ekranı bu informasiyadan təmizləmək lazımdırsa, *Ctrl və R* düymələrini basmaq lazımdır. Bu əməliyyatı, misal üçün digər disketdən istifadə edərkən aparmaq vacibdir.

Qeyd edilmiş fayl və ya kataloq. Əgər ekranda fayl və ya kataloqlardan biri boz rəng ilə qeyd edilibsə, belə fayl və ya kataloqa qeyd edilmiş deyəcəyik. Burada ↑, ↓, ←, →, *PgUp* və *PgDn* düymələrinin köməyi ilə, qeyd edilmiş hissənin yerini dəyişməklə, ekranda verilmiş ixtiyari fayl və ya kataloqu qeyd edilmiş şəklinə gətirə bilərik.

Tab düyməsi vasitəsilə ekranın qeyd edilmiş hissəsini NC-nın bir panelindən, digər yanaşı panelinə keçirmək olar və qeyd edək ki, bu zaman həmin panel aktiv və ya cari olacaqdır.

Faylı tez qeyd etmək tələb olunursa, onda Alt düyməsini basıb, bu düyməni buraxmadan lazımi faylın adının birinci gələn hərflərini yığmaq lazımdır. Bu hərflər panelin aşağısında çərçivə daxilində veriləcəkdir. Faylın adında iştirak edən kifayət sayda hərflər yığılan kimi NC lazımi faylı qeyd edəcəkdir. Adı həmin hərflərlə başlayan digər faylı qeyd etmək üçün isə *Ctrl* və *Enter* düymələrini basmaq kifayətdir. Əgər faylı qeyd etməyə ehtiyac yoxdursa, *Esc* düyməsini basmaq lazımdır.

Başqa kataloqa keçid. NC-nın panelinə başqa kataloqun mündəricatını çıxarmaq üçün aşağıdakı üsullardan birindən istifadə etmək olar:

1) Əgər paneldə kataloqun adını qeyd edib, *Enter* düy-

məsini bassaq, onda NC bu kataloqun mündəricatını ekrana çıxarar;

2) Əgər qeyri-aktiv paneldə, aktiv paneldə olan diskə keçsək, qeyri-aktiv paneldə də, aktiv paneldə olan kataloqun mündəricatı veriləcək;

3) *Alt və F10* düymələrini basıb, ekrana kataloqların ağacı olan pəncərəni çıxarıb, bu pəncərədə lazımi kataloqu qeyd edib, *Enter* düyməsini basmaqla, panelə qeyd edilmiş kataloqun mündəricatını çıxarmaq olar.

Başqa diskə keçid. NC-nın panelində başqa diskdəki cari kataloqun mündəricatını çıxarmaq üçün sol panel üçün *Alt* və *F1* düymələrini, sağ panel üçün *Alt* və *F2* düymələrini basmaq lazımdır. Bu zaman ekrana daxil olmaq mümkün olan disklərin siyahısı veriləcək. Bu siyahıda lazım olan diskin adını → və ← düymələri ilə qeyd edib, *Enter* düyməsini basmaq lazımdır. Nəticədə NC qeyd olunmuş diskin cari kataloqunun mündəricatını oxuyub, onu ekrana çıxaracaq.

Məlumat paneli. NC-da bir paneldə, digər paneldə ifadə olunmuş, disk və kataloq haqqındakı informasiyanı vermək olar. Əgər *Ctrl* və *L* düymələri basılsa, onda qeyri-aktiv panel məlumat panelinə çevriləcək. Bu paneli əvvəlki vəziyyətinə qaytarmaq üçün isə, yenidən *Ctrl* iə *L* düymələriini basmaq kifayətdir.

Məlumat panelinin yuxarısında «Info» sətri veriləcək və paneldə aşağıdakı məlumatlar veriləcək:

1) Kompüterin operativ yaddaşının baytlarla tutumu (...Bytes Memory);

2) Operativ yaddaşda boş olan yerin baytlarla tutumu (...Bytes Memory);

3) Cari diskin baytlarla tutumu (...bytes on drive...);

4) Cari diskdə boş olan yerin baytlarla tutumu (...bytes free on drive...);

5) NC-nın qonşu panelində verilmiş fayl və kataloqların sayı və onların baytlarla ümumi ölçüsü (...files use...bytes in...);

6) Cari diskin nişanı (volume label);

7) Cari diskin sıra nömrəsi (serial number).

Funksional düymələrdən istifadə qaydaları.
F1 – F10 düymələri

NC ekranın aşağı hissəsində funksional düymələrin məqsədi haqqında məlumat verir. Bu funksional düymələrin təyinatı aşağıda qısa olaraq verilir:

Düymə	*Yazılışı*	*Təyinatı*
F1	Help (Kömək)	NC ilə iş zamanı düymələrin məqsədi haqqında qısa məlumat
F2	Menu (Çağırış)	Menyunun siyahısındakı əmrlərin verilməsi
F3	View (Oxumaq)	Faylların nəzərdən keçirilməsi
F4	Edit (Redaktə etmək)	Faylların redaktə edilməsi
F5	Copy (Təkrarını almaq)	Faylların təkrarının alınması. Bu ekranın mərkəzində faylın təkrarının haraya göndəriləcəyi soruşulur. Yer göstərilmədikdə təkrar, digər paneldəki kataloqa göndərilir. Faylın təkrarını almaq üçün sonra *Enter* düyməsini, əmri ləğv etmək üçün isə *Esc* düyməsini basmaq lazımdır.
F6	Renmov (Yeni ad)	Faylın (kataloqun) adının dəyişdirilməsi və ya faylın digər kataloqa göndərilməsi. Bu əməliyyatları başlamaq üçün *Enter* düyməsini, əmri ləğv etmək üçün *Esc* düyməsini basmaq lazımdır.
F7	Mkdir (Yeni kataloq)	Alt kataloqun yaradılması
F8	Delete (Ləğv etmək)	Fayl və kataloqların ləğv edilməsi
F9	PullDn (Menyu)	NC-nın idarəedici menyusunun çağırılması
F10	Quit (Çıxış)	NC-dan çıxış

Alt və *F1* – *Alt* və *F10* düymələrinin kombinasiyası

Əgər *Alt* düyməsini bassaq, onda ekranın sonuncu sətri dəyişəcəkdir. Bu sətirdə *Alt* və *F1* – *Alt* və *F10* düymələrinin kombinasiyasının təyinatı haqqında məlumat verilir. Bu düymələrin kombinasiyasının təyinatı aşağıdakı kimidir:

Düymə	*Yazılışı*	*Təyinatı*
Alt və *F1*	Left (Disk)	Sol paneldə təsvir olunan diskin seçilməsi
Alt və *F2*	Right (Disk)	Sağ paneldə təsvir olunan diskin seçilməsi
Alt və *F3*	View...(Oxumaq)	Mətn faylının oxunması. Bu iş rejimi *F3* düyməsi ilə verilən rejimdən fərqli olaraq yalnız mətni fayların oxunması üçün nəzərdə tutulub
Alt və *F4*	Edit...(Yoxlama)	Faylın redaktə edilməsi (*F4* düyməsi ilə verilən redaktor NC-nın daxilində olan redaktordur)
Alt və *F5*	Comp (Sıxmaq)	Qeyd olunmuş faylların sıxılması (onların arxivə yerləşdirilməsi)
Alt və *F6*	DeComp (Çıxarmaq)	Qeyd olunmuş arxivdən faylların çıxarılması
Alt və *F7*	Find (Axtarış)	Faylın diskdə axtarılması
Alt və *F8*	History (Jurnal)	Əvvəlcədən verilmiş əmrlərin nəzərdən keçirilməsi və yenidən yerinə yetirilməsi.
Alt və *F9*	EGALn (Sətirlər)	Ekranda sətirlərin sayını EGA monitoru üçün 25-dən 43-ə qədər və VGA monitoru üçün isə 50-yə qədər dəyişdirmək və əksinə
Alt və F10	Tree (Ağac)	Digər kataloqa tez keçid

Ctrl və F1 – Ctrl və F10 düymələrinin kombinasiyası

Əgər Ctrl düyməsi basılarsa, onda ekranın sonuncu sətri dəyişəcəkdir. Bu sətirdə Ctrl və F1 – Ctrl və F10 düymələrinin kombinasiyasının təyinatı haqqında məlumatlar verilir. Bu düymələrin kombinasiyalarının təyinatı aşağıdakı kimidir:

Düymə	*Yazılışı*	*Təyinatı*
Ctrl və *F1*	Left (Sol)	Sol paneli ekrana çıxarmaq və ya oradan yığışdırmaq üçün
Ctrl və *F2*	Right (Sağ)	Sağ paneli ekrana çıxarmaq və ya oradan yığışdırmaq üçün
Ctrl və *F3*	Name (Ad)	Cari paneldə faylların adları üzrə çeşidlənməsi
Ctrl və *F4*	Exten (Tip)	Cari paneldə faylların ad genişləndirilməsi üzrə çeşidlənməsi
Ctrl və *F5*	Time (Zaman)	Cari paneldə faylların onların yaradıldığı zaman üzrə çeşidlənməsi
Ctrl və *F6*	Size (Ölçü)	Cari paneldə faylların onların ölçüləri üzrə çeşidlənməsi
Ctrl və *F7*	UnSort (Disk)	Cari paneldə faylların çeşidlənməsi
Ctrl və *F8*	Sync (Sinxro)	NC-nın hər iki panelindəki kataloqların mündəricatının sinxronlaşdırılması
Ctrl və *F9*	Print (Çap etmək)	Qeyd olunmuş fayl və ya fayllar qrupunun çap edilməsi
Ctrl və *F10*	Split (Bölmək)	Faylın bir neçə hissəyə bölünməsi (məsələn, faylın disketə yazılması üçün) və ya bir neçə fayl qeyd edilmişsə, bu faylların bir faylda birləşdirilməsi.

Shift və F1 – Shift və F10 düymələrinin kombinasiyası

Əgər Shift düyməsini bassaq, onda ekranın sonuncu sətri dəyişməyəcəkdir, bu sətir həmin düymə basılmadığı halda olan

şəkildə qalacaqdır. Lakin funksional düymələrin, *Shift* düyməsi ilə kombinasiyasının təyinatı başqadır:

Düymə	*Yazılışı*	*Təyinatı*
Shift və *F1*	Help (Kömək)	Diskin təmizlənməsi
Shift və *F2*	Menu (Çağırış)	Şəbəkə utilitləri
Shift və *F3*	View (Oxumaq)	Faylın nəzərdən keçirilməsi (faylın adı soruşulur)
Shift və *F4*	Edlit (Redaktə etmə)	Faylın redaktə edilməsi (faylın adı soruşulur)
Shift və *F5*	Copy (Təkrarın alınması)	Faylın təkrarının alınması (faylın adı və məqsədi soruşulur)
Shift və *F6*	Renmov (Yeni ad)	Faylın adının dəyişdirilməsi və ya onun göndərilməsi (faylın adı və məqsədi soruşulur)
Shift və *F7*	MkDir (Yeni kataloq)	Alt kataloqun yaradılması
Shift və *F8*	Delete (Çıxarmaq)	Faylın çıxarılması (faylın adı soruşulur)
Shift və F9	PullDn (Menyu)	NC-nın quruluşunun saxlanılması
Shift və *F10*	Quit (Çıxış)	Menyunun çağırılması

NC-da fayllarla iş qaydaları. NC fayllar və kataloqlar qrupunu seçib, onlar üzərində bəzi işlər, məsələn, onların təkrarını almaq, digər kataloqa keçirmək, ləğv etmək və s. aparmaq olar. Ayrıca fayl və ya kataloqu seçmək üçün Ins düyməsini basmaq lazımdır. Həmin düymənin təkrarən basılması həmin fayl və ya kataloqun seçilməsini ləğv edir.

Fayllar qrupunu şablon üzrə seçmək üçün klaviaturanın sağ tərəfindəki "+" düyməsini basıb, seçki şablonunu vermək lazımdır. Şablonda DOS əmrlərində olduğu kimi "*" və "?" işarələrindən istifadə edilir. Məsələn, *.doc* ad genişlənməsinə malik bütün faylları seçmək üçün "+" düyməsini basıb, **.doc* şablonunu klaviaturadan yığıb, *Enter* düyməsini basmaq lazımdır.

Fayllar qrupunu şablon üzrə seçilməsini ləğv etmək üçün klaviaturanın sağ tərəfindəki "–" düyməsini basıb, seçilməsi ləğv

edilən faylların şablonunu vermək lazımdır. Məsələn, bütün faylların seçilməsini ləğv etmək üçün "–" düyməsini basıb, *.* şablonunu yığıb, *Enter* düyməsini basmaq lazımdır.

Qeyd olunmamış faylları qeyd etmək üçün və əksinə qeyd olunmuşları isə qeyd olunmamış etmək üçün klaviaturanın sağ tərəfindəki * düyməsini basmaq lazımdır. Bu düymə basıldıqdan sonra alt kataloqlar qeyd edilməmiş olacaqlar.

Fayllar və kataloqların seçilmiş qrupu ilə funksional düymələrin köməyi ilə aşağıdakı əməliyyatları aparmaq olar:

Düymə	*Yazılışı*	*Təyinatı*
F5	Copy (Təkrarı)	Başqa kataloqa faylın təkrarının göndərilməsi
F6	Ren Mov (Yeni ad)	Faylı digər kataloqa göndərmək və ya adını dəyişdirmək
F8	Delete (Ləğv etmək)	Faylı ləğv etmək
Alt və *F5*	Comp (Sıxmaq)	Faylı arxivə göndərmək
Alt və *F6*	De Comp (Çıxarmaq)	Qeyd olunmuş arxivlərdən faylları çıxarmaq
Ctrl və *F10*	Split (Bölmək)	Qeyd olunmuş faylları bir faylda birləşdirmək

Faylların nəzərdən keçirilməsi. *F3* düyməsini basmaqla, NC kursorla qeyd olunmuş faylı nəzərdən keçirməyə imkan verir. Nəzərdən keçirilən fayl üzrə yerdəyişmələr etmək üçün ↑, ↓, *PgUp*, *PgDn*, ← və → düymələrindən istifadə edilir. *Home* və *End* düymələri isə uyğun olaraq nəzərdən keçirilən faylın əvvəlinə və sonuna keçməyə imkan verir.

NC-da faylların redaktə edilməsi. Kursor ilə qeyd olunmuş faylı redaktə etmək üçün *F4* düyməsini basmaq lazımdır. Faylları redaktə etmək üçün NC-nın daxili redaktorundan, eləcə də ixtiyari digər redaktordan istifadə etmək olar. Redaktor NC-nın menyusundakı Commands, Configuration, Editor (Komandalar, Konfiqurasiya, Redaktor) bölmələrindən istifadə edilməklə seçilir.

Əgər *Shift* və F4 düymələrini basdıqdan sonra diskdə olmayan faylın adını daxil etsək, bu adlı yeni fayl yaradılır.

Əgər faylın adını kursorla qeyd edib, *F4* düyməsi əvəzinə

Alt və *F4* düymələrini bassaq, onda qeyd olunmuş faylı digər bir redaktorla redaktə etmək olar, yəni *F4* düyməsi basıldıqda fayl, NC-nın daxili redaktoru, *Alt* və *F4* düymələri basıldıqda isə istifadəçinin seçdiyi digər bir redaktorla redaktə edilir.

NC-nın daxili redaktoru vasitəsilə redaktə etmə qaydaları. Burada kursor mətndəki cari mövqeni göstərir. Mətndəki bütün dəyişikliklər, kursorun göstərdiyi mövqedən başlayaraq yerinə yetirilir. Kursoru, mətn üzrə bir mövqe sola, sağa, yuxarı və aşağı hərəkət etdirmək üçün uyğun olaraq ←, →, ↑, ↓ düymələrini basmaq lazımdır. Bunlardan əlavə kursoru mətn üzrə aşağıdakı düymələrin vasitəsilə hərəkət etdirmək olar:

PgUp və *PgDn* ekran ölçüsündə bir səhifə mətni uyğun olaraq yuxarı və aşağı hərəkət etdirmək; *Ctrl* ilə ← və *Ctrl* ilə → mətni uyğun olaraq bir söz sola və sağa hərəkət etdirmək üçün;

Home və *End* – mətnin cari sətrinin uyğun olaraq əvvəli və sonuna keçmək üçün;

Ctrl ilə *Home* və *Ctrl* ilə *End* – uyğun olaraq redaktə edilən faylın əvvəli və sonuna keçmək üçün;

Alt ilə *F8* – mətnin verilmiş nömrəli (nömrə soruşulur) sətrinə keçmək üçün.

Mətni daxil etmək üçün, əvvəlcə kursoru mətnin daxil ediləcəyi yerə gətirib, mətni, uyğun hərf-rəqəm düymələrini basmaqla yığmaq lazımdır. Sətrin sonunu bildirmək və onu sona çatdırmaq üçün *Enter* düyməsini basmaq lazımdır.

Əgər mətn yığarkən, klaviaturanın yuxarı reqistrindəki simvollardan istifadə etmək lazımdırsa (məsələn, istifadə olunan əlifbanın böyük hərflərindən), onda *Shift* düyməsini basıb, onu buraxmadan lazımi simvollu düyməni basmaq kifayətdir.

Əgər iş zamanı bir əlifbadan digərinə keçmək lazımdırsa (məsələn, latın əlifbasından rus əlifbasına və əksinə), onda klaviaturanın bir əlifba hərfləri rejimindən, digər əlifba hərfləri rejiminə keçidini yerinə yetirmək lazımdır. Bu isə klaviaturanın uyğun drayveri vasitəsilə yerinə yetirilir və onu yerinə yetirmək üçün müxtəlif cür düymə kombinasiyaları verilə bilər (məsələn, çox vaxt rejimlərin dəyişdirilməsi *Shift* düyməsinin təkrarən iki dəfə basılması ilə yerinə yetirilir).

Daxili redaktor, kursorun göstərdiyi mövqedən, cari zaman və tarixin verilməsinə imkan verir. Bunun üçün *Alt* ilə *F3* düy-

mələrini basmaq lazımdır.

Sətir və simvolların pəzulması qaydaları. Sətir və simvolları, mətndən çıxarmaq üçün aşağıdakı düymələrdən istifadə etmək olar:

Del – kursorun olduğu mövqedən simvolun çıxarılması üçün;

Backspace (*Enter* düyməsindən yuxarıda yerləşən və üstündə sola istiqamətlənmiş ox olan düymə) – kursordan solda duran simvolun çıxarılması üçün;

Ctrl ilə *Y* – sətrin çıxarılması üçün;

Ctrl ilə *K* – kursorun cari mövqeyindən sətrin sonuna kimi mətnin çıxarılması üçün;

Ctrl ilə *Backspace* – kursordan solda duran sözün çıxarılması üçün;

Ctrl ilə *T* – kursordan sağda duran sözün çıxarılması üçün;

Fayllar üzərində əməliyyatlar. Redaktorda fayllar üzərində əməliyyatlar aşağıdakı düymələrin köməyi ilə yerinə yetirilir:

F2 – redaktə edilmiş faylın EHM-in yaddaşında saxlanılması üçün;

Shift ilə *F2* – redaktə edilmiş faylın yaddaşda digər adla saxlanması üçün (yeni ad soruşulur);

F10 və ya *Esc* – redaktə etmə rejimindən çıxış üçün *Shift* ilə *F10* –redaktə edilmiş faylı yadda saxlamaqla, redaktə etmə rejimindən çıxış üçün;

F9 – faylın çap üçün printerə çıxarılması üçün;

Alt ilə *F5* – redaktə olunan faylın tərkibinə digər faylın daxil edilməsi üçün (faylın daxil edilməsi kursorun göstərdiyi mövqedən başlanır);

Alt ilə *F9* – redaktə edilmiş faylı yaddaşda saxlayarkən, bu faylların təkrarı olan ehtiyat *.bak* fayllarının yaradılmasına ehtiyac olub, olmadığını təyin edir.

Mətn blokları ilə əməliyyatlar. NC-nın daxili redaktoru mətn bloklarını ayırıb, onlar üzərində müəyyən əməliyyatlar aparmağa imkan verir. Mətn blokları bir və ya bir neçə ardıcıl gələn sətirlərdən ibarət olur. Mətn blokunu ayırmaq üçün kursoru blokun birinci va ya axırıncı sətri üzərinə qoyub, F3 düyməsini basmaq lazımdır. Sonra isə kursoru mətn blokunun digər uc sərhəd sətrinə gətirib *F3* düyməsini birdə basmaq lazımdır.

Ayırılmış blok üzərində əməliyyatlar aparmaq üçün aşağı-

dakı düymələrdən istifadə olunur:

Shift ilə *F3* – mətn blokunun ayrılmasını ləğv etmək üçün;

F5 – mətn blokunu, kursorun qarşısındakı mövqeyə təkrarlamaq üçün;

F6 – mətn blokunu, kursorun qarşısındakı mövqeyə gətirmək üçün;

F8 – mətn blokunu çıxarmaq üçün;

Alt ilə *F10* – mətn blokunu, hər hansı bir fayla əlavə etmək üçün (bu zaman faylın adı soruşulur, əgər bu cür fayl yoxdursa, həmin fayl yaradılır)

Redaktə etmə rejimindən çıxış. Faylın redaktə etmə rejimindən istifadə etmək olar:

F10 və ya *Esc* – redaktə etmə rejimindən çıxış üçün;

Shift ilə F10 – redaktə olunan faylı saxlamaqla, redaktə etmə rejimindən çıxış üçün.

Fayl və kataloqların təkrarının alınması. NC ilə faylların təkrarını almaq üçün lazımi faylı və ya fayllar qrupunu qeyd edib, *F5* düyməsini basmaq lazımdır. Fayl və ya fayllar qrupunun təkrarını alarkən, bu təkrarın göndəriləcəyi kataloqun mündəricatını, bu zaman qeyri-aktiv panel olan yanaşı panelə çıxarmaq məsləhət olunur. *F5* düyməsini basdıqdan sonra ekranın mərkəzində fayl (fayllar) və kataloqun (kataloqlar) təkrarının hara köçürmək lazım olduğu barədə sorğu gəlir. Həmin sorğunun gəldiyi mətn sahəsində *Copy*: sözünün altında aşağıdakılardan birini yazmaq lazımdır:

1) fayl təkrarının göndəriləcəyi kataloqun adını;

2) fayl və ya kataloqa, onun təkrarının yerləşdiriləcəyi yeni ad verməli.

Sorğu mətn sahəsində təkrarın göndəriləcəyi yerin adı olan sətirdən aşağıda, təkrarın alınmasının 4 mümkün rejimi verilir, bu rejimlərdən birini seçmək üçün kursoru həmin rejimin qarşısındakı « [ ] » işarəsi üzərinə qoyub, probel düyməsini basmaq lazımdır. Fayl və kataloqların təkrarının alınmasını yerinə yetirmək üçün, təkrarlamanın rejimlərini təyin edən sətirdən aşağıda verilmiş *Copy* (*Yerinə yetirmək*) düyməsini basmaq (yəni bu sözü qeyd edilmiş edib, Enter düyməsini basmaqla) lazımdır. Bundan əvvəl isə sorğu mətn sahəsində təkrarın göndəriləcəyi yeri və təkrarlama rejimini təyin etmək lazımdır.

Fayl və kataloqların adlarının dəyişdirilməsi və onların bir yerdən digər yerə göndərilməsi. Fayl (fayllar) və kataloqların adlarını dəyişdirmək üçün, adları dəyişdirilən fayl və kataloqların adlarını kursorla qeyd edib *F6* düyməsini basmaq lazımdır. NC-nın sorğusuna cavab olaraq fayl və ya kataloqların yeni adlarını verib, sorğudakı Rename/Move (Yerinə yetir) düyməsini basmaq lazımdır.

NC eləcə də fayl və kataloqların bir kataloqdan digərinə göndərməyə imkan verir. NC-da faylların bir yerdən digər yerə göndərilməsi, faylların təkrarının alınmasından yalnız onunla fərqlənir ki, bu əməliyyət aparıldıqdan sonra göndərilən fayl və ya kataloq ləğv edilir. NC-da faylların göndərilməsi ilə onların təkrarının alınması arasında aşağıdakı fərqlər var:

1) faylların göndərilməsini başlamaq üçün *F6* düyməsini basmaq lazımdır;

2) faylların göndərilməsi rejimində sorğu Rename (Faylların adlarının dəyişdirilməsi) başlığına malikdir, göndərişi başlamaq üçün sorğudakı əmr düyməsi isə Rename/Move adlanır.

Fayl və kataloqların ləğv edilməsi. NC-nın köməyi ilə fayl və kataloqları yaddaşdan çıxarmaq, ləğv etmək üçün lazımi fayl və ya kataloqu (fayl və kataloqlar qrupunu) qeyd edib, *F8* düyməsini basmaq lazımdır. Bu zaman, əgər paneldə hansı isə fayl və kataloqlar qeyd edilibsə, onda seçilmiş bu fayl və kataloqlar qrupu çıxarılır, əks halda isə kursorun durduğu yerdə olan cari fayl və ya kataloq ləğv edilir.

Cari fayl və ya kataloqu çıxararkən, kursor ilə onu qeyd edib, *F6* düyməsini basmaq kifayətdir. Bu zaman ekrana, qeyd olunmuş fayl adı olan və faylın çıxarılması haqqında sorğu olan mətn sorğu çərçivəsi verilir. Faylı çıxarmaq üçün sorğu çərçivəsindəki *Delete* (Çıxarmaq) sorğu düyməsini basmaq, yəni həmin sorğu düyməsini aktiv edib, *Enter* düyməsini basmaq lazımdır.

Əgər qeyd olunmuş bir qrup fayl və kataloqlar yaddaşdan çıxarılırsa, bu zaman sorğu çərçivəsində, onların adları deyil, sadəcə olaraq fayl və kataloqların sayı göstərilir, və hər bir çıxarılan fayl və ya kataloqun çıxarılmasını təsdiqləyən sorğu çərçivəsi ekrana verilir. Bu zaman sorğu çərçivəsindəki dörd mümkün cavab düyməsindən birini seçib, onu aktiv edib *Enter* düyməsini basmaq lazımdır:

1) *Delete* (*Çıxarmaq*) – sorğudakı verilmiş adlı faylı çıxarmaq və işi davam etdirərək, digər çıxarılan fayllar haqqında sorğunun verilməsi;

2) *All* (*Hamısı*) – bu qeyd olunan və sonrakı qeyd olunmuş faylların (kataloqlardan başqa) sorğusuz ləğv edilməsi;

3) *Skip* (*Buraxmaq*) – bu adı çəkilən faylı ləğv etmədən işi davam etdirərək digər qeyd olunmuş faylların çıxarılması haqqında sorğunun verilməsi;

4) *Cancel* (*Dayandırmaq*) – qeyd olunmuş faylı ləğv etməmək, faylların çıxarılması prosesini dayandırmaq.

Diskdə faylların axtarılması. Axtarışa başlamaq üçün aşağıdakı əməlləri yerinə yetirmək lazımdır:

1) faylların axtarışı, yerinə yetiriləcək diski cari etməli
2) faylların axtarılacağı kataloqu cari etməli
3) *Alt* ilə *F7* düymələri basmalı

Bu zaman ekranda faylların axtarış parametrləri haqqında sorğu veriləcək. Axtarışın bütün uyğun parametrlərini verib, sorğudakı Start (başlamaq) düymə-sini aktiv edib, *Enter* düyməsini basmaq lazımdır. Sorğu çərçivəsində, faylların axtarışı üçün aşağıdakı mümkün parametrlər haqqındakı sahələr verilir:

1. *Find Files* (*faylı tap*) – burada axtarılan faylların adını və ya faylların bir-birindən probellə ayrılan adlarını verməli;

2. *Location* (*kataloq*) – burada proqram cari olan kataloqun adını verir. Bu sahədə, kataloqun adını, istifadəçiyə, faylların axtarılması üçün lazım olan digər kataloq və kataloqların adı ilə əvəz etmək olar. Kataloqların sayı bir neçə olduqda, onların adları bir-birindən ya probellə, ya da nöqtəli vergüllə ayrılır. Axtarış bütün disk üzrə aparıldıqda bu sahə nəzərə alınmır;

3. *Containing* (*tərkibi*) – bu sahədə, axtarılan fayllarda olan simvolları göstərmək olar.

Sorğu çərçivəsinin *Search Locations* (axtarış yeri) düzbucaqlısında aşağıdakı mümkün rejimlərindən birini seçmək lazımdır, bunun üçün kursoru seçilmiş rejim üzərinə gətirib, probel düyməsini basmaq lazımdır:

4. *Entire disk* (*bütün disk*) - axtarış bütün disk üzrə aparılır. Diskin adı Entire disk sözlərindən sağda verilir. Diskin adını dəyişdirmək tələb olunduqda *Drive* (*disk*) sorğu düyməsini basmaq (kursoru bu düymə üzərində qoyub *Enter* düyməsini bas-

maq) olar.

5. *Location(s) and above* (kataloq və aşağıdakılar) – axtarış Location(s) sahəsində verilmiş kataloq və onun bütün alt kataloqlarında aparılır.

6. *Location(s) only* (*Yalnız kataloqda*) – axtarış yalnız *Location(s)* sahəsində göstərilən kataloqda aparılır.

Axtarış yerinə yetirilərkən, ekranda aşağıdakılar verilir:

1. Çərçivənin yuxarı hissəsində, tapılmış faylların adları verilməyə başlanır;

Çərçivənin sağ tərəfində (*Quit FF* (*çıxış*) düyməsinin üstündə) tapılmış faylların sayı göstərilir;

2. Axtarış çərçivənin aşağı hissəsində beş əvvəlki düymə əvəzinə (start, disk, *F10*-ağac, genişlənmiş və çıxış düymələri) Stop (dayanmaq) – axtarışın sona çatdırılması, View (nəzərdən keçirmə) - qeyd olunmuş faylın nəzərdən keçirilməsi, *Goto* (*keçid*) – qeyd olunmuş fayl yerləşən kataloqa keçid, düymələri qalır.

Tapılmış fayllar siyahısını nəzərdən keçirmək üçün ↓, ↑, *PgUp* və *PgDn* düymələrindən istifadə edilir.

3. Faylların axtarışı zamanı, artıq tapılmış faylları nəzərdən keçirmək tələb olunduqda *F3* düyməsini və ya sorğudakı *View* düyməsini basmaq lazımdır. Bu zaman faylların axtarışı dayandırılır, fayl nəzərdən keçirildikdən sonra isə axtarış davam etdirilir.

4. Axtarış nəticəsində tapılmış faylın yerləşdiyi kataloqa keçmək üçün sorğudakı *Goto* düyməsini basmaq lazımdır. Bu zaman axtarış dayandırılır və qeyd olunmuş faylın yerləşdiyi kataloqa keçid yerinə yetirilir, bu fayl isə paneldə cari fayl, yəni kursorla qeyd olunmuş fayl olur.

Axtarışı vaxtından əvvəl dayandırmaq üçün ya *Esc* düyməsini, ya da sorğudakı *Stop* düyməsini basmaq lazımdır.

Faylların çapa verilməsi. Faylı çapa vermək üçün NC-nın panelində kursorun faylın adı üzərində qoyub, *Ctrl* ilə *F9* düymələrini basmaq lazımdır. Lakin bundan əvvəl çap qurğusunun iş vəziyyətinə gətirmək lazımdır. Bu zaman NC ekrana faylın çapa verilməsi haqqında sorğu verir, faylı çapa vermək üçün ya *Enter* düyməsini, ya da sorğudakı *OK* düyməsini basmaq lazımdır.

NC-da kataloqlarla iş qaydaları. Paneldə çıxarılmış kataloqda yeni alt kataloq yaratmaq üçün *F7* düyməsini basmaq

lazımdır. Bu zaman NC ekrana alt kataloqun adı haqqında sorğu çıxarır: *Create the directory* (*Kataloq yaratmaq*). *Alt* kataloqun adını yazıb *Enter* duyməsini basmaq lazımdır. *Alt* kataloqun yaradılmasını ləğv etmək üçün *Esc* düyməsini basmaq lazımdır.

Kataloqların təkrarının alınması, adının dəyişdirilməsi, bir yerdən digərinə göndərilməsi və ləğv edilməsi, NC-da fayllar üçün həmin işlərin aparıldığı qaydada yerinə yetirilir. Bu qaydalar isə yuxarıda fayllar üçün təsvir olunmuşdur.

NC-nın panelində diskdəki kataloqlar ağacını çıxartmaq olar və kursorun idarəetmə düymələri vasitəsilə kataloqlar ağacı boyunca hərəkət edib, lazımi kataloqu qeyd edib, onun tərkibini yanaşı digər paneldə nəzərdə keçirmək olar. Eləcə də klaviaturanın sağındakı "+" və "–"düymələri vasitəsilə qeyd olunmuş kataloqların daxilindəki alt kataloqları nəzərdən keçirmək olar.

Kataloqlar ağacını ekrana çıxartmaq üçün sol və ya sağ panelin seçilməsindən asılı olaraq, menyudakı *sol (left)* və ya *sağ (right)* qrupunda *ağac (tree)* əmrini seçmək lazımdır (bu kataloqlar ağacının sol və sağ panelə çıxarılmasından asılıdır). Ekranı əvvəlki iş rejiminə qaytarmaq üçün isə sol panel üçün *Alt* ilə *F1*, sağ panel üçün isə *Alt* ilə *F2* düymələrini basmaq sonra isə *Enter* düyməsini basmaq lazımdır.

Lazımi kataloqu tapdıqdan sonra onun üzərində aşağıdakı əməlləri aparmaq olar: kataloqlar ağacında cari kataloqları (kursorla qeyd edilmiş kataloqu) *F5* düyməsini basmaqla onun təkrarını almaq, *F6* düyməsini basmaqla adını dəyişdirmək və bir yerdən digərinə göndərmək, *F8* düyməsini basmaqla çıxartmaq, *Alt* ilə *F5* düymələrini basmaqla arxivə göndərmək, *F7* düyməsini basmaqla cari kataloqda alt kataloq yaratmaq olar.

Diskdəki kataloqlar ağacını nəzərdən keçirmək üçün, eləcə də diskdəki bir kataloqdan digərinə keçmək üçün *Alt* ilə *F10* düymələrini basmaqla ekrana kataloqlar ağacını çıxartmaq olar. Buradakı hər hansı kataloqa keçmək üçün kataloqlar ağacı verilən çərçivədə lazımi kataloqun adını kursorla qeyd edib, *Enter* düyməsini basmaq lazımdır.

Kataloqlar ağacı verilən çərçivədən aşağıdakı əməlləri yerinə yetirmək olar:

F7 (MkDir) düyməsini basmaqla, alt kataloq yaratmaq (alt kataloqun adını klaviaturadan yığmaqla);

F8 (Delete) düyməsini basmaqla, kataloqu çıxartmaq;

F6 (Rename) düyməsini basmaqla, katloqun adını dəyişmək;

F2 (Rescan) düyməsini basmaqla, diskdəki katloqlar haqqındakı informasiyanı almaq.

2.5. Windows əməliyyat sistemi

Windows əməliyyat sistemi – müxtəlif təyinatlı proqramların mürəkkəb kompleksidir. Bura Windows sisteminin iş rejimlərinin və onların müxtəlif parametrlərinin quraşdırılması proqramları, fayllarla iş proqramları, alət prqoramları və s. daxildir. Windows sisteminə daxil olan bəzi proqramları qeyd edək:

1) *Kalkulyator* (Calc.exe). Bu proqram onluq, ikilik, onaltılıq ədədlərlə işləməyə və onları birini digərinə keçirməyə imkan verir.

2) *Bloknot* (NotePad.exe). Kiçik mətni fayllarla iş üçün nəzərdə tutulmuş mətn redaktorudur.

3) *Lazer səsləndiricisi* (CdPlayer.exe). Kompakt disklərin səs fayllarını səsləndirməyə imkan verir.

4) *Fonoqraf* (Sndrec32.exe). Mikrofon vasitəsilə daxil edilən səs fayllarını yazmağa, səsləndirməyə və redaktə etməyə imkan verir.

5) *Bələdçi* (Explorer.exe). Fayllarla bütün mümkün əməliyyatları aparmağa, ixtiyari proqramları işə salmağa və s. imkan verir.

6) *Öyrədici proqram* (WinTutorial.exe). Windows-da işi öyrədən proqram.

Bunlardan əlavə ScanDisc proqramı bərk diskdəki ola biləcək səhvləri yoxlayır və aradan qaldırır, DriveSpace proqramı diskdəki verilənləri sıxlaşdırır və əlavə boş yerlər ayırır.

Windows-un tərkibinə WordPad mətn redaktoru, Paint qrafik redaktoru, elektron poçt, faksimil məlumatlar qəbul edib, göndərmək qabiliyyəti olan (Exchange.exe) proqram, İnternet şəbəkəsində işi təmin edən MS Explorer proqramı da daxildir.

Windows sistemində işlə bağlı bəzi əsas anlayışları verək:

1) **Sənəd anlayışı.** Yadda saxlanılması tələb olunan ixtiyari informasiya verilən adlanır. Bütün verilənlər isə proqramlara və

sənədlərə bölünür. Sənədlər – proqramlarla yaradılan və emal edilən verilənlərdir. Hər iki tip verilənlər fayllarda saxlanılır. Sadəlik üçün proqram saxlanılan fayl «proqram», sənəd saxlanılan fayl isə «sənəd» adlandırılır.

2) **Qovluq anlayışı.** Qovluq daxilinə digər qovluqlar, fayllar və digər obyektlər yerləşdirilə bilən konteyner rolunu oynayır.

3) **Pəncərə anlayışı.** Pəncərə ekranın düzbucaqlı çərçivə ilə məhdudlaşdırılmış və onu ayrıca ekran kimi işlətməyə imkan verən hissəsidir. Ekranda eyni zamanda bir neçə pəncərə yerləşdirilə bilər, lakin onlardan yalnız biri – cari (aktiv) adlanan pəncərə ilə işləmək olur. Hər bir açılan proqram, qovluq və sənəd öz pəncərəsində yerləşdirilir. Ekranda pəncərələrin ölçülərini, vəziyyətini və bir-birinə nəzərən qarşılıqlı vəziyyətini dəyişdirmək olur.

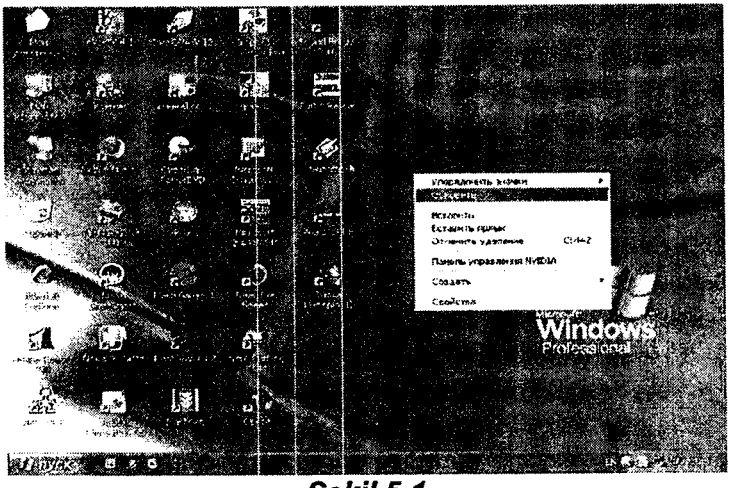
Windows sisteminin əsas elementləri aşağıdakılardır: fayl sistemi, istifadəçinin əməliyyat sistemi ilə ünsiyyətini təmin edən qrafik örtük, periferik qurğuların qoşulma və tənzimləmə sistemləri, əməliyyat sisteminin nizamlama proqramları, arayışlar sistemi, Windows sisteminə daxil olan tətbiqi və xidməti proqramlar külliyatı.

Fayl sistemi. Windows-da fayl sistemi MS DOS sistemində olan təyinata malikdir və onun genişlənməsidir. Fayl sisteminin əsas anlayışları fayl və qovluqdur. Fayl – artıq qeyd etliyimiz kimi ad ilə işarələnən və xarici yaddaş daşıyıcılarında (bərk və ya elastik disk, CD-ROM) saxlanılan verilənlər külliyatıdır. Windows-da fayl MS DOS-da verdiyimiz xassələri özündə saxlayır, bundan əlavə ümumiləşmə xatirinə buradakı qurğular da (printer, Lpt1, Lpt2, Com1 və s. qoşquları) fayl kimi qəbul olunur. Fayl MS DOS-da olduğu kimi əsas ad və ad genişlənməsi ilə ifadə olunur və xüsusi nişanla işarə edilir. Nişan – kiçik simvoldan, şəkildən ibarətdir. Windows-da hər bir obyektin (fayl, qovluq) adı öz nişanı ilə təmin edilir, bu isə obyekti daha tez «tapmağa» imkan verir. MS DOS-dan fərqli olaraq Windows-da fayllara uzun adlar vermək olur. Lakin fayl adında MS DOS-da olduğu kimi ən çoxu 8, ad genişlənməsində isə ən çoxu 3 simvol vermək daha əlverişli olur. Ad genişlənmələri kimi MS DOS-da olduğu standart işarələmələrdən (*exe, com, bat, bmp, doc* və s.) istifadə olunur.

Qovluq bildiyimiz kataloq, alt kataloq anlayışlarının ana-

loqu olub faylların, başqa qovluqların yerləşdirilməsi üçün nəzərdə tutulub. Hər bir qovluğa ad verilir və hər bir qovluq xüsusi nişanla işarələnir. Burada da faylın tam adı, ünvanı, fayla gedən yol anlayışları saxlanılır.

Sistemin qrafik örtüyü – istifadəçiyə sistemlə sadə, əlverişli ünsiyyət və idarəetmə imkanları verir. Bu örtük isttifadəçi interfeysi adlanır. İstifadəçi interrfeysinin imkanlarına baxaq. Windows sistemi EHM-ə yükləndikdə ekrana sistemin iş stolu adlanan pəncərə (şək. 5.1) verilir.



Şəkil 5.1

Bu pəncərə sisteminin əsas pəncərəsidir, burada sistemin əsas idarəetmə elementləri yerləşir. Onlara məsələlər lövhəsi, *«Пуск»* (baş menyu) düyməsi, kontekst menyusu, sistem nişanları külliyatı aiddir. Bundan əlavə istifadəçi iş stoluna qovluqlar, proqramlar, sənədlər və müxtəlif obyektlərin nişanları (yarlıklarını) yerləşdirə bilər. Məsələlər lövhəsi adətən, pəncərənin aşağı hissəsində yerləşdirilir. Bu lövhənin sol kənarında *«Пуск»* düyməsi və bir-birinin ardınca bir neçə düymə yerləşdirilə bilər. Sağ kənarında isə cari zaman və klaviaturadakı şrift seçimi (rus, ingilis, azərbaycan) indikatorları yerləşir. Qeyd edək ki, tələb olunduqda burada digər indikatorlar da yerləşdirilə bilər. *«Пуск»* düyməsi ekrana baş menyunu çağırmaqla bütün fayllara, sistemin imkanlarına, onun tənzimlənməsi vasitələrinə, EHM-in işinin sona çatdırılmasına keçidi təmin edir. Aktiv sənəd və proqram-

ların düymələri *«Пуск»* düyməsindən sağda yerləşir və həmin obyektlərə keçidi (bu düymələri sıxmaqla) təmin edir.

Sistem nişanlarına aşağıdakılar aiddir:

1) **Mənim kompüterim (Мой компьютер).** Bu nişan EHM-in ixtiyari fayl və qovluqlarına (bərk və elastik disklər, CD-ROM, printer və s.), sistemi idarəetmə və tənzimləmə vasitələrinə və s. keçidi təmin edən qovluq açır.

2) **Zibil qutusu (Корзина).** Bura ləğv olunan fayl və qovluqlar köçürülür. Səhvən ləğv olunan obyektləri bərpa edərək bu qutudan çıxarmaq olur.

3) **Şəbəkə göstəricisi (Сетевое окружение).** Bu nişan kompüter şəbəkəsinə qoşulduğu halda EHM-in bu şəbəkədəki yerini, şəbəkə serverini, qoşulma vəziyyətini və s. göstərir.

4) **Mənim sənədlərim (Мои документы).** Bura istifadəçinin hazırladığı sənədlər, ən çox istifadə etdiyi digər sənəd və proqramlar da yerləşdirilə bilir.

5) **İnternet Explorer.** Bu nişan İnternet qlobal kompüter şəbəkəsinə keçməyə imkan verən tətbiqi proqrama girişi təmin edir.

Bundan əlavə istifadəçinin istəyi ilə iş stoluna onu maraqlandıran obyektlərin – qovluq, sənəd, proqram və s. nişanları da çıxarıla bilər. Lakin sistemin baş qovluğunu, onun işi üçün əhəmiyyətli olmayan obyektlərlə doldurmaq rasional addım deyil. Bunun əvəzinə iş stolunda bu obyektlərin yarlıklarından isifadə etmək daha əlverişlidir. Yarlık – hər hansı bir obyekti (fayl, qovluq, proqram və s.) ifadə edən nişandır, lakin hər bir obyekt üçün istənilən sayda yarlık yaratmaq mümkündür. Yarlık obyektin təkrarını almağa, onu bir yerdən başqa yerə köçürməyə imkan vermir, ondan yalnız proqramı işə salmaq, sənəd və ya qovluğu açmaq üçün istifadə etmək olur. Ümumiyyətlə, işarə etdiyi obyektin yarlıkı *.lnk* tipli kiçik (374 *bayt*) bir fayldır və burada həmin obyektin parametrləri və yerləşdiyi yer haqqında məlumatlar saxlanılır. Yarlıka müraciət etdikdə sistem bu yarlıkda saxlanılan məlumatlardan istifadə edərək obyekti tapıb açır və ya iş vəziyyətinə gətirir. Nişan, yarlık və s. üçün kontekst menyusunu çağırmaq olur. Bunun üçün siçan göstəricisini obyekt üzərinə yerləşdirib, siçanın sağ düyməsini sıxmaq, menyu bölmələrini obyekt üçün seçmək üçün isə sol düyməsini sıxmaq lazımdır. Bu menyu-

da *açmaq (открыть), göndərmək (отправить), kəsmək (вырезать), təkrarını almaq (копировать), yarlık yaratmaq (создать ярлык), ləğv etmək (удалить), adını dəyişdirmək (переименовать)* və *xassələr (свойства)* bölmələri var. Burada *açmaq*la obyekti açmaq, *göndərmək*lə onu diskə, yaddaşın digər hissəsinə və s. göndərmək olur, *kəsmək* obyekti iş stolundan kəsib götürərək, mübadilə buferində saxlayır, *təkrarını almaq* ilə obyektin təkrarını alıb mübadilə buferində saxlamaq olur, *yarlık yaratmaq* ilə obyekt üçün daha bir yarlık yaradılır və iş stolunda yerləşdirilir, *ləğv etmək* isə obyekti iş stolundan zibil qutusuna göndərir, *adını dəyişdirmək*lə obyektə başqa ad vermək olur, nəhayət, *xassələr* ilə obyekt haqqında tam informasiya – onun tipini, tutduğu yaddaş həcmini, diskdəki vəziyyətini öyrənmək olur.

Baş menyu *Mənim kompüterim* nişanı kimi Windows sisteminin bütün proqramlarına və tənzimləmə vasitələrinə keçməyə imkan verir. Baş menyu iş stoluna *«Пуск»* düyməsi ilə çağırılır. Əvvəlcə ekrana menyunun birinci (ali) səviyyəsi çağırılır.

Baş menyunun əsas bölmələrinin təyinatına baxaq:

1) ***Proqramlar (Программы).*** Bu bölmə kompüterdə işə salınması mümkün olan proqram və ya proqramlar qovluğunun siyahısını verir. Bu siyahıya daxil olan bəzi proqramları qeyd edək: Standart proqramlar qovluğu WordPad, Paint, Bloknot, kalkulyator kimi alət proqramları birləşdirir. MS DOS seansı MS DOS-un işini təmin edən proqram işə salır. Bələdçi fayl sistemi ilə işi təmin edən və onun bütün obyektlərinə keçidi təmin edən proqramdır. Buraya eləcə də MS Word, MS Excel, MS Power Point, MS Access və s. proqramlar da daxil edilir.

2) ***Sənədlər (Документы).*** Bu bölmə sistemdə işlənmiş axırıncı 15 sənəd fayllarının siyahısını ekrana çıxarır. Bu siyahıdakı ixtiyari sənədi təkrarən açmaq üçün onun adını siçanın sol düyməsi ilə sıxmaq kifayətdir.

3) ***Tənzimləmə (Настройка).*** Bu bölmə ayrı-ayrı qurğularda təmzimləmə işlərini aparmağa imkan verən sistem elementlərinin siyahısını çıxarır.

4) ***Axtarış (Поиск).*** Bu bölmə fayl və qovluqların axtarışını təmin edir.

5) ***Arayış (Справка).*** Bu bölmə Windows-un arayışlar siste-

mi ilə işləməyə imkan yaradır.

6) *Yerinə yetirmək (Выполнить).* Bu bölmə ilə adına və ya ona gedən yola əsasən proqramı iş vəziyyətinə gətirmək və ya qovluq açmaq olur.

7) *İşi sona çatdırmaq (Завершение работы).* Bu bölmə ilə kompüterdə işi sona çatdırmaq və ya sistemi yenidən yükləmək olur.

Qeyd etdiyimiz kimi Windows-un tərkibinə çoxlu sayda müxtəlif təyinatlı proqramlar daxildir. Hər bir belə proqramın işə salınması zamanı o, öz pəncərəsində ekrana gətirilir. Bu proqramlar da öz növbəsində sənəd fayllarının açılması üçün pəncərələr yaradır. Ümumiyyətlə, sistemdə proqram, sənəd və qovluq, elcəcə də dialoq və arayış pəncərələri mövcuddur. Qovluq pəncərəsinə bu qovluğun tərkibi çıxarılır. Əvvəlcə proqram və ona yaxın olan qovluq pəncərəsinə baxaq. Hər bir belə pəncərənin, buradakı vəziyəti idarə edən əmr düymələri, alətlər lövhəsi, menyu və s. kimi elementləri var. Qeyd edək ki, Windows sisteminin hər bir pəncərəsində bəzi istisnalar olmaqla, demək olar ki, eyni elementlər ardıcıllığı olur.

Pəncərələrdə yerləşən əsas idarəetmə elementlərini qeyd edək:

1) *Əmr düyməsi.* Düymə düzbucaqlı formada olub, düymənin təyinatını göstərən simvol (məsələn, «×») və ya yazı (məsələn, *OK*, *İmtina* və s.) ilə müşayət olunur. Bunlardan birincilər pəncərə və dialoq lövhələrinin vəziyyətinin idarə edilməsi üçün, ikincilər isə dialoq lövhələrindəki dialoqun təşkil edilməsi üçün istifadə olunur. Düyməyə uyğun əməliyyatı yerinə yetirmək üçün onu siçanın sol düyməsi ilə sıxmaq lazımdır. Burada *OK* düyməsi əməliyyat və ya əmrin yerinə yetirilməsinin zəruri olduğunu təsdiq edir. *İmtina (Отмена)* düyməsi isə əvvəlki əməli ləğv edir.

2) *Pəncərə menyusu* – proqram və qovluq pəncərələrində bir sətirdə üfüqi yerləşmiş bölmələr ardıcıllığından ibarətdir. Menyu bölmələrinin sayı Windows-un müxtəlif pəncərələrində dəyişir, belə ki, qovluq pəncərəsində 4, MS Office proqramlarında 9 bölüm var. Bu bölmələrin ad və təyinatı da müxtəlif olur, lakin *Fayl, Düzəliş, Görünüş* və *?* bölmələri pəncərələrin çoxunda mövcuddur. Menyu bölmələrini açmaq üçün onları siçanın sol düyməsi ilə sıxmaq lazımdır. Nəticədə ekrana onun alt bölmələrinin ardıcıllığı

gətirilir. Analoji qaydada bu ardıcıllıqdan lazımi alt bölmə seçilir, bu isə hər hansı bir əmr və ya əməliyyatın yerinə yetirilməsinə səbəb olur.

3) *Alətlər lövhəsi.* Bu lövhə düymələr ardıcıllığından ibarətdir. Hər bir düymə pəncərəyə yerləşdirilmiş sənədlə müəyyən əməliyyatın yerinə yetirilməsini təmin edir. Əməliyyatın növü düymədə təsvir olunmuş nişanla müəyən edilr. Bu düymənin təyinatı haqqında qısa arayış almaq üçün siçan göstəricisini düymə üzərinə yerləşdirmək kifayətdir.

4) *Dialoq lövhəsi.* Bu lövhə istifadəçi ilə dialoq qurmaq üçün nəzərdə tutulub, burada sual verilir, istifadəçi isə həmin suala verilə billəcək cavabı seçir. Bundan əlavə o, xəbərdarlıq edə bilər, müəyyən informasiya çatdıra bilər, köməkçi məsləhət verə bilər və s.

İndii isə pəncərə elementlərinin yerləşdirilməsinə baxaq:

1) *Sistem menyusunun düyməsi.* Bu düymə pəncərəyə çıxarılmış obyektin nişanı ilə ifadə olunur və pəncərənin birinci sətrinin sol hissəsində yerləşir. O, pəncərənin işini idarə edən sistem menyusunu çağırır.

2) *Pəncərənin başlığı.* Başlıq pəncərənin birinci sətrinin mərkəzində yerləşir və pəncərənin adından ibarətdir. Ad kimi adətən, bu pəncərədə gətirilən faylın (qovluğun) adından istifadə olunur. Başlığın fonunun rəngi açıq və ya tünd ola bilər. Tünd rəngli fon pəncərənin aktiv, yəni cari anda işə hazır olduğunu bildirir. Bu cür pəncərəyə cari pəncərə deyilir.

a) *b)* *c)*

Şəkil 5.2

3) *a) – c) formalı düymələr* (şək. 5.2) pəncərənin birinci sətrinin sağ tərəfində yerləşir. Burada *a)* düyməsi pəncərəni ekrandan çıxarır, bu zaman sistem pəncərənin obyekti ilə işi davam etdirir və pəncərənin düyməsi iş stolunun məsələlər lövhəsində saxlanılır. *b)* düyməsi uyğun olaraq pəncərəni bütün ekran boyunca açır və əvvəlki ölçülərinə qaytarır. *c)* düyməsi pəncərəni ekrandan çıxarır və pəncərənin obyekti ilə işi sona çatdırır.

4) *Pəncərənin menyusu* qeyd etdiyimiz kimi pəncərənin ikinci sətrində yerləşir, bölmələrinin sayı və adları isə müxtəlif pəncərələrdə müxtəlif olur.

5) *Alətlər lövhəsi* pəncərə menyusundan aşağıda yerləşir. Bəzi pəncərələrdə bu cür bir neçə lövhə ola bilir. Həmin lövhədə uyğun pəncərədə iş üçün tələb olunan alətlər ardıcıllığı verilir.

6) *Cari vəziyyət sətiri* pəncərənin axırıncı sətirini tutur və alətlər lövhəsinin düymələrinin, menyu bölmələrinin və s. təyinatı haqqında məlumat verilməsi üçün istifadə olunur.

7) *İşçi sahə* – pəncərənin yerdə qalan hissəsini tutur və bura istifadəçini maraqlandıran obyekt çıxarılır.

8) *Pəncərənin çərçivəsi* – pəncərənin ölçülərinin və onun ekrandakı vəziyyətinin idarə edilməsində əhəmiyyətli elementdir.

9) *Çevirmə zolaqları* ölçüləri işçi sahənin ölçülərini aşan sənədləri nəzərdən keçirməyə imkan verir. Bu zolaqlar pəncərənin işçi sahəsinin sağ və aşağı tərəflərində yerləşir.

Pəncərələrlə aparıla bilən əməliyyatları qeyd edək:

1) Pəncərəni aktivləşdirmək, yəni cari anda onu iş vəziyyətinə gətirmək üçün pəncərənin ixtiyari yerini siçanın sol düyməsi ilə sıxmaq lazımdır, nəticədə başlığın fonunun rəngi tündləşir, pəncərə isə tam formada əks olunur. Bu əməliyyatdan ekranda bir neçə pəncərə olduqda və bir pəncərədə işi qurtarıb onu ekrandan çıxarmamaq şərtilə digər pəncərədə işə keçmək tələb olunduqda istifadə edilir.

2) Pəncərənin ölçülərini maksimum artırmaq (əvvəlki ölçülərini qaytarmaq) üçün *b)* düyməsini sıxmaq (*b)* düyməsini təkrar sıxmaq) (şək.5.2) lazımdır.

3) Pəncərənin ölçülərini dəyişdirmək üçün siçan göstəricisini pəncərənin çərçivəsi üzərinə yerləşdirib siçanın sol düyməsini sıxıb saxlamaq şərtilə siçanı pəncərə lazımi ölçüləri alana qədər hərəkət etdirmək lazımdır.

4) Pəncərənin ekrandakı vəziyyətini dəyişdirmək üçün siçan göstəricisini pəncərə başlığı üzərinə yerləşdirib siçanın sol düyməsini sıxıb saxlayaraq pəncərəni ekranın tələb olunan hissəsinə qədər çəkib aparmaq lazımdır.

5) Pəncərəni oradakı obyektlə işi sona çatdırmadan ekrandan kənarlaşdırmaq üçün siçanla *a)* düyməsini (şək. 5.2) sıxmaq lazımdır.

6) Pəncərəni oradakı obyektlə işi sona çatdıraraq ekrandan çıxarmaq üçün siçanla *c)* düyməsini (şək. 5.2) sıxmaq (və ya *Alt+F4* kombinasiyasından istifadə etmək) tələb olunur.

İndi isə dialoq pəncərələrinə baxaq. Bu pəncərələrdən istifadəçiyə hər hansı bir informasiyanı çatdırmaq, hər hansı bir sorğuya cavab almaq, hər hansı bir obyekti, məsələn, faylı seçmək üçün istifadə olunur. Birinci halda istifadəçi ona çatdırılan informasiyanı nəzərə alaraq *OK* düyməsini sıxmalı, ikinci halda sorğunu cavablandırmaq üçün uyğun düyməni sıxmalıdır. Üçüncü halda istifadəçi dialoq pəncərəsindəki daxil etmə sahəsində klaviaturadan obyektin adını (məsələn, fayl adı), hər hansı parametrin qiymətini və s. yığmalıdır. Qeyd edək ki, pəncərədə bu cür bir neçə daxil etmə sahələri ola bilər. Bu zaman bir sahədən digərinə keçmək üçün *Tab* düyməsindən istifadə etmək olar. Burada lazım olan əməliyyatların yerinə yetirilməsinin təsdiqi üçün *OK* və ya *Enter* düymələrini, imtina üçün isə *İmtina* və ya *Esc* düymələrini sıxmaq lazımdır.

Nəhayət, arayış pəncərələrinə baxaq. Əməliyyat sistemində bütün arayışlar müxtəlif ölçülü səhifələrə bölünüb. Hər bir səhifədə onun başlığında göstərilmiş bir mövzu tam şəkildə çatdırılır. Səhifələr öz növbəsində Windows sisteminin standart pəncərələrinə (menyunun olmamasını nəzərə almasaq) çıxarılır. Bu pəncərənin də ölçülərini dəyişdirmək, ekran boyunca sürüşdürmək olur. Pəncərədə *«Mündəricat» (Содержание), «Geriyə» (Назад)* və *«Parametrlər» (Параметры)* düymələri olur.

«Parametrlər» düyməsi ilə ekrana səhifədəki mətnlə müxtəlif əməliyyatlar aparmağa imkan verən menyu çıxarılır. Bəzi səhifələrdə adətən, hər hansı bir abzasın əvvəli və ya sonunda, ya da səhifənin sonunda düymələr olur. Bu düymələri siçanın sol düyməsi ilə sıxmaqla bu abzasların izahını verən və ya səhifəyə əlavə olan yeni səhifəyə keçmək olur. *«Geriyə»* düyməsi ilə isə əvvəlki səhifəyə qayıtmaq olur. *«Mündəricat»* düyməsi isə sistemləşdirilmiş şəkildə bütün arayışları çıxarılır. Çıxan siyahıdan arayış alınacaq bölməni axtarıb taparaq, açıb bu arayışa yiyələnmək olur.

Windows sistemində bəzi iş qaydalarına baxaq. Əvvəlcə iş stolunda qovluq və fayllarla iş qaydalarına baxaq:

1) Qovluğun (faylın) nişan və ya yarlıkını qeyd etmək üçün onu siçanın sol düyməsi ilə sıxmaq lazımdır.

2) Qovluğu (faylı) açmaq üçün onun nişanını siçanın sol düyməsi ilə ikiqat sıxmaq lazımdır. Nəticədə qovluğun (faylın)

pəncərəsi ekrana çıxarılır.

3) Hər hansı X qovluğunun (faylının) nişanını çıxarmaq üçün *Mənim kompüterim* qovluğunu açıb, burada X qovluğuna (faylına) rast gəlinənədək onun daxil olduğu diskin qovluğunu və digər qovluqları açmaq lazımdır.

4) X qovluğunu bağlamaq üçün *Backspase* düyməsini sıxmaq və ya pəncərənin alətlər lövhəsindəki başlığı əks çevrilmiş ox işarəsini sıxmaqla X qovluğunun yerləşdiyi daha yüksək səviyyəli qovluğa keçmək lazımdır.

5) X qovluğunun elementlərinin təsvir görünüşünü dəyişmək üçün X qovluğunu açıb, buradakı *Görünüş* menyu bölməsinə daxil olaraq, onun *Siyahı, Cədvəl* və s. alt bölmələrindən hansısa birini seçirik.

6) X qovluğundakı proqramı işə salmaq və ya sənəd faylını açmaq üçün X qovluğunu açdıqdan sonra obyektin nişanı və ya yarlıkını siçanın sol düyməsi ilə ikiqat sıxmaq lazımdır.

7) X qovluğunun (faylının) təkrarını digər Y qovluğuna yerləşdirmək üçün X obyektinin nişanını (yarlıkını deyil!) qeyd edib *Düzəlişlər* menyu bölməsinin *Təkrarını almaq* alt bölməsini seçmək, sonra isə Y qovluğunu açaraq *Düzəlişlər* menyu bölməsinin *Daxil etmək* alt bölməsini seçmək (yəni siiçanın sol düyməsi ilə sıxmaq) lazımdır.

8) X qovluğunu (faylını, yarlıkını) Y qovluğuna göndərmək üçün 7-ci bənddə verdiyimiz əməliyyatlarda *Düzəlişlər* menyu bölməsində *Təkrarını almaq* əvəzinə *Kəsmək* alt bölməsini seçərək, onları təkrar etmək kifayətdir.

9) X qovluğunu (faylını, yarlıkını) ləğv etmək üçün X obyektinin nişanını qeyd edib, *Del* düyməsini və ya alətlər löhəsindəki ləğv etmək düyməsini sıxmaq lazımdır.

10) X diskinin Y qovluğunda qovluq yaratmaq üçün *Mənim kompüterim* qovluğunda X diskini və Y qovluğunu açıb, *Fayl* menyu bölməsinin *Yaratmaq* alt bölməsində *Qovluq* sətrinə keçmək lazımdır. Sonra isə pəncərənin axırıncı sətrində yaranan daxiletmə sahəsində qovluğun adını daxil edib, *Enter* düyməsini sıxmaq lazımdır. Fayl yaratmaq üçün isə *Fayl* menyu bölməsinin *Yaratmaq* alt bölmməsində faylın yaradılacağı uyğun proqram adını seçib sıxmaq lazımdır. Nəticədə pəncərənin son sətrindəki sahədə faylın şərti adı və ad genişlənməsi verilir. Burada şərti adı

dəyişdirib, ad genişlənməsini isə saxlayıb *Enter* düyməsini sıxmaq lazımdır.

11) X qovluğunun (faylının) yarlıkını yaratmaq üçün əvvəlcə X qovluğunun (faylının) nişanını qeyd edib, sonra pəncərənin *Fayl* bölməsinin *Yarlık yaratmaq* alt bölməsinə keçmək lazımdır.

İndi isə *Bələdçi* proqramı ilə işə baxaq. Bu proqramda fayllarla ixtiyari əməliyyatlar yerinə yetirmək olar. Proqramı sistemin Baş menyusunun *Proqramlar* bölməsindən tapıb açmaq olur. Nəticədə ekrana Windows sisteminin həmin adlı standart pəncərəsi çıxarılır. Onun işçi sahəsində fayl sisteminin strukturunu görmək olar. Pəncərənin işçi sahəsi şaquli istiqamətdə iki hissəyə bölünüb. Sol tərəfdə qrafik şəkildə EHM-dəki qovluqlar ağacı təsvir olunub ki, bura *İş stolu, Mənim kompüterim, Disk qovluqları və s.* daxildir. Hər bir qovluğun öz nişanı var. Daxilində digər qovluqlar olan qovluqların nişanlarının sol tərəfində «+» nişanı qoyulur. Əgər bu işarəni siçanın sol düyməsi ilə sıxsaq, onun daxilindəki qovluqların siyahısı çıxarılır və «+» işarəsi «-» ilə əvəz olunur. Əgər indi «-» işarəsini sıxsaq, siyahı ləğv olunacaq və əvvəlki vəziyyət bərpa olunacaq. Qovluğun nişanını siçanın sol düyməsi ilə sıxdıqda pəncərənin sağ tərəfində qovluğun tərkibi göstəriləcəkdir. Bələdçi proqramının axırıncı sətrində cari vəziyyət sətri yerləşir. Burada cari qovluğun tutduğu yaddaş həcmi və cari diskdə qalan azad yaddaş həcmi haqqında məlumat verilir. Bələdçi proqramında aparıla bilən əsas əməliyyatları qeyd edək:

1) Qovluq və ya faylı cari etmək üçün onun nişanını siçanın sol düyməsi ilə sıxmaq lazımdır.

2) Qovluqlar (fayllar) qrupunu cari etmək üçün (məsələn, k-cıdan n-yə qədər) k-cı obyektin nişanını siçanın sol düyməsi ilə sıxıb, *Shift* düyməsini sıxıb saxlayaraq k-cı obyektin nişanını siçanın sol düyməsi ilə sıxmaq lazımdır.

3) Qovluğu açmaq üçün qovluq pəncərənin sol tərəfində olduqda onun nişanını siçanın sol düyməsi ilə bir dəfə, sağ tərəfində olduqda isə ikiqat sıxmaq lazımdır.

4) Qovluğu bağlamaq üçün yəni, onu yerləşdiyi daha ali səviyyəli qovluğa qaytarmaq üçün pəncərənin alətlər lövhəsindəki başlığı əks tərəfə çevrilmiş ox işarəsini siçanın sol düyməsi ilə ikiqat sıxmaq lazımdır. Qeyd edək ki, burada qovluq (fayl, yarlık)

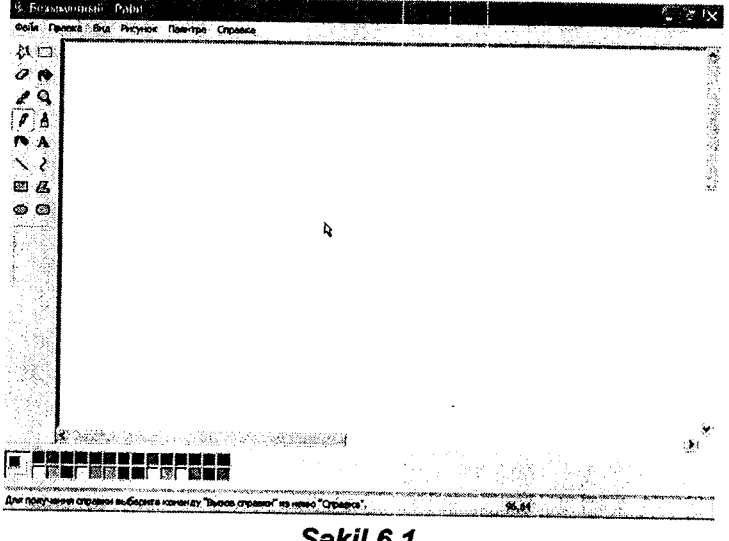
yaratmaq, təkrarını almaq, yerini dəyişdirmək və ləğv etmək yuxarıda təsvir etdiyimiz qaydalarla yerinə yetirilir.

5) Proqramı (exe və ya com tipli faylı) işə salmaq və ya sənəd faylını açmaq üçün obyekt nişanını qeyd edib, onu siçanın sol düyməsi ilə ikiqat sıxmaq lazımdır.

6) Fayl sistemində qovluğu (faylı) axtarmaq üçün pəncərənin *Servis* menyu bölməsinin *Tapmaq alt* bölməsində *Qovluq* və fayl sətrini seçib, ekrana verilən dialoq pəncərəsində «Fayl» sahəsinə faylın adını, «*Qovluq*» sahəsinə diskin adını (C:,D: və s.) yazıb, bu pəncərədəki «*Tapmaq*» düyməsini sıxmaq lazımdır.

2.6. MS Paint qrafik redaktoru

Windows əməliyyat sisteminin tərkibinə daxil olan MS Paint qrafik redaktoru çox da mürəkkəb olmayan təsvirlərin qurulması üçün nəzərdə tutulub. Proqram sistemin Baş menyusunun *Proqramlar* bölməsinin *Standart* alt bölməsinə keçməklə işə salınır. Bu zaman ekrana proqramın iş stolu adlanan pəncərə verilir (şək. 6.1).



Şəkil 6.1

Redaktorun iş stolunun birinci iki sətri başlıqdan, idarəetmə düymələrindən və menyudan ibarətdir. Pəncərənin yerdə qa-

lan hissəsi aşağıdakı beş oblasta bölünüb:

1) Alətlər lövhəsi (adətən pəncərənin sol tərəfində yerləşir) hər birində bir cüt düymə olan 8 sırada yerləşdirilmiş 16 düymədən ibarətdir. Sadəlik üçün alətləri, məsələn 1.1 aləti, yəni birinci sətirdəki birinci alət kimi işarə edək. Burada 1.1 və 1.2 düymələri uyğyn olaraq rəsmin ixtiyari sahəsini və düzbucaqlı sahəsini qeyd edir. 2.1 düyməsi ilə rəsmin seçdiyimiz hissəsini silə bilirik. 2.2 düyməsi rəsmin seçdiyimiz qapalı bir hissəsini verdiyimiz rənglə rəngləyə bilir. 3.1 düyməsi rəng seçimini təmin edir. Bu aləti seçdikdən sonra kursoru təsvirdəki ixtiyari rəng üzərinə qoyub siçanın sol düyməsini sıxmaqla həmin rəngi seçmiş oluruq və ondan rəsmin ixtiyari hissəsində istifadə edə bilərik. 3.2 düyməsi ilə təsvirin görünüş miqyasını dəyişdirmək olur. 4.1 aləti xətlər çəkilməsi üçün nəzərdə tutulub. Bunun üçün siçanın sol düyməsini sıxıb saxlayaraq kursoru lazım olan istiqamətdə siçan vasitəsilə hərəkət etdirmək lazımdır. 4.2 aləti isə 4.1 alətinə nisbətən daha qalın xətlərin çəkilməsini təmin edir. 5.1 düyməsi ilə kursorun cari anda durduğu nöqtəyə rəng çiləmək olur. 5.2 düyməsi ilə təsvirə mətn əlavə etmək olur. 6.1 düyməsi ilə rəsmə düz xətt parçası, 6.2 düyməsi ilə isə əyri xətt daxil etmək olur. 7.1 düyməsi düzbucaqlı (kvadrat), 7.2 düyməsi – çoxbucaqlı fiqur, 8.1 düyməsi ellips (çevrə), 8.2 düyməsi isə oval künclü düzbucaqlı çəkməyə imkan verir.

2) Əlavə xarkteristikalar sahəsi. Bu sahə alətlər lövhəsindən aşağıda verilir və məsələn alətlə təsvir olunan xəttin qalınlığını, obyektin tipini (çevrə, rənglənmiş çevrə, konturlu və ya konturusuz çevrə və s.) təyin etməyə imkan verir. Bunun üçün kursoru tələb olunan formalı obyektin təsviri üzərinə yerləşdirib siçanın sol düyməsini sıxmaq lazımdır.

3) Palitra pəncərənin aşağı hissəsində yerləşir və müxtəlif rəngli 28 düymədən ibarətdir. Palitra pəncərənin fonunun rəngini, təsvir olunan elementlərinin rəngini təyin etməyə imkan verir.

4) Rəng pəncərəsi palitranın sol tərəfində yerləşir və iki düzbucaqlıdan ibarətdir. Aşağıdakı düzbucaqlının rəngi fonun rənginə, yuxarıdakının rəngi isə təsvir olunacaq elementlərin (xətt, nöqtə və s.) rənginə uyğun gəlir.

5) Cari vəziyyət sətri – pəncərənin son sətridir. Burada alətlər lövhəsinin düymələrinin təyinatı və siçan kursorunun koordi-

natları haqqında məlumatlar verilir.

6) Redaktorun iş stolounun yerdə qalan hissəsi isə bilavasitə təsvirin qurulması üçün nəzərdə tutulmuş işçi sahəsidir. Bu sahənin aşağı və sağ tərəflərində nəzərdən keçirmə zolaqları yerləşir. Təsvirin ölçüləri işçi sahənin ölçülərini üstələdikdə bu zolaqların köməyi ilə təsviri üfüqi və şaquli istiqamətlərdə hərəkət etdirərkən onu tam şəkildə nəzərdən keçirmək olur.

Redaktorun menyu sətri 6 bölmədən: *Fayl (Файл), Düzəlişlər (Правка), Görünüş (Вид), Şəkil (Рисунок), Palitra (Палитра)* və *Arayış (Справка)* bölmələrindən ibarətdir. Bu bölmələrə baxaq:

1) ***Fayl*** bölməsinin aşağıdakı alt bölmələri var:

a) *Yaratmaq (Создать)* əmri ilə yeni təsvir yaratmaq üçün rəsm sahəsi açmaq olur.

b) *Açmaq (Открыть)* əmri ilə bərk və ya elastik diskdə olan mövcud rəsm fayllarını proqram pəncərəsinə gətirmək olur. Bu zaman açılan dialoq pəncərəsində lazım olan faylı tapıb, qeyd edərək bu pəncərədəki *Açmaq* düyməsini sıxmaq lazımdır.

c) *Saxlamaq (Сохранить)* əmri yaradılmış təsviri və ya onun üzərində edilmiş dəyişiklikləri diskdə yadda saxlayır. Bu zaman açılan dialoq pəncərəsindəki *Faylın adı* yazı sahəsində təsvir faylının adını yığıb, buradakı *Saxlamaq* düyməsini sıxmaq lazımdır.

ç) *Necə saxlamaq (Сохранить как)* əmri ilə cari sənədi başqa qovluqda, diskdə başqa adla saxlamaq olur.

d) *İlkin baxış (Предварительный просмотр)* əmri cari səhifənin çapa göndərilməzdən əvvəl, onu tam şəkildə nəzərdən keçirməyə imkan verir.

e) *Səhifə maketi (Макет страницы)* əmri ilə açılan pəncərədə cari sənəddə səhifənin ölçülərini, formatını təyin etmək olur.

ə) *Çap (Печать)* əmri səhifəni çapa göndərir. Bu zaman açılan pəncərədə çap olunacaq səhifələrin nömrələrini, sayını və s. təyin etmək olur.

f) *Göndərmək (Отправить)* əmri ilə cari rəsmi elektron poçtla digər istifadəçiyə göndərmək olur.

g) *İş stolunu örtmək (Заполнить рабочий стол)* əmri ilə cari sənəddəki təsviri iş stolunun mərkəzində yerləşdirmək olar.

h) *Çıxış (Выход)* əmri cari proqram pəncərəsini bağlayır.

Bu zaman əgər cari sənəd yadda saxlanılmayıbsa, onda onu ekrana verilən dialoq pəncərəsindəki *Hə* düyməsini sıxmaqla yadda saxlamaq olar.

2) ***Düzəlişlər*** menyu bölməsinin aşağıdakı alt bölmələri var:

a) *Ləğv etmək (Отмена)* əmri axırıncı yerinə yetirlmiş əməliyyatın nəticəsini ləğv edir.

b) *Təkrar (Повтор)* əmri ilə axırıncı yerinə yetirilmiş əməliyyatı təkrar yerinə yetirmək olur.

c) *Kəsmək (Вырезать)* əmri ilə cari təsvirdə qeyd olunmuş hissəni, mübadilə buferində saxlamaq olar və yeni kəsmə əməliyyatı aparılana qədər bu hissə cari təsvirin ixtiyari yerində bərpa oluna bilər və ya Windows-un ixtiyari proqramında açılan sənədə daxil edilə bilər.

ç) *Təkrarını almaq (Копировать)* əmri təsvirin qeyd olunmuş hissəsinin təkrarını alıb, mübadilə buferində saxlayır.

d) *Daxil etmək (Вставить)* əmri buferdə saxlanılmış obyekti cari təsvirdə ixtiyari yerə daxil edə bilir.

e) *Hər şeyi qeyd etmək (Выделить все)* əmri təsviri tamamilə qeyd edir.

ə) *Qeyd edilənin silinməsi (Очистить выделения)* əmri qeyd olunmuş hissəni təmizləyir.

f) *Təkrarını fayla köçurmək (Копировать в файл)* əmri ilə cari təsviri tam və ya onun bir hissəsini qeyd edərək başqa fayla köçürmək olur. Bunun üçün uyğun dialoq pəncərəsində köçürülmə yerinə yetiriləcək faylın adını verib *OK* düyməsini sıxmaq lazımdır.

g) *Fayldan daxil etmək (Вставить из файла)* əmri digər fayldakı təsviri cari sənəddəki rəsmə əlavə edir.

3) ***Görünüş*** menyu bölməsinin aşağıdakı alt bölmələri var:

a) *Alətlər lövhəsi (Набор инструментов)* əmri pəncərədəki alətlər lövhəsini ekrandan çıxarır və ya yerinə qaytarır.

b) *Palitra (Палитра)* əmri pəncərdəki palitranı ekrandan çıxarır və ya yerinə qaytarır.

c) *Cari vəziyyət sətri (Строка состояния)* əmri pəncərədəki cari vəziyyət sətrini ekrandan çıxarır və ya yerinə qaytarır.

ç) *Miqyas (Масштаб)* əmri ilə rəsmin görünüş miqyasını dəyişdirmək olur.

d) *Rəsmə baxış (Просмотреть рисунок)* əmri ilə pəncə-

rənin bütün elementlərini ekrandan çıxararaq rəsmə tam baxmaq olur.

e) *Mətn atributları lövhəsi (Панель атрибутов текста)* əmri təsvirə mətn daxil edərkən şriftin, onun ölçüsünün, stilinin və s. seçilməsi üçün istifadə olunan lövhəni ekrana çıxarır və ya oradan ləğv edir.

4) ***Şəkil*** bölməsində aşağıdakı alt bölmələr var:

a) *Çevirmək və döndərmək (Отразить)* əmri ilə təsviri şaquli, üfüqi istiqamətlərdə çevirmək və 90, 180, 270 dərəcəli bucaqlar altında döndərmək olur.

b) *Uzatmaq və əymək (Растянуть и наклонить)* əmri təsviri üfüqi və şaquli istiqamətlərdə uzatmağa və əyməyə imkan verir.

c) *Rəngləri çevirmək (Обратить цвета)* əmri təsvirdə qeyd olunmuş hissənin rəngini ona əks olan rəngə (məsələn, ağ rəngi qara rəngə) dəyişdirir.

ç) *Atributlar (Атрибуты)* əmri təsvirin verildiyi iş sahəsinin ölçülərini, ölçü vahidlərini və s. parametrləri seçməyə imkan verir.

d) *Təsvirin silinməsi (Очистить)* əmri iş sahəsindəki təsviri tam şəkildə silir.

5) ***Palitra*** menyu bölməsinin aşağıdakı alt bölməsi var:

Palitranı dəyişdirmək (Изменить палитру) əmri proqram pəncərəsindəki palitrada verilmiş 28 rəngi proqramda olan daha 48 rəng çalarlarından hər hansı biri ilə dəyişdirməyə imkan verir.

6) ***Arayış*** menyu bölməsi ilə isə proqramda iş qaydaları haqqında müxtəlif arayışlar almaq olur.

İndi isə qrafik redaktorda bəzi iş qaydalarına baxaq:

1) İşə başlamazdan əvvəl ekranın təmizlənməsini *Şəkil* menyu bölməsinin *Təsvirin silinməsi* əmri ilə yerinə yetirmək olar.

2) Rənglərin təyin olunması. Bunun üçün kursoru palitranın lazım olan rənginin üzərinə qoyub, fonun rəngini vermək üçün sağ düyməsini, xətlərin, nöqtələrin rəngini vermək üçün isə sol düyməsini sıxmaq lazımdır.

3) İş sahəsini bütün ekran boyunca açmaq üçün *Görünüş* bölməsindəki *Rəsmə baxış* əmrini vermək, əvvəlki vəziyyətə qayıtmaq üçün isə pəncərənin ixtiyari nöqtəsini siçanın sol düyməsi ilə sıxmaq lazımdır.

4) Pəncərədəki təsvirin ölçülərini kiçiltmək üçün *Fayl* böl-

məsindəki *İlkin baxış* əmrindən, böyütmək üçün isə *Görünüş* bölməsinin *Miqyas* alt bölməsindəki *İri ölçü* və ya *Seçmək (ölçünü)* əmirlərindən istifadə edilir.

5) Düz xətt parçasını qurmaq üçün əvvəlcə 6.1 düyməsini sıxıb, əlavə xarakteristikalar pəncərəsində xəttin qalınlığını seçməli, sonra kursoru parçanın başlanğıc nöqtəsinə qoyub, siçanın sol düyməsini sıxıb saxlayaraq kursoru parçanın son nöqtəsinə doğru sürüşdürmək lazımdır. Qeyd edək ki, ciddi üfüqi və ya şaquli xətt çəkərkən *Shift* düyməsini sıxıb saxlamaq lazımdır.

6) Təsviri silmək üçün 2.2 düyməsini sıxıb, əlavə xarakteristikalar pəncərəsində pozanın qalınlığını seçib siçanın sol düyməsini sıxıb saxlayaraq silinən zolağın əvvəlindən sonunadək kursoru sürüşdürmək lazımdır. Qeyd edək ki, pozan təsviri fonun rəngi ilə silir (rəngləyir).

7) Düzbucaqlını (oval künclü düzbucaqlını) qurmaq üçün 7.1 (8.2) düyməsini sıxıb, əlavə xarakteristikalar lövhəsində düzbucaqlının tipini (çərçivəli, çərçivəsiz, rənglənmiş) seçirik, sonra kursoru çəkəcəyimiz düzbucaqlının yuxarı sol küncünə yerləşdirib, siçanın sol düyməsini sıxıb saxlayaraq, kursoru düzbucaqlının diaqonalı boyunca yuxarıdan aşağıya doğru sürüşdürmək lazımdır.

8) Çoxbucaqlını qurmaq üçün 7.2 düyməsini sıxıb, əlavə xarakteristikalar pəncərəsindən çoxbucaqlının tipini (çərçivəli, çərçivəsiz, rənglənmiş) seçirik, çoxbucaqlının rənglənəcəyi rəngi seçirik, sonra isə yuxarıda düz xətt parçasını qurmaq üçün verdiyimiz qaydadan istifadə edərək, çoxbucaqlının tərəflərini çəkirik. Bu zaman növbəti düz xəttin başlanğıc nöqtəsi əvvəlkinin son nöqtəsi olacaqdır.

9) Çevrə və ya ellipsin qurulması üçün 8.1 düyməsini sıxıb, əlavə xarakteristikalar pəncərəsində fiqurun tipi (rənglənmiş, konturlu və ya kontursuz fiqur) seçilir, fiquru rəngləyəcəyimiz rəngi seçirik, sonra isə siçanın sol düyməsini sıxıb saxlayaraq, kursoru sürüşdürməklə ellipsi çəkirik. Çevrə qurarkən isə *Shift* düyməsini sıxıb saxlamaq lazımdır.

10) Qapalı konturla məhdudlaşdırılmış oblastın rənglənməsi üçün 2.2 düyməsini sıxmaq, rəngi seçmək və kursoru kontur daxilindəki ixtiyari nöqtəyə yerləşdirib, siçanın sol düyməsini sıxmaq lazımdır.

11) Əyri xəttin qurulması üçün 6.2 düyməsini sıxmaq və düz xətt parçasının qurulması üçün yuxarıda verdiyimiz əməliyyatlar ardıcıllığını yerinə yetirmək lazımdır. Bundan sonra siçanın sol düyməsini sıxıb saxlayaraq, siçanı sürüşdürməklə, düz xətt parçasını da istənilən istiqamətlərdə aymək olur.

12) Karandaşın (fırçanın) köməyi ilə müxtəlif fiqurların qurulması üçün 4.1 (4.2) düyməsini sıxıb, əlavə xarakteristikalar pəncərəsində karandaş (fırça) üçün xəttin qalınlığını seçmək, sonra siçanın sol düyməsini sıxıb saxlayaraq, siçanı hərəkət etdirmək lazımdır.

13) 5.1 düyməsi ilə müxtəlif fiqurların qurulması üçün bu düyməni sıxıb, sonra 12-ci bənddəki əməliyyatları təkrar etmək lazımdır.

14) Ekrana mətn hissəsi daxil etmək üçün 5.2 düyməsini sıxıb, sonra kursoru mətnin veriləcəyi yerin əvvəlinə yerləşdirib, siçanın sol düyməsini sıxmaq lazımdır. Nəticədə ekrana mətnin çıxarılacağı oblastı işarə edən çərçivə verilir, bura klaviaturadan mətni yığmaq lazımdır.

15) Lupa rejimində işləmək üçün 3.2 düyməsini sıxmaq lazımdır. Bu zaman ekrana düzbucaqlı çərçivə verilir, siçanın sol düyməsini sıxıb saxlamaqla çərçivəni böyüdülməsi tələb olunan təsvir hissəsinə sürüşdürmək lazımdır. Sonra isə siçanın sol düyməsini sıxmaq lazımdır. Nəticədə həmin hissədəki təsvir böyüdülmüş olur.

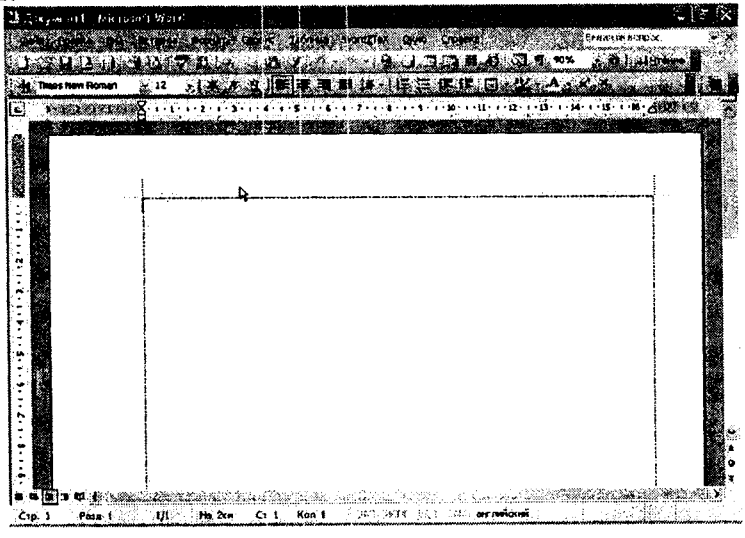
16) Təsvirin diskə, fayla yazılması və təsvirin fayldan ekrana çıxarılması əməliyyatları Windows sisteminin proqramları üçün yuxarıda verdiyimiz standart prosedurlara uyğundur.

2.7. MS Word mətn redaktoru

Microsoft Word 2000 mətn redaktoru mətnlərin, sənədlərin hazırlanması və emalı proqramıdır. MS Word yükləndikdə ekrana proqramın iş stolu adlanan pəncərə çıxarılır (şək. 7.1).

Burada birinci sətir başlıq – Windows sisteminin standart pəncərə başlıqları ilə eynidir. İkinci sətirdə proqramın menyu sətiri yerləşir. Menyu sətrindən aşağıda piktoqramlı düymələr ardıcıllığından ibarət alətlər lövhəsi yerləşir. Hər bir düyməyə hər hansı bir əmr uyğun gəlir, düymə üzərindəki piktoqram isə əmrin

mahiyyətini ifadə edir. Düymələrin çoxu menyuda olan ən çox istifadə olunan əmrləri təkrarlayırlar. Adətən, menyu sətri altında iki növ alətlər lövhəsi – *Standart* və *Formatlaşdırma* lövhələri yerləşir. Bu lövhələri, ixtiyari düyməni siçanın sağ düyməsini sıxmaqla ekrana gələn kontekst menyusu ilə idarə etmək əlverişli olur.



Şəkil 7.1

Proqramın iş stolunun yuxarı və sol tərəfində yerləşən üfüqi və şaquli koordinat xətkeşlərinin köməyi ilə səhifənin kənarlarından buraxılan sahələri, abzas ölçülərini təyin etmək, sütunların enini dəyişdirmək və s. yerinə yetirmək olur. Proqram pəncərəsinin son sətri cari vəziyyət sətri müxtəlif məlumat və arayışların gətirilməsi üçün nəzərdə tutulub.

MS Word redaktoru sənədi müxtəlif rejimlərdə nəzərdən keçirməyə imkan verir:

1) *Adi*-əməliyyatların bir çoxunun yerinə yetirilməsi üçün ən əlverişli rejimdir.

2) *Web-səhifə* – sənədi Web-səhifə formasında ifadə edir;

3) *Səhifə ayırıcları* – sənədi çapa çıxarılacaq formada ifadə edir;

4) *Struktur* – sənədin strukturu ilə işi təmin edir, mətn və başlıqları göstərməyə və ya gizlətməyə imkan verir, alt sənədlərin

yaradılmasını və onlarla işi təmin edir.

Rejimlər arasında keçid *Görünüş* menyu bölməsinin uyğun alt bölmələrinin və ya üfüqi çevirmə zolağından solda yerləşən düymələrin köməyi ilə yerinə yetirilir. Pəncərələrin sağ və aşağı hissələrində yerləşən şaquli və üfüqi çevirmə zolaqları mətnin redaktorun pəncərəsində şaquli və üfüqi istiqamətlərdə hərəkət etdirilməsini təmin edirlər.

Redaktorun menyu bölmələrinə baxaq. Menyu sətrində 9 bölmə var: *Fayl (Файл), Düzəlişlər (Правка), Görünüş (Вид), Daxil etmək (Вставка), Format (Формат), Servis (Сервис), Cədvəl (Таблица), Pəncərə (Окно)* və *Arayış (Справка).*

1) ***Fayl*** menyu bölməsinin aşağıdakı alt bölmələri var:

a) *Yaratmaq (Создать)* əmri *Ümumi, Məktublar* və *Fakslar, Qeydlər, Hesabatlar, Nəşrlər, Digər sənədlər* və *Web-səhifələr* şablonları əsasında yeni sənəd yaradır.

b) *Açmaq (Открыть)* əmri mövcud sənədi yaddaşdan proqram pəncərəsinə çağırır.

c) *Bağlamaq (Закрыть)* əmri cari sənəd olan proqram pəncərəsini bağlayır. Bu zaman sənəddə dəyişikliklər olmuşdursa, onda bu dəyişikliklərin yadda saxlanılmasını təklif edən dialoq pəncərəsi açılır.

ç) *Saxlamaq (Сохранить)* əmri cari sənədi yaddaşda saxlayır. Bu zaman açılan dialoq pəncərəsində faylın adını verib *Saxlamaq* düyməsini sıxmaq lazımdır.

d) *Necə saxlamaq (Сохранить как)* əmri mövcud sənədi başqa adla, başqa yerdə saxlanılmasını təmin edir.

e) *Web-səhifə kimi saxlamaq (Сохранить как Web-страницу)* əmri ilə cari sənəd Web-səhifə kimi HTML formatında yaddaşda saxlanılır.

ə) *Səhifə parametri (Параметры страницы)* əmri ilə açılan eyni adlı dialoq pəncərəsində sənədin səhifələrinin lazım olan parametrlərini vermək olur.

f) *İlkin baxış (Предварительный просмотр)* əmri cari sənədin çap edilməzdən əvvəl səhifələrinə baxışı təmin edir.

g) *Web-səhifə kimi ilkin baxış (Предварительный просмотр Web-страницы)* əmri ilə cari sənədə Web-səhifə kimi ilkin baxışı təmin etmək olur.

ğ) *Çap (Печать)* əmri ilə cari sənədi çapa göndərmək

olur. Bu zaman ekrana çıxarılan eyni adlı dialoq pəncərəsində çapın parametrlərini müəyyən etmək olur.

h) *Göndərmək (Отправить)* əmri cari sənədi elektron məktub və ya faks məlumat kimi göndərməyə imkan verir.

x) *Xassələr (Свойства)* əmri cari sənəd haqqında əlavə məlumatlar almağa imkan verir.

2) ***Düzəlişlər*** bölməsinin aşağıdakı alt bölmələri var:

a) *İmtina (Отменить)* əmri yerinə yetirilmiş axırıncı əməliyyatı ləğv edir.

b) *Təkrar etmək (Повторить)* əmri axırıncı yerinə yetirilmiş əməliyyatı təkrar edir.

c) *Kəsmək (Вырезать)* əmri cari sənəddə qeyd olunmuş mətn hissəsini kəsib mübadilə buferində saxlayır.

ç) *Təkrarını almaq (Копировать)* əmri cari sənəddə qeyd olunmuş mətn hissəsinin təkrarını alıb mübadilə buferində saxlayır.

d) *Daxil etmək (Вставить)* əmri bufer yaddaşdakı mətn hissəsini cari sənəddə ixtiyari yerə daxil edir.

e) *Xüsusi daxiletmə (Специальная вставка)* əmri bufer yaddaşda olan və Windows-un digər proqramlarında yaradılmış obyektləri cari sənədə xüsusi format əsasında daxil etməyə imkan verir. Bu zaman açılan dialoq pəncərəsindən həmin format seçilə bilər.

ə) *Hiperistinad kimi daxil etmək (Вставить как гиперссылку)* əmri digər sənəddən mübadilə buferinə köçürülmüş mətn hissəsini hiperistinad şəklində cari sənədə daxil etməyə imkan verir.

f) *Silmək (Очистить)* əmri cari sənəddə qeyd olunmuş hissəni və ya obyekti ləğv etməyə imkan verir.

g) *Hamısını qeyd etmək (Выделить все)* əmri cari sənəddəki mətni tamamilə qeyd edir.

ğ) *Tapmaq (Найти)* əmri ilə açılan dialoq pəncərəsində *Tapmaq* yazı sahəsində söz və ya söz birləşmələrinin nümunəsini verməklə onların cari sənəddəki mətndən tapılması təmin edilir.

h) *Əvəz etmək (Заменить)* əmri açılan dialoq pəncərəsində *Tapmaq* yazı sahəsində verilən söz və ya söz birləşmələrini cari sənəddəki mətndə axtarıb taparaq dialoq pəncərəsinin *Əvəz etmək* sahəsində verdiyimiz söz və ya söz birləşmələrinin nümu-

nələri ilə əvəz edilməsini təmin edir.

x) *Keçmək (Перейти)* əmri ilə cari sənədin konkret səhifəsinə, sətrinə, obyektinə və s. keçmək olur. Bunun üçün açılan dialoq pəncərəsində keçiləcək obyektin koordinatlarını vermək lazımdır.

ı) *Əlaqələr (Связи)* əmri seçilmiş obyekt və onun yaradıldığı proqram arasındakı əlaqəni göstərir. Əmrin yerinə yetirilməsi nəticəsində açılan pəncərədə bu əlaqəni qurmaq, yeniləşdirmək, obyekti başqası ilə əvəz etmək olur.

i) *Obyekt (Объект)* əmri cari sənədə daxil edilmiş və digər proqramlarda hazırlanmış hər hansı bir obyekti redaktə etmək və ya yenisi ilə əvəz etməyə imkan verir.

3) ***Görünüş*** menyu bölməsinin aşağıdakı alt bölmələri var:

a) *Adi (Обычный)* əmri cari sənədin adi formada görünüşünü təmin edir.

b) *Web-sənəd (Web-документ)* əmri cari sənədin Web-sənəd görünüşünü təmin edir.

c) *Səhifə ayırıcları (Разметка страницы)* əmri cari sənədə çapa çıxarılacaq formaya uyğun görünüş verir.

ç) *Alətlər lövhəsi (Панели инструментов)* əmri ilə proqram pəncərəsinə lazım olan alətlərin verilməsini və lazım olmayanların çıxarılmasını yerinə yetirmək olar.

d) *Struktur (Структура)* əmri ilə struktur iş rejiminə keçmək olar.

e) *Xətkeş (Линейка)* əmri ilə ekrana şaquli və üfüqi xətkeşləri vermək və ya çıxarmaq olur.

ə) *Sənəd sxemi (Схема документа)* əmri ilə sənəd pəncərəsi üfüqi istiqamətdə iki hissəyə bölünür. Sol tərəfdə sənədin strukturu, sağda isə sənədin özü verilir.

f) *Kolontitullar (Колонтитулы)* əmri səhifədə yuxarı və aşağı kolontitulların yaradılması və redaktə edilməsini təmin edir.

g) *İstinadlar (Сноски)* əmri ilə cari sənəddəki nömrəli istinadlar redaktə edilə bilir.

ğ) *Qeydlər (Примечания)* əmri ilə cari sənəddə olan qeydlər redaktə edilir.

h) *Bütün ekran boyu (Во весь екран)* əmri cari sənədin bütün ekran boyu əks olunmasını təmin edir.

x) *Miqyas (Масштаб)* əmri cari sənədin görünüş miq-

yaslarını dəyişdirməyə imkan verir.

4) ***Daxil etmək***-menyusuna aşağıdakı alt bölmələr aiddir:

a) *Bölünmə (Разрыв)* əmri ilə səhifədəki mətn bölünür və kursorun durduğu mövqedən aşağıdakı mətn hissəsi yeni səhifəyə keçirilir.

b) *Səhifələrin nömrələnməsi (Номера страниц)* əmri cari sənədin səhifələrinin, bu zaman açılan dialoq pəncərəsində verəcəyimiz parametrlərə uyğun nömrələnməsini təmin edir.

c) *Tarix və zaman (Дата и время)* əmri cari sənədə, bu zaman açılan dialoq pəncərəsində seçdiyimiz formalı cari tarix və zaman daxil edir.

ç) *Avtomətn (Автотекст)* əmri cari sənədə tarix, zaman, ad, soyad, müxtəlif termin və mətn hissələri avtomatik daxil etməyə imkan verir.

d) *Sahə (Поле)* əmri cari sənədə düstur, simvol, mündəricat və paraqraf nömrələrini daxil edir və s. kimi əməliyyatlar yerinə yetirir.

e) *Simvol (Символ)* əmri cari sənədə klaviaturada nəzərdə tutulmamış simvolların daxil edilməsini təmin edir.

ə) *Qeyd (Примечания)* əmri ilə cari sənəddə mətnə kursorun durduğu mövqedən qeydlərin daxil edilməsi təmin olunur.

f) *İstinad (Сноска)* əmri ilə səhifənin, sənədin sonuna cari sənəddəki ixtiyari söz və ya söz birləşməsi üçün əlavə nömrələnmiş informasiyanın daxil edilməsi təmin edilir.

g) *Ad (Имя)* əmri mətndə rəsmlərlə, cədvəllərlə, düsturlarla və s. avtomatik nömrələnmiş adların verilməsini təmin edir.

ğ) *Kəsişən istinad (Перекрестная ссылка)* əmri mətndə müxtəlif paraqrafların başlığına, cədvəllərə və s. istinad edilməsini təmin edir.

h) *Mündəricat və göstəricilər (Оглавление и указатели)* əmri cari sənəddə mündəricatın, şəkillərin siyahısının və s. müəyyən formatla yaradılmasını təmin edir.

x) *Rəsm (Рисунок)* əmri ilə cari sənədə şəkillər, avtofiqurlar və s. daxil etmək olur. Bu əmrin menyusuna *Şəkillər*, *Avtofiqurlar*, *WordArt obyekti* və *Diaqram* bölmələri aiddir. Birinci iki bölmə proqramda olan hazır şəkilləri və digər fayllarda olan şəkilləri sənədə daxil etməyə imkan verir. Avtofiqurlar bölməsi ilə proqramda olan hazır fiqurları, WordArt ilə proq-

ramdakı hazır yazı stillərinə gətirməklə ixtiyari yazını və *Diaqram* bölməsi ilə müxtəlif tipli diaqramları cari sənədə gətirmək olur.

ı) *Yazı* (*Надпись*) əmri mətn, şəkil, qrafik və s. üzərində digər mətn fraqmentinin, şəkilin, cədvəlin və s. yerləşdirilməsi əməliyyatını həyata keçirir.

i) *Fayl* (*Файл*) əmri ilə cari sənədə kursorun durduğu mövqeyə digər fayldan sənəd daxil etmək olur. Həmin faylın adı və ya ona gedən yol bu zaman açılan dialoq pəncərəsində göstərilməlidir.

j) *Obyekt* (*Объект*) əmri ilə cari sənədə kursorun durduğu mövqedən Windows-un digər proqramlarında yaradılan obyektləri daxil etmək olur.

k) *İçlik* (*Закладка*) əmri cari sənəddə avtomatik olaraq müəyyən seçilmiş sözü, obyekti nişanlamağa imkan verir.

q) *Hiperistinad* (*Гиперссылка*) əmri cari sənəddən müxtəlif fayllara istinadı təmin edir.

5) ***Format*** menyu bölməsinə aşağıdakı bölmələr aiddir:

a) *Şrift* (*Шрифт*) əmri cari sənəddə istifadə ediləcək şrifti, onun ölçülərini, rəngini, şriftlərarası intervalı və s. parametrləri seçməyə imkan verir.

b) *Abzas* (*Абзац*) əmri ilə cari sənəddə abzasların verilmə qaydaları müəyyən olunur. Bu zaman açılan pəncərədə abzas üçün uyğun parametrləri seçmək olur.

c) *Siyahı* (*Список*) əmri cari sənəddə sətirlər qarşısında müxtəlif nişanlar, rəqəmlər yerləşdirilməsini təmin edir.

ç) *Sərhəd və rəngləmə* (*Границы и заливка*) əmri ilə cari sənəddə qeyd olunmuş mətn hissəsini və ya bütün səhifəni çərçivə daxilinə almaq olur. Burada çərçivə üçün rəng və naxışlar da daxil etmək olur.

d) *Sütunlar* (*Колонки*) əmri cari sənəddəki mətnin bir neçə sütunlara bölünməsini təmin edir. Bu zaman açılan pəncərədə sü-tunların tipini, sayını, onlar arasındakı məsafəni vermək olur.

e) *Tabulyasiya* (*Табуляция*) əmri cari sənəddə aparılan tabulyasiyanın mövqeyini, mövqelərarası intervallararı təyin edir.

ə) *Bukvisa* (*Буквица*) əmri abzasın birinci simvolunun, bu zaman açılan pəncərədə parametrləri verilən xüsusi formada çıxarılmasını təmin edir.

f) *Mətnin istiqaməti* (*Направление текста*) əmri ilə cari

sənəddə cədvəldəki, mətndəki sözlərin şaquli və ya üfüqi istiqamətlərdə verilməsini təmin etmək olur.

g) *Registr* (*Регистр*) əmri ilə cari sənəddə qeyd olunmuş mətn hissəsində cümlənin ilk hərfinin böyük olmasını, bütün hərflərin kiçik olmasını, böyük hərflərin kiçik hərflərlə və kiçik hərflərlərin böyuk hərflərlə əvəz olunmasını təmin etmək olur.

ğ) *Fon* (*Фон*) əmri ilə cari sənəddə mətnin verildiyi fonun rəngini seçib təyin etmək olur.

h) *Mövzu* (*Тема*) əmri ilə cari sənəd üçün hazır yazı mövzularını seçib tətbiq etmək olur.

x) *Avtoformat* (*Автоформат*) əmri cari sənədi seçilmiş şablona uyğun avtomatik formatlaşdırır.

ı) *Stil* (*Стиль*) əmri ilə cari sənəddə qeyd olunmuş mətn hissəsi üçün yazı stili seçib tətbiq etmək olur.

i) *Obyekt* (*Объект*) əmri ilə cari sənəddə daxil edilmiş obyektin formatını seçmək olur.

6) ***Servis*** menyu bölməsində aşağıdakı alt bölmələr var:

a) *Düzgün yazılış* (*Правописание*) əmri cari sənəddəki mətndə orfoqrafik və qrammatik səhvlərin təyin edilməsi və aradan qaldırılmasını təmin edir.

b) *Dil* (*Язык*) əmri cari sənəddəki qeyd olunmuş müxtəlif dillərdə verilən mətn hissələrinin səhvlərinin düzəldilməsi üçün dillərin seçimini, eləcə də bu cür mətn hissələrində naməlum sözlərin sinonim və ya mənaca yaxın sözlərlə əvəz edilməsini və sətrin sonunda sözü hecaya bölməklə yeni sətrə keçilməsini təmin edir.

c) *Statistika* (*Статистика*) əmri cari sənəddə istifadə olunmuş sözlərin, durğu işarələrinin, simvolların, eləcə də səhifələrin, abzasların, sətirlərin sayını müəyyən edir.

ç) *Xülasə* (*Автореферат*) əmri cari sənəddəki mətnin xülasəsini hazırlayır.

d) *Avtoəvəz* (*Автозамена*) əmri cari sənəddə mətnin daxil edilməsi zamanı buraxıla biləcək səhvləri qabaqcadan avtomatik olaraq düzəltməyə və mətnin daxil edilməsi zamanı bir sıra simvolların başqaları ilə avtomatik əvəz olunmasına imkan verir.

e) *Düzəlişlər* (*Исправления*) əmri cari sənəddə edilmiş düzəlişləri mətndə əks etdirir, onları qəbul etmək və ya ondan imtina etməyə imkan verir.

ə) *Birləşmə* (*Слияние*) əmri yaradılmış məktub mətninə

müxtəlif ünvanları və ünvan sahibinin informasiyasını birləşdirməklə çox saylı məktubların avtomatik yaradılmasını təmin edir.

f) *Müdafiə qurmaq* (*Установить защиту*) əmri ilə cari sənədi, ona bir çox düzəlişlər edilməsindən müdafiə etmək olur.

g) *Müdafiədən imtina* (*Снять защиту*) əmri tətbiq olunmuş müdafiəni ləğv edir.

ğ) *Konvertlər və poçt nişanları* (*Конверты и наклейки*) əmri poçt konvertləri və nişanları hazırlayıb çap edilməsini təmin edir.

h) *Məktub ustası* (*Мастер писем*) əmri ilə avtomatik olaraq müxtəlif məktub mətnlərinin hazırlanması təmin edilir.

x) *Tənzimləmə* (*Настройка*) əmri ilə ekranda alətlər lövhəsinin verilməsi, bu lövhəyə yeni düymələrin əlavə edilməsi və ya düymələrin çıxarılması və s. əməliyyatları yerinə yetirmək olur.

ı) *Parametrlər* (*Параметры*) əmri ilə proqramda çap parametrlərinin dəyişdirilməsi, səhifənin müxtəlif ölçü vahidləri ilə ölçülməsi, avtomatik orfoqrafik yoxlama aparmaq və s. parametrlər təyin etmək olur.

7) ***Cədvəl*** bölməsinə aşağıdakı alt bölmələr aiddir:

a) *Cədvəlin çəkilməsi* (*Нарисовать таблицу*) əmri ilə ekrana verilən pəncərədəki alətlərdən istifadə etməklə cədvəlin damalarını, sətir və sütunlarını çəkmək olur.

b) *Əlavə etmək* (*Добавить*) əmri aşağıdakı bölmələri olan kontekst menyu açır:

1) *Cədvəl* əmri. Bu zaman açılan dialoq pəncərəsində göstərdiyimiz sayda sətir və sütunları olan cədvəl yaradılır.

2) *Soldan sütunlar* əmri ilə cari sütunun sol hissəsindən cədvələ sütun əlavə olunur.

3) *Sağdan sütunlar* əmri isə cari sütunun sağ tərəfindən cədvələ sütun əlavə edir.

4) *Yuxarıdan sətir* əmri cari sətirdən yuxarıda cədvələ yeni sətir əlavə edir.

5) *Aşağıdan sətir* əmri cari sətirdən aşağıda cədvələ yeni sətir əlavə edir.

6) *Damalar əmri* cari damaları sola və yuxarı sürüşdürməklə cədvələ yeni dama, sətir və sütun əlavə edir.

c) *Ləğv etmək* (*Удалить*) əmrinin kontekst menyusundan *Cədvəl, Sütunlar və Sətirlər* əmrləri cari və ya qeyd olun-

muş cədvəlləri və cədvəl sətir və sütunlarını ləğv edir. *Damalar* əmri isə cari damaları sola və yuxarı sürüşdürməklə cari və ya qeyd olunmuş dama, sətir və sütunları ləğv edir.

ç) *Qeyd etmək* (*Выделить*) əmri cari cədvəlin, sütunun, sətrin və damanın qeyd edilməsini təmin edir.

d) *Damaların* birləşdirilməsi (*Объеденить ячейки*) əmri cədvəldə qeyd edilmiş damaları birləşdirir.

e) *Damaların* bölünməsi (*Разбить ячейки*) əmri sədvəldə cari damanın bir neçə sətir və sütuna bölünməsini təmin edir.

ə) *Cədvəli bölmək* (*Разбить таблицу*) əmri cədvəli kursorun durduğu yerdən iki hissəyə bölür.

f) *Avtoformat* (*Автоформат*) əmri cədvələ proqramda olan hazır formatların tətbiq olunmasını təmin edir. Bu zaman açılan dialoq pəncərəsindən format nümunəsi seçilir.

g) *Avtoseçim* (*Автоподбор*) əmri ilə cədvəldəki informasiyanın eninə görə cari sütunun eni, pəncərənin eninə görə cədvəlin eni, sütunların hündürlüyü və s. nizamlanır.

ğ) *Başlıqlar* (*Заголовки*) əmri çox səhifəli cədvəllərdə birinci sətri başlıq kimi seçir və başlıq avtomatik olaraq cədvəlin bütün səhifələrində əks olunur.

h) *Mətni cədvələ keçirmək* (*Преобразовать в таблицу*) sənəddə qeyd olunmuş mətn hissəsini cədvəl formasına keçirir. Bu zaman açılan pəncərədə sətir və sütunların sayı göstərilməlidir.

x) *Cədvəli mətnə keçirmək* (*Преобразовать в текст*) əmri cari sənəddə qeyd olunmuş cədvəli mətn formasına keçirir.

ı) *Çeşidləmə* (*Сортировка*) əmri cədvəldəki informasiyanı sütun boyu əlifba sırasına görə, artma və azalmaya görə çeşidləməyə imkan verir. Bu zaman açılan dialoq pəncərəsində sütunları və çeşidləmə qaydasını müəyyən etmək tələb olunur.

i) *Düstur* (*Формула*) əmri ilə açılan dialoq pəncərəsində müxtəlif funksiya və düsturları daxil edərək onlar üzrə hesablamalar aparmaq olur.

j) *Cədvəl torunu əks etdirmək və ya gizlətmək* (*Отобразить или скрыть сетку*) əmri cədvəlin torunu gizlətmək və ya əks etdirməyə imkan verir.

k) *Cədvəlin xassələri* (*Свойства таблицы*) əmri cari sənəddə cədvəli sol, sağ və ya mərkəz istiqamətində nizamlayır, cədvəlin sətir və sütunlarının ölçülərini tənzimləyir və s.

8) *Pəncərə* menyu bölməsinin aşağıdakı alt bölmələri var:

a) *Yeni* (*Новое*) əmri cari proqram pəncərəsinin təkrarını alıb, onu yeni pəncərədə əks etdirir.

b) *Hamısını nizamlamaq* (*Упорядочить все*) əmri vasitəsilə açılmış pəncərələri onların açılma ardıcıllığına görə düzmək olur.

c) *Bölmək* (*Разделить*) əmri ilə cari pəncərəni kursoru tələb olunan yerə qoyub, siçanın sol düyməsini ikiqat sıxmaqla iki yerə bölmək olur. Buradakı *Bölünmənin ləğv edilməsi* əmri ilə pəncərənin bölünməsi əməliyyatını ləğv etmək olur. Qeyd edək ki, *Pəncərə* menyu bölməsinin alt bölmələrinin sonunda açılmış pəncərələrin adlarının siyahısı gətirilir. Həmin siyahıdan lazım olan pəncərəni seçib, açmaq olur.

9) Arayış menyu bölməsi proqramda iş qaydaları haqqında arayışlar əldə etmək üçün nəzərdə tutulmuşdur.

İndi isə MS Word 2000 mətn redaktorundakı bəzi vacib iş qaydalarını qeyd edək. Yeni sənəd yaratmaq üçün *Fayl* menyu bölməsinin *Yaratmaq* əmrini seçmək lazımdır. Açılan dialoq pəncərəsində lazım olan şablonu seçib, *OK* düyməsini sıxmaq tələb olunur. MS Word-ün şablonları *dot* ad genişlənməsinə malikdir. Adi sənədlər isə *Yeni sənəd* şablonu əsasında qurulur və bunun üçün ☐ düyməsindən istifadə olunur. Mövcud sənədin açılması üçün *Fayl* bölməsinin *Açmaq* əmrini seçmək və ya 📂 düyməsini sıxmaq lazımdır. Nəticədə açılan dialoq pəncərəsinin *Qovluq* yazı sahəsində diski seçib, ondan aşağıdakı sahədə qovluğu siçanın sol düyməsinin ikiqat sıxılması ilə açaraq, lazım olan faylı seçmək lazımdır. MS Word sənədi *doc* ad genişlənməsinə malikdir. Dialoq pəncərəsinin yuxarı sətrində açılmış qovluğun tərkibini aşağıdakı 4 formada göstərməyə imkan verən düymələr yerləşir:

1) Fayl və qovluqların siyahısı kimi;

2) Fayl və qovluqlar haqqında informasiya verilən cədvəl kimi;

3) Sol sahədə seçilmiş faylın xassələri göstərilən forma;

4) Sağ sahədə seçilmiş faylın fraqmenti göstərilən forma.

Qeyd edək ki, siyahıda yalnız MS Word sənədləri olan fayllar verilə-cəkdir. Digər tip və ya bütün tip faylları siyahıya çıxartmaq üçün pəncərədəki *Fayl tipləri* yazı sahəsindən istifadə edilir.

Sənədi yadda saxlamaq üçün *Fayl* bölməsinin *Saxlamaq* əmrinə keçmək və ya  düyməsini sıxmaq lazımdır. Sənəd yaddaşda birinci dəfə saxlanıldıqda ekrana verilən dialoq pəncərəsinin *Qovluq* sahəsində sənədin saxlanılacağı diski, ondan aşağıdakı sahədə isə qovluğu seçmək lazımdır. Faylın tipi sahəsində faylın formatını seçib, *Faylın adı* sahəsində sənəd faylına ad verib, buradakı *Saxlamaq* düyməsini sıxmaqla sənəd faylını yaddaşda saxlamaq olur. Qeyd edək ki, faylı təkrar yadda saxlayarkən, sənəd avtomatik olaraq elə həmin adla saxlanılır. Sənədi başqa adla və ya başqa qovluqda saxlamaq tələb olunduqda *Fayl* menyu bölməsinin *Necə saxlamaq* əmrin seçib, yuxarıdakı əməliyyatları yerinə yetirmək lazımdır. Sənədin bağlanması üçün *Fayl* bölməsinin *Bağlamaq* əmrini seçmək və ya sənəd pəncərəsindəki  düyməsini sıxmaq lazımdır.

İndi isə mətnlə iş qaydalarına baxaq. Sənəd pəncərəsində kursor mətnin daxil ediləcəyi mövqeyi bildirir. Səhifənin kənarına çatdıqda kursor avtomatik olaraq növbəti sətrin əvvəlinə keçir. Növbəti abzasın əvvəlinə keçmək üçün *Enter* düyməsini sıxmaq lazımdır. Sənəddə mətnin daxil edilməsinin iki rejimi var: *daxil etmək* və *əvəz etmək*.

Daxil etmək rejimində yeni simvolları daxil edərkən, buradakı mövcud mətn sağa doğru yerdəyişmə edir. *Əvəz etmək* rejimində isə köhnə mətn yenisi ilə əvəz olunur. Rejimlər arası keçid cari vəziyyət sətrindəki **ЗАМ** indikatorunun siçanın sol düyməsi ilə ikiqat sıxmaqla yerinə yetirilir. Mətnin müyyən bir hissəsi ilə iş aparmaqdan əvvəl onu qeyd etmək tələb olunur. Bunun üçün istifadə olunan qaydalara baxaq: kursoru ayrılacaq mətn hissəsinin əvvəlinə yerləşdirib, siçanın sol düyməsini sıxıb saxlayaraq kursoru fraqmentin sonuna qədər sürüşdürmək, bir sözü ayırmaq üçün siçanın sol düyməsi ilə onu ikiqat sıxmaq, bir cümləni ayırmaq üçün *Ctrl* düyməsini sıxıb saxlayaraq cümləni siçanın sol düyməsi ilə sıxmaq və s. Ayrılışı siçanın sol düyməsini sənəddə ixtiyari yerdə sıxmaqla ləğv etmək olar. Yeni ayrılış apardıqda əvvəlki ləğv olunur. Kursordan sağdakı simvolu *Delete*, soldakını isə *Backspace* düyməsi ilə silmək olur. Mətn hissəsini silmək üçün isə, əvvəlcə onu qeyd etməli, sonra *Delete* düyməsini sıxmaq lazımdır. Əgər mətn hissəsi ayırıb, sonra klaviaturadan yeni mətn yığsaq, yeni mətn qeyd etdiyimiz hissəsini əvəz edəcək. Abzası iki

abzasa bölmək üçün kursoru birinci abzasın sonuna yerləşdirib, *Enter* düyməsini sıxmaq lazımdır. İki abzasdan birini qurmaq üçün kursoru birinci abzasın sonuna yerləşdirib, *Delete* düyməsini sıxmaq və ya kursoru ikinci abzasın əvvəlinə qoyub *Backspace* düyməsini sıxmaq lazımdır. Axırıncı yerinə yetirilmiş redaktə etmə əmliyyatını ləğv etmək üçün *Düzəlişlər* bölməsinin *Ləğv etmək* əmrini və ya  düyməsini sıxmaq, bu əmliyyatı bərpa etmək üçün isə həmin bölmənin *Təkrar etmək* əmrini seçmək və ya  düyməsini sıxmaq lazımdır. Mətn hissəsinin təkrarını almaq üçün əvvəlcə bu hissəni ayırmaq, sonra *Düzəlişlər* bölməsinin *Təkrarını alma*q əmrini seçmək və ya  düyməsini sıxmaq lazımdır. Bundan sonra kursoru mətndə, təkrarın veriləcəyi yerə yerləşdirib *Düzəlişlər* bölməsinin *Daxil etmək* əmrini və ya  düyməsini sıxmaq lazımdır. Nəticədə qeyd olunmuş mətn hissəsinin təkrarı *Clipboard* adlanan mübadilə buferində yerləşdirilir və sonra sənəddə kursorun durduğu mövqedən daxil edilir. Buferdəki fraqmentdən ixtiyari sayda istifadə etmək olar, lakin buferdə yeni mətn hissəsi verildikdə əvvəlki hissə ləğv edilir. Mətn hissəsini bir yerdən başqa yerə köçürmək üçün isə bu hissəini qeyd edərək, *Düzəlişlər* bölməsinin *Kəsmək* əmrini seçmək və ya  düyməsini sıxmaq, sonra kursoru mətn hissəsinin köçürüləcəyi yerə yerləşdirib, *Düzəlişlər* bölməsinin *Daxil etmək* əmrini seçmək və ya  düyməsini sıxmaq lazımdır. Mətn hissəsini qeyd etdikdən sonra siçanın sol düyməsini sıxıb saxlayaraq, bu hissəsini mətnin lazım olan hissəsinə sürüşdürmək olar, bu zaman *Ctrl* düyməsini də sıxıb saxlasaq, həmin mətn hissəsinin təkrarı alınacaq.

MS Word 2000 redaktorunda 12 oyuqdan ibarət mübadilə buferi mövcuddur. Burada təkcə MS Word deyil, həm də digər proqramlardan məsələn, MS Excel-dən də fraqmentlərin təkrarını alıb, yerləşdirmək olur. *Mübadilə buferinin lövhəsini* ekrana çıxarmaq üçün *Görünüş* bölməsinin *Alətlər lövhəsi* alt bölməsinin *Mübadilə buferi* əmrinə keçmək lazımdır. Fraqmenti buferə yerləşdirmək üçün onu qeyd edib  düyməsini sıxmaq, buferdən sənədə gətirmək üçün isə onun buferdəki nişanını sıxmaq lazımdır.

Klaviaturada olmayan simvolun mətnə daxil edilməsi üçün kursoru simvolun daxil ediləcəyi yerə qoyub, *Daxil etmək* bölməsində *Simvol* əmrini seçmək lazımdır. Bu zaman açılan dialoq

pəncərəsində *Simvollar* bölməsini seçib, *Şrift* sahəsində şrift tipini təyin edib, cədvəldə lazımi simvolu seçdikdən sonra buradakı *Daxil etmək* düyməsini sıxmaq lazımdır.

Mətn hissəsini sənəddə axtarıb tapmaq üçün *Düzəlişlər* bölməsinin *Tapmaq* əmrindən istifadə edilir. Açılan dialoq pəncərəsində *Tapmaq* yazı sahəsində axtarılan mətn hissəsini yığıb buradakı *Tapmaq* düyməsini sıxmaq lazımdır. Burada pəncərədə axtarışla əlaqədar əlavə məlumatlar verməklə axtarış yönümünü təyin etmək olar. Müəyyən mətn hissəsini digəri ilə əvəz etmək üçün isə *Düzəlişlər* bölməsinin *Əvəz etmək* əmrini seçmək lazımdır. Açılan pəncərədə *Tapmaq* sahəsində əvəz ediləcək mətn fraqmentini, *Əvəz etmək* sahəsində isə onun yerinə veriləcək fraqmenti yığmaq lazımdır. Bundan sonra əvvəlcə *Tapmaq*, sonra isə *Əvəz etmək* düymələrini sıxmaq lazımdır.

Mətni formatlaşdırmaq üçün onu qeyd etmək lazımdır, əks halda seçilən format yalnız yeni yığılacaq mətn hissəsinə şamil ediləcək. Mətndəki simvolların parametrlərinin dəyişdirilməsi üçün Format bölməsinin Şrift əmrindən istifadə edilir. Bu zaman ekrana verilən eyni adlı dialoq pəncərəsinin Şrift yazı sahəsində şriftin tipini, *Yazı forması* sahəsində isə şrift üçün *Adi, Kursiv, Qalın, Qalın kursiv* formalarından hər hansı birini seçmək olur. *Ölçü* sahəsi ilə şriftin punktlarla (1 *punkt*=0,375 *mm*) ölçüsünü, *Altından xətt çəkmək* sahəsi ilə simvolların altından çəkilən xəttin tipini, *Rəng sahəsi* ilə simvolların rəngini seçmək olur. *Effektlər* çərçivəsindəki üstündən xətt çəkmək və üstündən ikiqat xətt çəkmək əmrləri ilə mətndə simvollar üzərindən uyğun olaraq bir və ikiqat xətlər vermək olur. Burada *Yuxarı indeks* və *Aşağı indeks* əmrləri ilə simvolların ölçüləri kiçildilir və uyğun olaraq mətn yuxarı və aşağı indekslərdən yerləşdirilir. Kölgə əmri ilə simvolların yanında kölgə yaranır, *Kontur* əmri simvolların yalnız konturunu təsvir edir, *Qaldırılmış* əmri simvolları vərəq səthindən qaldırılmış formada, *Həkk edilmiş* əmri isə vərəq səthinə həkk olunmuş formada təsvir edirlər. *Kiçildilmiş böyük hərflər* əmri kiçik hərfləri kiçildilmiş böyük hərflərə, *Hamısı böyük hərflərlə* əmri isə kiçik hərfləri böyük hərflərə çevirir. Nəhayət, *Gizlədilmiş* əmri qeyd olunmuş simvolları çap zamanı kağız üzərinə çıxarmır. Simvolların interval və vəziyyətini dəyişdirmək üçün *Şrift* dialoq pəncərəsinin İnterval bölməsindən istifadə edilir.

Yığılmış mətndə simvolların registrini dəyişdirmək üçün

Format bölməsinin *Registr* əmrini seçmək lazımdır. Açılan dialoq pəncərəsindəki Cümlələrdə olduğu kimi, *Hamısı kiçik hərflə, Hamısı böyük hərflə, Böyük hərflə başlamaq və Registri dəyişdirmək* bölmələrindən birini seçib *OK* düyməsini sıxmaq lazımdır. Bu bölmələr qeyd olunmuş hissədə uyğun olaraq cümlədəki birinci sözü böyük hərflə başlayır, bütün hərfləri kiçik edir, bütün hərfləri böyuk edir, bütün sözləri böyük hərflə başlayır və böyük hərfləri kiçik, kiçik hərfləri isə böyük hərflərlə əvəz edir.

Abzasların parametrlərinin müəyyən edilməsi üçün *Format* bölməsinin *Abzas* əmrindən istifadə edilir. Bu zaman açılan dialoq pəncərəsində abzaslar üçün lazım olan parametrləri seçib, *OK* düyməsini sıxmaq lazımdır.

Tabulyasiyadan mətn və ədəd sütunlarının dəqiq düzümünü təmin etmək üçün istifadə edilir. Tabulyasiya mövqelərini təyin etmək üçün *Format* bölməsinin *Tabulyasiya* əmrindən istifadə edilir. Bu zaman açılan dialoq pəncərəsində *Sol kənarına* görə, *Mərkəzə görə, Sağ kənarına* görə bölmələri mətni tabulyasiya mövqelərinə nəzərən uyğun olaraq sol kənarına, mərkəzinə və sağ kənarına görə düzümünü tənzimləyir. Buradakı *Bölgüyə görə* bölməsi ədədləri onluq vergülə görə, mətni isə sağ kənarına görə tənzimləyir.

MS Word redaktorunda qeydli, nömrələnmiş və çoxsəviyyəli nömrələnmiş siyahılar qurmaq olur. Siyahı elementi mətnin abzasları qəbul edilir. Siyahını yaratmaq üçün siyahı elementləri abzasları qeyd edib, *Format* bölməsinin *Siyahı* əmrini seçmək lazımdır. Bu zaman açılan dialoq pəncərəsində siyahı parametrlərini vermək olur.

Cari sənəddə səhifələrin parametrlərini təyin etmək üçün *Fayl* bölməsinin *Səhifənin parametrləri* əmri seçilir. Bu zaman açılan dialoq pəncərəsində *Sahələr bölməsi* ilə səhifənin kənarlarından buraxılacaq məsafələr santimetr ölçü vahidi ilə təyin edilir. Cilidləmə sahəsi ilə soldan və ya yuxarıdan cilidləmə üçün sahənin eni müəyyən olunur. Eləcə də burada *Kağızın ölçüləri* bölməsi ilə səhifənin ölçüləri, onun *kitab* və ya *albom* formalı olması təyin edilir.

Mətn avtomatik olaraq səhifələrə bölünək üçün kursoru bölgü aparılacaq yerə qoyub, *Daxil etmək* bölməsinin *Bölmək* əmrinə keçmək lazımdır. Açılan dialoq pəncərəsindəki *Yeni səhifə* sahəsinə keçib, *OK* düyməsini sıxmaq lazımdır. Əgər sənəddə

səhifələr müxtəlif parametrli olmalıdırsa, onda onu bir neçə bölməyə ayırmaq lazımdır. Sənədə yeni bölmə daxil etmək üçün pəncərədəki *Növbəti səhifədən*, *Cari səhifədən*, *Cüt səhifədən* və ya *Tək səhifədən* sahələrinin birindən istifadə edilir. Sənəddə səhifələri nömrələmək üçün *Daxil etmək* bölməsinin *Səhifə nömrələri* əmrini seçmək lazımdır. Açılan pəncərədə *Vəziyyət* sahəsində nömrənin səhifədəki mümkün qoyulma vəziyyətlərindən (yuxarı və ya aşağı) birini seçmək tələb olunur. Burada eləcə də nömrəni səhifənin solunda, mərkəzində, sağında və s. qaydada yerləşdirmək olur. Pəncərədəki Birinci səhifədəki nömrə bölməsi ilə birinci səhifədəki nömrəni qoymamaq da olar.

Kolontitul – sənədin hər bir səhifəsində yuxarı və ya aşağı hissələrdə çap olunan mətn və ya şəkildir. Kolontitul yaratmaq üçün Görünüş menyu bölməsinin *Kolontitul* əmrini seçmək lazımdır. Bu zaman ekran səhifələrinin nişanlanması iş rejiminə keçir və ekrana Kolontitulların alətlər lövhəsi çıxarılır. Lövhədən istifadə edərkən, kolontitulların parametrlərini təyin etmək olur.

Sənədi çapa göndərmək üçün *Fayl* bölməsinin *Çap* əmrindən istsfadə edilir. Bu zaman açılan dialoq pəncərəsində Ad yazı sahəsində printerin tipini seçmək, Səhifələr çərçivəsində isə çapa verilən səhifələrin diapazonunu təyin etmək olur. Bu çərçivədə *Hamısı*, *Cari*, *Qeyd olunmuş fraqment* və *Nömrələr* bölmələrindən birini seçmək olar. *Nüsxələr* yazı sahəsində çapa verilən səhifələrdən neçə nüsxə çap olunacağı müəyyən olunur. Çapa çıxarmaq siyahısında *Diapazonun bütün səhifələrini*, *Tək səhifələrini* və *Cüt səhifələrini* qiymətləndirən hər hansı birini seçmək olur.

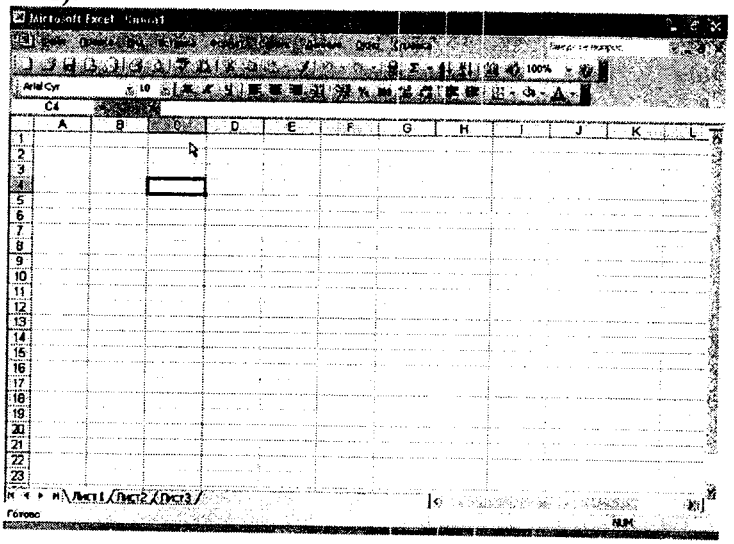
MS Word digər proqramlarda yaradılmış, eləcə də proqramın öz şəkil çəkmək bölməsinin köməyi ilə yaradılan qrafik obyektləri sənədə daxil etməyə imkan verir. *Şəkil çəkmək* lövhəsini *Daxil etmək* bölməsinin *Alətlər lövhəsi* alt bölməsindən çağırmaq olur. Bu lövhənin köməyi ilə xətt, ox, ellips, düzbucaqlı, çevrə və s. qurmaq olur. Daxil edilmiş qrafik obyekti rəngləmək, formasını, rəngini dəyişmək, ona digər obyektləri əlavə etmək və s. mümkündür. Digər proqramlarda yaradılan qrafik obyekti sənədə daxil etmək üçün *Daxil etmək* bölməsinin rəsm alt bölməsindəki *Fayldan* əmrini seçmək lazımdır. Açılan dialoq pəncərəsində *Qovluq* yazı sahəsində diski, bu sahədən aşağıdakı sahədə isə rəsm olan faylın yerləşdiyi qovluğu seçmək lazımdır. Proqramda olan hazır şəkilləri sənədə daxil etmək üçün isə *Daxil etmək* bölməsinin

Rəsm alt bölməsindəki *Şəkillər* əmrini seçmək lazımdır.

Sənədə cədvəl daxil etmək üçün *Cədvəl* menyu bölməsinin Əlavə etmək alt bölməsinin *Cədvəl* əmrini seçmək lazımdır. Açılan dialoq pəncərəsində cədvəlin sətri və sütunlarının sayını verib, *OK* düyməsini sıxmaq lazımdır. Cədvəllə iş üçün nəzərdə tutulmuş bütün əmrlər menyusunun *Cədvəl* bölməsində yerləşir.

2.8 MS Excel cədvəl prosessoru

Microsoft Excel 2000 – elektron cədvəllərinin yaradılması və emalı üçün nəzərdə tutulmuş proqramdır. Proqram EHM-ə yükləndikdə ekrana proqramın iş stolu adlanan pəncərə verilir (şək. 8.1).



Şəkil 8.1

Pəncərənin birinci sətri başlıqdan, ikinci sətri isə menyu sətrindən ibarətdir. Menyu sətrindən aşağıda piktoqram formasında verilmiş düymələr ardıcıllığından ibarət alətlər lövhəsi yerləşir. Bu düymələr proqramın menyusunda olan və ən çox istifadə edilən əmrləri təkrarlayır və onlardan istifadəni sadələşdirir. Alətlər menyu sətrində aşağıdakı iki növ alətlər lövhəsi – *Standart* və *Formatlaşdırma* lövhələri yerləşir. Bu lövhədən aşağıda adətən düsturlar sətri, pəncərənin aşağı hissəsində isə *Cari vəziyyət sətri*

yerləşir. Proqram cədvəllərlə aşağıdakı iki rejimdə: *Adi* (əməliyyatların çoxunun yerinə yetirilməsi üçün əlverişli olan rejim) və *Səhifələrin nişanlanması* (çapdan əvvəl cədvəlin son formatlaşdırılması üçün əlverişli rejim) rejimlərində işi təmin edir.

Əvvəlcə proqramın menyu bölmələrinə baxaq. Menyu sətrində 9 bölmə var: *Fayl (Файл), Düzəlişlər (Правка), Görünüş (Вид), Daxil etmək (Вставка), Format (Формат), Servis (Сервис), Verilənlər (Данные), Pəncərə (Окно)* və *Arayış (Справка).*

1) *Fayl* menyu bölməsinə aşağıdakı alt bölmələr aiddir:

a) *Yaratmaq* (*Создать*) əmri *Ümumi* və *Qərar* bölmələri olan dialoq pəncərəsi açır. *Ümumi* bölmədə kitab və istifadəçinin yaratdığı şablonlar, *Qərar* bölməsində isə *Sifarişlər, Avans hesabatları* və s. şablonlar yerləşir. Bu şablonlardan ixtiyari birini seçməklə yeni kitabı (sənədi) onun əsasında yaratmaq olur.

b) *Açmaq* (*Открыть*) əmri mövcud sənədi proqram pəncərəsinə çağırır.

c) *Bağlamaq* (*Закрыть*) əmri cari sənəd pəncərəsini bağlayır.

ç) *Saxlamaq* (*Сохранить*) əmri sənədi yaddaşda saxlayır.

d) *Necə saxlamaq* (*Сохранить как*) əmri cari sənədi başqa adla, başqa yerdə, digər tip sənəd kimi yaddaşda saxlamağa imkan verir.

e) Web-*səhifə* kimi saxlamaq (*Сохранить как Web-страницу*) əmri sənədi yaddaşda Web-səhifə kimi saxlamağa imkan verir.

ə) *İşçi oblastı yadda saxlamaq* (*Сохранить рабочую область*) əmri işçi oblastı yadda saxlayır. Bu zaman açılan dialoq pəncərəsində əvvəlcə açılmış kitab səhifələrini yadda saxlayıb, sonra işçi oblastı yadda saxlamaq olur.

f) Web-*səhifəyə ilkin baxış* (*Предварительный просмотр Web-страницы*) əmri Web-səhifə kimi yadda saxlanılmış sənədin ilkin baxışını təmin edir.

g) *Səhifələrin parametrləri* (*Параметры страницы*) əmri səhifənin ölçülərini və s. parametrləri müyyən etməyə imkan verir.

ğ) *Çap sahəsi* (*Область печати*) əmri Müəyyən etmək və

Ləğv etmək bölmələri olan menyu açır. Onlardan birincisi cədvəldə qeyd olunmuş damaları çap olunacaq fraqment kimi təyin edir, ikncisi isə ayrılmış fraqmentin çap sahəsi kimi təyin olunmasını ləğv edir.

h) *Çap* (*Печать*) əmri cari sənədi çapa göndərir. Bu zaman açılan dialoq pəncərəsində çap parametrləri verilir.

x) *Xassə* (*Свойства*) əmri cari sənəd haqqında məlumatlar almağa və əlavə məlumatlar daxil etməyə imkan verir.

ı) *Göndərmək* (*Отправить*) əmri cari sənədi elektron poçt və ya faksla başqa istifadəçiyə göndərə bilir.

i) *Çıxış* (*Выход*) əmri cari proqram pəncərsini bağlayıb, proqramdan çıxışı təmin edir.

2) *Düzəlişlər* menyu bölməsində aşağıdakı alt bölmələr var:

a) *Ləğv etmək* (*Отменить*) əmri axırıncı yerinə yetirilmiş əməliyyatı ləğv edir.

b) *Təkrar etmək* (*Повторить*) əmri axırıncı yerinə yetirilmiş əməliyyatı təkrar edir.

c) *Kəsmək* (*Вырезать*) əmri qeyd olunmuş fraqmenti kəsib, mübadilə buferinə yerləşdirir.

ç) *Təkrarını almaq* (*Копировать*) əmri qeyd olunmuş fraqmentin təkrarını alıb, mübadilə buferində saxlayır.

d) *Daxil etmək* (*Вставка*) əmri mübadilə buferindəki fraqmenti cari damaya kursorun durduğu mövqedən daxil edir.

e) *Xüsusi daxil etmək* (*Специальная вставка*) əmri mübadilə buferindəki informasiyanı tamamilə və ya digər formalarda cari damaya daxil etməyə imkan verir.

ə) *Hiperistinad kimi daxil etmək* (*Вставить как гиперссылку*) əmri digər proqramlarda yaradılmış və mübadilə buferində saxlanmış informasiyanı hiperistinad şəklində cari damaya daxil etməyə imkan verir.

f) *Tamamlama* (*Заполнить*) əmri qeyd olunmuş damalara informasiyanın daxil edilməsini avtomatlaşdırır, təkrarlanan və müəyyən addımla artan ədədlər daxil edir.

g) *Silmək* (*Очистить*) əmri ilə cari damadakı informasiyanı tam şəkildə, yalnız formatı, yalnız informasiyanı və yalnız qeydi silmək olur.

ğ) *Ləğv etmək* (*Удалить*) cari sənəddə qeyd olunmuş damaları və ya cari sətri, sütunu, damanı ləğv edir.

h) *Vərəqi ləğv etmək* (*Удалить лист*) əmri cari vərəqi ləğv edir.

x) *Vərəqin təkrarının alınması və ya yerinin dəyişdirilməsi* (*Переместить/копировать лист*) əmri ilə cari vərəqin təkrarını almaq və ya kitabda vərəqlərin yerləşmə ardıcıllığını dəyişdirmək olur.

ı) *Tapmaq* (*Найти*) əmri cari sənəddə verilmiş simvol, söz və ya söz birləşmələrini nümunə əsasında axtarıb tapmağa imkan verir.

i) *Əvəz etmək* (*Заменить*) əmri cari sənəddə simvol, söz və ya söz birləşmələrini nümunə əsasında taparaq, verilmiş simvol, söz və ya söz birləşmələri ilə əvəz etməyə imkan verir.

j) *Keçid* (*Перейти*) əmri ünvanı göstərilən damaya avtomatik keçidi təmin edir.

k) *Əlaqələr* (*Связи*) əmri cari sənəddə olan Windows-un digər proqramlarında yaradılmış obyektlərin mənbələri ilə əlaqəsini göstərir.

q) *Obyekt* (*Объект*) əmri cari sənəddə olan digər proqramlarda yaradılmış digər obyektlər üzərində iş aparmağa imkan verir.

3) *Görünüş* menyu bölməsinə aşağıdakı alt bölmələr daxildir:

a) *Adi* (*Обычный*) əmri cari sənəddə adi iş rejimini təyin edir.

b) *Səhifənin nişanlanması* (*Разметка страницы*) əmri isə eyni adlı iş rejimini təyin edir.

c) *Alətlər lövhəsi* (*Панел инструментов*) əmri ilə iş stoluna alətlərin daxil edilməsini və ya çıxarılmasını təmin etmək olar.

ç) *Cari vəziyyət sətri* və *Düstur sətri* (*Строка состояния*, *Строка формул*) əmrləri ilə proqram pəncərəsinə uyğun olaraq cari vəziyyət və düstur sətrlərini daxil etmək və ya oradan çıxarmaq mümkündür.

d) *Kolontitullar* (*Колонтитулы*) əmri cari vərəqə kolontitulların daxil edilməsini və onların redaktə edilməsini təmin edir.

e) *Qeydlər* (*Примечания*) əmri cari vərəqədəki bütün qeydləri ekrana çıxarır.

ə) *Təqdim olunma* (*Представления*) əmri cari vərəqin bir neçə ifadə formasını, çap parametrlərini yadda saxlayıb, ehtiyac olduqda istifadə etməyə imkan verir.

f) *Bütün ekran boyu* (*Во весь екран*) əmri ekranda yalnız menyu sətrini və sənəd pəncərəsindəki informasiyanı əks etdirməyə imkan verir.

g) *Miqyas* (*Масштаб*) əmri cari sənəddə pəncərənin görünüşünü müxtəlif miqyaslarda ifadə edə bilir.

4) *Daxil etmək* menyu bölməsinə aşağıdakı alt bölmələr aiddir:

a) *Dama* (*Ячейка*) əmri ilə açılan dialoq pəncərəsindəki *Sağa sürüşdürməklə* damalar, *Aşağı sürüşdürməklə* damalar, *Sətir*, *Sütun* bölmələri vasitəsilə uyğun olaraq cari damadan sağda, aşağıda yeni dama, sətir və sütünlar əlavə etmək olur.

b) *Sətirlər* (*Строки*) əmri cari damadan əvvəl yeni sətir əlavə edir.

c) *Sütun* (*Столбец*) əmri cari damadan əvvəl yeni sütun əlavə edir.

ç) *Vərəq* (*Лист*) əmri cari vərəqdən əvvəl yeni vərəq əlavə edir.

d) *Diaqram* (*Диаграмма*) əmri verilən qiymətlər əsasında diaqram, qrafik, histoqram qurulmasını təmin edir.

e) *Səhifənin bölünməsi* (*Разрыв страницы*) əmri cari səhifəni bölərək yeni səhifəyə keçidi təmin edir.

ə) *Funksiya* (*Функция*) əmri ilə açılan pəncərədə lazım olan funksiyanı seçib, funksiya arqumentini verib, *OK* düyməsini sıxmaqla funksiyanı hesablayıb, nəticəni cari damada yerləşdirmək olur.

f) *Ad* (*Имя*) əmri cari damaya və ya qeyd olunmuş damalara müəyyən adlar daxil etməyə imkan verir.

g) *Qeyd* (*Примечание*) əmri verildikdə açılan yazı sahələrinə damalardakı verilənlər haqqında izahedici qeydlər daxil etmək olur.

ğ) *Rəsm* (*Рисунок*) əmri cari sənəddə proqramda və ya fayllardan şəkillər, avtofiqurlar, WordArt mətnləri və s. daxil etməyə imkan verir.

h) *Xəritə* (*Карта*) əmri cari sənədə proqramda olan xəritələri daxil etməyə imkan verir.

x) *Obyekt* (*Объект*) əmri Windows-un digər proqramlarında hazırlanmış obyektləri cari sənədə daxil etməyə imkan verir.

i) *Hiperistinad* (*Гиперссылка*) əmri ilə açılan dialoq pəncərəsində istinad olunacaq məlumatın ünvanını və ya fayla gedən yolu göstərib *OK* düyməsini sıxmaqla həmin istinadı cari damaya yerləşdirmək olur.

5) *Format* menyu bölməsinə aşağıdakı alt bölmələr aiddir:

a) *Dama* (*Ячейка*) əmri ilə açılan dialoq pəncərəsindəki *Ədəd, Nizamlama, Şrift, Sərhəd, Görünüş* və *Müdafiə* bölmələri ilə uyğun olaraq cari damada və ya qeyd olunmuş damalarda ədədlərin verilmə formasını (ədəd, pul vahidi, tarix və zaman, kəsr və s.) təyin etməklə informasiyanın yazılış istiqamətini müəyyən etmək, qeyd olunmuş damaları birləşdirmək və s., şrift seçmək, cari damanı seçilmiş formalı çərçivəyə almaq, cari dama və ya qeyd olunmuş damaları seçdiyimiz rənglə rəngləmək və nəhayət, cari damadakı informasiyanı düzəlişlərdən müdafiə etmək olur.

b) *Sətir* (*Строка*) əmri ilə cari sətrin hündürlüyünü təyin etmək, bu sətrin hündürlüyünü mətnin şriftinə uyğunlaşdırmaq və cari sətri gizlətmək və ya ekrana çıxarmaq olur.

c) *Sütun* (*Столбец*) əmri ilə cari sütunun enini təyin etmək, onu gizlətmək və ya əks etdirmək olur.

ç) *Vərəq* (*Лист*) əmri cari vərəqin adını, fonunun rəngini dəyişdirməyə, onu gizlətməyə və ya əks etdirməyə imkan verir.

d) *Avtoformat* (*Автоформат*) əmri ilə cari sənəddə qeyd olunmuş damaları, bu əmrlə açılan dialoq pəncərəsindəki formatlardan hər hansı birinə uyğunlaşdırmaq olur.

e) *Şərti formatlaşdırma* (*Условное форматирование*) əmrinin açdığı dialoq pəncərəsində tələb olunan şərtləri daxil etməklə yeni format təyin etmək olur.

ə) *Stil* (*Стиль*) əmri ilə cari sənəddəki yazı stilini dəyişdirmək olur.

6) *Servis* menyu bölməsinə aşağıdakı alt bölmələr aiddir:

a) *Orfoqrafiya* (*Орфография*) əmri cari sənəddə olan mətnin orfoqrafik və qrammatik yazılışını yoxlayır.

b) *Avtoəvəz* (*Автозамена*) əmri cari sənəddə mətnin daxil edilməsi zamanı mətndə avtomatik düzəlişlərin aparılmasını və qeyd olunan simvolların başqaları ilə avtomatik əvəz olunmasını

təmin edir.

c) *Düzəlişlər* (Исправления) əmri cari sənəddə aparılmış düzəlişlərə nəzarət etməyə, onların qəbul olunması və ya onlardan imtina edilməsinə imkan yaradır.

d) *Müdafiə* (*Защита*) əmri cari vərəqi, kitabı edilə biləcək düzəlişlərdən müdafiə edir.

e) *Parametrin seçilməsi* (*Подбор параметра*) əmri ilə açılan pəncərədə nəticənin veriləcəyi damanın ünvanını, onun qiymətini və dəyişiləcək damanın ünvanını göstərib *OK* düyməsini sıxmaqla düsturda iştirak edən damalardakı ədədi qiyməti tələb olunan nəticəyə uyğun nizamlamaq olar.

ə) *Asılılıqlar* (Зависимости) əmri cari sənəddə düsturda baş vermiş səhvlərin mənbəyini, asılı və təsvir edən damaları göstərir.

f) *Tənzimləmə* və *Parametrlər* (Настройка, Параметры) əmrləri ilə proqram pəncərəsini tələb olunan formada tənzimləmək olur.

7) *Verilənlər* menyu bölməsində aşağıdakı alt bölmələr var:

a) *Çeşidləmə* (Сортировка) əmri cari sənəddəki verilənləri əlifba sırası ilə, eləcə də, artma və azalma üzrə çeşidləməyə imkan verir.

b) *Filtr* (*Фильтр*) əmri ilə cari sənəddə yalnız müəyyən şərtləri ödəyən verilənlərin əks olunmasını təmin etmək olur.

c) *Forma* (*Форма*) əmri ilə açılan pəncərədəki *Əlavə etmək*, *Ləğv etmək*, *Geriyə*, *Davamı* və *Kriteriya* bölmələrinin köməyilə cari vərəqə verilənlər daxil etmək, onları ləğv etmək, əvvəlki, sonrakı verilənlərə baxmaq və müəyyən şərtləri ödəyən verilənləri axtarıb tapmaq olur.

ç) *Yekun* (*Итоги*) əmri aralıq, yekun və ümumi yekun nəticələri tapmağa imkan verir.

d) *Yoxlama* (*Проверка*) əmri ilə açılan pəncərədə daxil ediləcək verilənlərin tipini, alacaq qiymətlər oblastını verməklə, verilənlərin daxil edilməsi zamanı səhvlərin qarşısını almaq olur.

e) *Sütünlar üzrə mətn* (*Текст по столбцам*) əmri ilə cari damadakı mətn bir neçə sütun şəklində ifadə olunur.

ə) *Konsolidasiya* (*Консолидация*) əmri ayrı-ayrı diapazonlarda olan ədədi verilənlər üzərində əməliyyatlar aparmağa imkan yaradır.

f) *Qruplar və strukturlar* (Группы и структуры) əmri cari sənəddə qeyd olunmuş sətir və ya sütunları qruplaşdırır, eləcədə, düstur və istinadlar əsasında qeyd olunmuş səhifənin strukturunu yaradır.

g) *Yekun cədvəllər* (*Сводные таблицы*) əmri cari verilənlər üçün yekun cədvəl hazırlayır.

ğ) *Xarici verilənlər* (*Внешние данные*) əmri ilə verilənlər bazasına göndərilən sorğu əsasında alınan verilənləri cari sənədə daxil etmək olur.

h) *Verilənlərin yeniləşdirilməsi* (*Обновление данных*) əmri sorğu ilə cari sənədə daxil edilmiş verilənləri yeniləşdirə bilir.

8) *Pəncərə* menyu bölməsinin aşağıdakı alt bölmələri var:

a) *Yeni* (*Новое*) əmri cari pəncərənin tərkibi ilə eyni olan yeni proqram pəncərəsi açır.

b) Yerləşdirmək (*Расположить*) əmri ilə ekrana verilən dialoq pəncərəsində Yanaşı, Aşağıdan yuxarı, Soldan sağa, Pillələrlə formalarından hər hansı birini seçərək açılmış bütün pəncərələri ekrana həmin formada gətirmək olur.

c) *Gizlətmək* (*Скрыть*) əmri ilə cari pəncərəni gizlətmək olur.

ç) *Əks etdirmək* (*Отобразить*) əmri ilə cari pəncərəni ekranda əks etdirmək olur.

d) *Bölmək* (*Разделить*) əmri ilə cari pəncərəni seçdiyimiz hissədən iki yerə bölmək olar.

e) *Sahəni qeyd etmək* (*Закрепить области*) əmri cari pəncərədə seçilmiş sahəni qeyd etməyə imkan verir.

9) *Arayış* menyu bölməsi proqramda iş qaydaları haqqında müxtəlif arayışlar almağa imkan verir.

İndi isə MS Excel 2000 proqramında iş qaydalarına baxaq. MS Excel faylı işçi kitab adlanır. İşçi kitab isə adları (Vərəq 1, Vərəq 2,...) işçi kitabın pəncərəsinin aşağı hissəsində sadalanan işçi vərəqlərdən ibarətdir. Bu vərəq adlarını siçanın sol düyməsi ilə sıxmaqla kitab daxilində bir vərəqdən digərinə keçmək olur. İşçi vərəq 256 sütundan və 65536 sətirdən ibarət cədvəldir. Sütunlar latın əlifbasının hərfləri ilə, sətirlər isə rəqəmlərlə adlandırılır. Cədvəlin hər bir daması sətrin və sütunun adından ibarət ünvana malikdir. Məsələn, *F7* ünvanı, damanın *F* sütunu ilə 7 sətrinin kəsişdiyi yerdə durduğunu bildirir. Cədvəlin damalarından biri hə-

mişə aktiv olur. Aktiv dama çərçivəyə alınır. Damanı aktiv etmək üçün onu siçanın sol düyməsi ilə sıxmaq kifayətdir. Bir-birinin yanında yerləşən bir neçə damanı qeyd etmək üçün kursoru damalardan birinə yerləşdirib siçanın sol düyməsini sıxıb saxlamaqla siçanı hərəkət etdirmək lazımdır. Bir-birinə yaxın olmayan dama qrupları ayırarkən, əvvəlcə bir qrup dama qeyd edib, sonra *Ctrl* düyməsini sıxıb saxlayaraq digər dama qruplarını da qeyd etmək lazımdır. Cədvəldə bütöv sütun və ya sətr qeyd etmək üçün, onun adını siçanın sol düyməsi ilə sıxmaq lazımdır. Bir neçə sütun və ya sətir qeyd etmək tələb olunduqda isə, birinci sütun və ya sətrin adını siçanın sol düyməsi ilə sıxıb saxlayaraq, qeyd etməni bütün oblasta şamil etmək olar. Bir neçə vərəqi qeyd etmək üçün *Ctrl* düyməsini sıxıb saxlayaraq vərəq adlarını siçanın sol düyməsi ilə sıxmaq lazımdır.

Damaya verilənləri daxil etmək üçün onu aktiv edib, verilənləri klaviaturadan daxil etmək lazımdır. Bu zaman verilənlər dama ilə yanaşı redaktə etmə sətrinə də çıxarılır. Daxil etməni sona çatdırmaq üçün *Enter* düyməsini sıxmaq lazımdır. Bu zaman verilənlərin daxil edilməsi prosesi sona çatır və növbəti dama aktiv olur. Damadakı verilənləri redaktə etmək üçün damanı aktiv edib, *F2* düyməsini sıxmaq və ya damanı siçanın sol düyməsi ilə ikiqat sıxmaq lazımdır. Bu zaman kursor damaya gətirilir, redaktə işinin sonunda *Enter* düyməsini sıxmaqla redaktə etmək rejimindən çıxmaq olur. Yeni işçi kitab yaratmaq üçün *Fayl* bölməsindəki *Yaratmaq* əmrini seçib, açılan dialoq pəncərəsində işçi kitabın yaradılacağı şablonu qeyd edib, *OK* düyməsini sıxmaq lazımdır.

Adi işçi kitablar kitab şablonu əsasında yaradılır. Bu cür işçi kitabı  düyməsini sıxmaqla da yaratmaq olar. Mövcud işçi kitabın açılması üçün Fayl bölməsinin *Açmaq* əmrini seçmək və ya  düyməsini sıxmaq lazımdır. Nəticədə açılan pəncərənin *Qovluq* sahəsində kitabın yerləşdiyi diski, ondan aşağıdakı sahədə isə qovluğu və kitabın özünü seçmək lazımdır. İşçi kitabı yaddaşda saxlamaq üçün *Fayl* bölməsinin *Saxlamaq* əmrini seçmək və ya  düyməsini sıxmaq lazımdır. Sənəd yaddaşda ilk dəfə saxlanılarkən, ekrana verilən pəncərədə *Qovluq* sahəsində kitabın saxlanılacağı diski, ondan aşağıdakı sahədə qovluğu, *Fayl tipi* sahəsində formatı, *Fayl adı* sahəsində isə kitabın adını vermək

və buradakı *Saxlamaq* düyməsini sıxmaq lazımdır. Kitab yaddaşda təkrarən saxlandıqda, o avtomatik olaraq həmin faylda saxlanıla-caqdır. Kitabı başqa adla və ya başqa qovluqda yadda saxlamaq tələb olunduqda, Fayl bölməsinin Necə saxlamaq əmrini seçmək lazımdır. İşçi kitabı bağlamaq üçün *Fayl* bölməsinin *Bağlamaq* əmrini seçmək və ya  düyməsini sıxmaq lazımdır.

Cədvəldəki hesablamalar düsturlarla yerinə yetirilir. Düstur riyazi operatorlardan, qiymətlərdən, funksiya və dama adlarına istinadlardan ibarət ola bilər. Düsturun yerinə yetirilmə nəticəsi düsturun yerləşdiyi damadakı qiymət olur. Düstur «=» işarəsi ilə başlayır və düsturda +, -, *, / hesab əməl operatorlarından istifadə edilə bilər. Düsturdakı hesablama qaydaları adi riyazi qanunlarla tənzimlənir. Düstura misal olaraq *=(A5+B9)*C7; =D7*T13+K17* və s. göstərmək olar. Sabitlər – damaya daxil edilən və hesablamalar zamanı dəyişdirilə bilməyən mətni və ədədi qiymətlərdir. Damaya müraciət onun koordinatlarının daxil edilməsi ilə yerinə yetirilir. Boş damanın qiyməti sıfır qəbul edilir. Damaya müraciət iki tip ola bilir:

1) nisbi müraciət – bu zaman dama, düstur verilmiş damaya nisbətən sürüşdürülməklə işarə olunur (məsələn, *F7*).

2) mütləq müraciət – bu zaman dama $ işarəli dama koordinatları ilə işarə olunur (məsələn, F7*). Eləcə də bu tiplərin kombinasiyalarından da istifadə olunur (məsələn, *F$7*).

Damalar qrupuna müraciət üçün xüsusi işarələrdən istifadə olunur:

1) : (iki nöqtə) işarəsi damalar blokuna müraciəti formalaşdırır. Bu işarə ilə blokun yuxarı sol və aşağı sağ damaları işarələnir. Məsələn, *C4:D6* ifadəsi *C4, C5, C6, D4, D5, D6* damalarına müraciət deməkdir.

2) ; (nöqtə vergül) işarəsi damaların birləşdirilməsi deməkdir. Məsələn, *D2:D4; D6:D8* ifadəsi *D2, D3, D4, D6, D7, D8* damalarına müraciət deməkdir. Düsturu damaya daxil etmək üçün «=» işarəsini və hesablama düsturunun ifadəsini vermək lazımdır. *Enter* düyməsini sıxmaqla damada hesablamanın nəticəsini almaq olar.

MS Excel proqramında funksiya müəyyən məsələlərin həlli üçün bir neçə hesablama əməliyyatını birləşdirə bilir. Burada

funksiyalar bir və ya bir neçə arqumenti olan düsturlardır. Arqument kimi ədədi qiymətlər və ya dama ünvanları göstərilir. Məsələn, *=СУММ(A5:A9) A5, A6, A7, A8, A9* damalarının cəmini, *=СРЗНАЧ(B4:B6)* isə *B4, B5, B6* damalarındakı qiymətlərin ədədi ortasını tapır. Funksiyalar biri digərinin tərkibində də ola bilir. *Məsələn,*

=СУММ(C1:C20)+ОКРУГЛ(СРЗНАЧ(D4:D9);2); və s.

Funksiyanı damaya daxil etmək üçün damanı düstur üçün qeyd edib sonra *Daxil etmək* bölməsinin *Funksiya* əmrini seçməklə və ya düyməsini sıxmaqla açılan dialoq pəncərəsində *Kateqoriya* sahəsində funksiyanın tipini, *Funksiya* sahəsində isə funksiyanı seçib *OK* düyməsini sıxmaq lazımdır. Burada pəncərənin Число1, Число2 və s. sahələrində funksiyanın arqumentlərini (ədədi qiymət və ya damalara müraciət) daxil etmək tələb olunur. Qeyd edək ki, damaya СУММ cəmləmə funksiyasını düyməsini sıxmaqla da daxil etmək olar.

Cədvəl şəklində daxil edilmiş verilənlərin emalı üçün və eyni tipli düsturların daxil edilməsi üçün düsturlar massivindən istifadə etmək əlverişli olur. Məsələn, *B1, C1, D1, E1* damalarındakı ədədlərin mütləq qiymətlərini hesablamaq üçün hər bir damaya ayrı-ayrılıqda düstur tətbiq etmək əvəzinə, bütün damalar üçün massiv daxil etməklə bir düsturla kifayətlənmək olar. Proqramda massivi qeyd etmək üçün düsturlar massivi ətrafına { } fiqurlu mötərizə əlavə edilir. Məsələn, *{=ABS(B1:E1)}*. Düsturlar massivini yaratmaq üçün aşağıdakı əməliyyatlar ardıcıllığını yerinə yetirmək lazımdır: düsturlar massivinin yerləşdiyi damaları qeyd etmək, düsturu adi qayda ilə daxil edib, arqumentlər kimi arqument-damalar qrupunu daxil edib, sonda *OK* düyməsi əvəzinə *Ctrl+Shift+Enter* düymələri kombinasiyasından istifadə etmək lazımdır.

Damaların daxil edilməsi üçün onların əvəz edəcəkləri damaları cədvəldə qeyd edib, *Daxil etmək* bölməsində *Damalar* alt bölməsini seçmək lazımdır. Nəticədə açılan pəncərədə daxil edilən elementin tipini (sağa sürüşdürməklə dama, aşağı sürüşdürməklə dama, sətir, sütun) seçib, *OK* düyməsini sıxmaq lazımdır. Sətir və ya sütunların daxil edilməsi üçün onların əvəz edəcəkləri sətir və ya sütunları qeyd edib, *Daxil etmək* bölməsinin *Sətirlər* və ya *Sütunlar* alt bölməsini seçmək lazımdır. Cədvəlin

elementlərini ləğv etmək üçün onları qeyd edib, *Düzəlişlər* menyu bölməsinin *Ləğv etmək* əmrini seçmək lazımdır. Cədvəldəki damaların daxilindəkiləri onların özlərini ləğv etmədən silmək üçün bu damaları qeyd edib, *Düzəlişlər* menyu bölməsinin *Təmizləmək* əmrini seçmək və ya *Delete* düyməsini sıxmaq lazımdır. Damaların tərkibinin təkrarını almaq və yerini dəyişdirmək üçün mübadilə buferindən (Clipboard) istifadə edilir. Damanın tərkibinin təkrarını almaq üçün əvvəlcə həmin damaları qeyd etmək, sonra isə *Düzəlişlər* bölməsinin *Təkrarını almaq* əmrini seçmək və ya düyməsini sıxmaq lazımdır. Buferdə saxlanılan verilənləri damaya daxil etmək üçün həmin damaları qeyd edib *Düzəlişlər* bölməsinin *Daxil etmək* əmrini seçmək və ya düyməsini sıxmaq lazımdır. Damaların tərkibini başqa damalara köçürmək üçün tərkibi köçürüləcək damaları qeyd edib, *Düzəlişlər* menyu bölməsinin *Kəsmək* əmrini seçmək və ya düyməsini sıxmaq lazımdır. Sonra bu fraqmentin köçürüləcəyi oblastın yuxarı sol damasını qeyd edib *Düzəlişlər* bölməsinin *Daxil etmək* əmrini seçmək və ya düyməsini sıxmaq lazımdır. Cədvəlin ixtiyari obyekti üzərində siçanın sağ düyməsini sıxmaqla bu obyektin emalı üçün tələb olunan əmrlər ardıcıllığı verilən kontekst menyusu açılır.

MS Excel 2000 proqramında 12 yaddaş oyuğu olan mübadilə buferi var. Onun köməyi ilə cədvəl fraqmentlərinin təkcə Excel daxilində deyil, həm də Windows-un digər proqramları daxilində təkrarını alıb köçürmək olur. Buferi ekrana çıxarmaq üçün *Görünüş* menyu bölməsinin *Alətlər lövhəsi* alt bölməsinin *Mübadilə buferi* əmrini seçmək lazımdır. Fraqmentin təkrarını buferə salmaq üçün onu qeyd edib, düyməsini sıxmaq lazımdır. Buferdəki fraqmenti sənədə yerləşdirmək üçün onun buradakı nişanını siçanın sol düyməsi ilə sıxmaq lazımdır.

Kitabdakı vərəqlərin adını dəyişdirmək üçün onun yarlıkını siçanın sol düyməsi ilə ikiqat sıxıb yeni adı daxil edirlər. İşçi kitabın vərəqlərinin təkrarını almaq və yerlərini dəyişdirmək üçün vərəqlərinin təkrarı alınacaq kitabı və vərəqlər köçürüləcək kitabı açmaq, vərəqi qeyd edib, *Düzəlişlər* menyu bölməsinin *Yerini dəyişmək/Təkrarını almaq* əmrini seçmək lazımdır. Bu zaman açılan pəncərənin *Kitaba* yazı sahəsində vərəqləri qəbul edəcəyi kitabın adını seçmək, *Vərəqin qarşısına* yazı sahəsində isə qarşısında

köçürülən vərəq yerləşdiriləcək vərəqi seçmək lazımdır. Bu zaman vərəqin təkrarını almaq tələb olunarsa, onda pəncərədəki uyğun bölməni qeyd etmək lazımdır. Vərəqin ləğv edilməsi üçün onu aktiv edib, *Düzəlişlər* bölməsinin *Vərəqi ləğv etmək* əmrini seçmək lazımdır. Vərəq daxil etmək üçün əvəzlənəcək vərəqi qeyd edib *Daxil etmək* menyu bölməsinin *Vərəq* əmrini seçmək lazımdır.

Cədvəldə hər bir ədədi müxtəlif formatlarda vermək olar. Formatı dəyişdirmək üçün damanı qeyd edib, *Format* bölməisnin *Damalar* əmrini seçmək lazımdır. Nəticədə açılan pəncərədə *Ədəd* bölməsini seçib, *Ədəd formatları* siyahısında formatın tipini, ondan sağdakı sahədə isə formatın parametrlərini seçmək lazımdır.

Cədvəldə susmaqla damalar standart en və hündürlüyə malikdir. Sətrin hündürlüyü şriftin ölçüsü ilə təyin edilir. Sətrin hündürlüyünü və sütünun enini onların başlıqlarının sərhədlərini lazım olan qiymətə qədər sürüşdürməklə dəyişmək olar. Əgər sütunların başlıqları sərhəddində siçanın sol düyməsini ikiqat sıxsaq, sütunun eni ən uzun tərkibli damanın eni qədər olur. Cədvəldə sütunların (sətirlərin) enini (hündürlüyünü) dəqiq təyin etmək üçün sütunları (sətirləri) qeyd edib, *Format* menyu bölməsinin *Sütun (Sətir)* alt bölməsində *En (Hündürlük)* əmrini seçmək lazımdır. Nəticədə açılan pəncərədə *Sütunun eni (Sətrin hündürlüyü)* yazı sahəsində qiymətlər daxil edib, *OK* düyməsini sıxmaq lazımdır.

Sətir və ya sütunları cədvəldə gizlətmək üçün həmin sətir və ya sütunları qeyd edib, *Format* bölməsinin *Sətirlər* və ya *Sütunlar* alt bölmələrini seçib, buradakı *Gizlətmək* əmrini vermək lazımdır. Sətir və ya sütunları cədvəldə əks etdirmək üçün isə gizlədilmiş sətir və ya sütunun hər iki tərəfindən sətirləri və ya sütunları qeyd edib, *Format* bölməsində *Sətirlər* və ya *Sütunlar* alt bölmələrini seçib *Əks etdirmək* əmrini vermək lazımdır.

MS Excel-in köməyilə verilənlər bazası yaradıb onu emal etmək olur. Burada verilənlər bazası – eyni tipli yazılışlardan (sətirlərdən) ibarət cədvəldir. Cədvəlin sütunları verilənlər bazasında yazı sahələridir. Yazı sahələrinin adları üçün verilənlər bazasının birinci sətri ayrılır. Verilənlər bazası ilə işləmək üçün əvvəlcə uyğun cədvəli yaratmaq lazımdır. Əgər cədvəldə dama qeyd edib *Verilənlər* bölməsinin verilənlər bazasının emal etmə əmrlərindən ixtiyari birini seçsək, proqram avtomatik olaraq bü-

tün cədvəli təyin edir və emal edir. Verilənlərin cədvəldə çeşidlənməsi üçün bir damanı qeyd edib *Verilənlər* bölməsinin *Çeşidləmə* əmrini seçmək lazımdır. Bu zaman ədədlər artım və azalma sırası ilə, mətn əlifba sırası ilə, məntiqi ifadələr isə əvvəlcə yalan, sonra doğru qiyməti verilməklə çeşidlənirlər. Verilənlər bazası ilə iş üçün xarakterik olan əməliyyatlar (axtarış, çeşidləmə, yekunların vurulması) zamanı proqram cədvəli avtomatik olaraq verilənlər bazası kimi qəbul edir.

Diaqramma – cədvəllərdəki verilənlərin müqayisəsi və analizi üçün onların qrafik şəkildə ifadəsidir. Diaqrammada damalardakı ədədi verilənlər nöqtə, xətt, zolaq, sütün, sektor və s. şəklində təsvir olunur. Diaqrammanı qurmaq üçün iş vərəqində onun uyğun verilənlərini qeyd edib, *Daxil etmək* bölməsinin *Diaqramma* əmrini seçmək lazımdır. Açılan dialoq pəncərəsində diaqrammanın tipini, formatını və digər parametrlərini təyin etmək olar. Diaqrammanın tip və qurulma parametrlərini dəyişdirmək üçün siçanın sağ düyməsini onun üzərində sıxmaqla açılan kontekst menyusunda uyğun əmri seçmək lazımdır. Diaqrammanı ləğv etmək üçün onu qeyd edib *Delete* düyməsini sıxmaq lazımdır.

Cədvəlləri çapa göndərməzdən əvvəl *Fayl* bölməsinin *Səhifənin parametrləri* əmri ilə səhifələrin parametrlərini təyin etmək olur. Açılan pəncərədə *Səhifə* bölməsində vərəqin ölçüləri və oriyentasiyası, təsvirin miqyası və çapın keyfiyyəti təyin edilir. Cədvəli çapa vermək üçün *Fayl* menyu bölməsinin *Çap* əmrini seçmək lazımdır. Açılan pəncərədə *Ad* sahəsində printeri, *Çap* bölməsində çapa veriləcək səhifələri təyin etmək olur. Burada *Çapa çıxarmaq* bölməsi ilə qeyd olunmuş diapazon vərəqlərini, qeyd olunmuş vərəqləri və ya bütün kitabı çapa vermək olar. *Surətlərin sayı* bölməsi çapa çıxarılan nüsxələrin sayını təyin edir.

II BÖLMƏ

PROQRAMLAŞDIRMANIN ƏSASLARI

III FƏSİL

PROQRAMLAŞDIRMAYA GİRİŞ

3.1. Alqoritm anlayışı

İnformatika elminin əsaslarından biri olan alqoritm anlayışı, EHM-dən əvvəl yaranmış və riyaziyyatın əsas anlayışlarından biri olmuşdur. «Alqoritm» sözü özbək riyaziyatçısı Əl-Xorezminin (IX əsr) adından götürülüb və riyaziyatda dörd cəbri əməliyyatın (toplama, çıxma, vurma və bölmə) yerinə yetirilmə qaydalarını işarə etmək üçün istifadə olunub. Hal-hazırda alqoritm anlayışı təkcə riyaziyyatda istifadə olunmur. Bu anlayışdan insan fəaliyyətinin bir çox sahələrindən istifadə olunur. Məsələn, istehsal prosesinin idarə edilməsi alqoritmindən, şahmat oyununun alqoritmindən, labirintdə yolun tapılması alqoritmindən, kosmik raketin uçuşunun idarə edilməsi alqoritmindən və s. danışmaq olar.

Alqoritm – verilmiş məsələnin həllinin tapılması üçün zəruri olan əməliyyatlar ardıcıllığının dəqiq və sadə təsviridir. Alqoritm anlayışının izahı üçün «alqoritmin icraçısı» anlayışının böyük əhəmiyyəti vardır. Belə ki, alqoritm hər hansı konkret bir icraçı üçün, məsələn, insan üçün, xüsusi maşın – EHM üçün və s. qurulur. Beləliklə, demək olar ki, alqoritm – qoyulmuş məsələni həll etmək və ya müəyyən məqsədə çatmaq üçün müəyyən əməllər ardıcıllığının yerinə yetirilməsi haqqında icraçıya verilən dəqiq və aydın göstərişlərdir.

Alqoritm, onun tərifini açıqlayan aşağıdakı əsas xassələrə malikdir:

1. ***Diskretlik xassəsi.*** Alqoritm diskret (latınca – fasiləli) olmalıdır. Diskret dedikdə, fasilələlik kəsilməzliyə qarşı qoyulur. Məsələn, hər hansı kəmiyyətin zamana görə diskret dəyişməsi – müəyyən fasilələrlə (sıçrayışlarla) baş verən dəyişmədir və ya tam ədədlər çoxluğu həqiqi ədədlər çoxluğuna qarşı diskretdir.

Bu xassə ondan ibarətdir ki, alqoritm məsələnin həlli prosesini, sadə addımların yerinə yetirilməsi ardıcıllığı şəklində ifadə etməlidir və hər bir addımın yerinə yetirilməsi üçün sonlu zaman fasiləsi tələb olunur, yəni başlanğıc verilənlərin araşdırılması və nəticənin alınması zamana görə diskret yerinə yetirilir.

2. ***Müəyyənlik xassəsi.*** Bu xassə ondan ibarətdir ki, alqoritmin hər bir addımı dəqiq, birqiymətli və aydın olmalıdır. Bu xassəyə əsasən alqoritmin yerinə yetirilməsi mexaniki xarakter daşımalı və həll olunan məsələ haqqında əlavə məlumat və ya göstəriş tələb etməməlidir.

3. ***Kütləvilik xassəsi.*** Bu xassəyə əsasən müəyyən məsələnin həlli üçün qurulan alqoritm ümumi şəkildə qurulur, yəni alqoritm yalnız başlanğıc qiymətləri ilə fərqlənən müəyyən sinif məsələlərin həlli üçün tətbiq edilə bilsin. Burada başlanğıc qiymətlər də, alqoritmin tətbiq oblastı adlanan hər hansı eyni bir oblastdan seçilə bilər. (Bəzi hallarda başlanğıc qiymətlər heç verilməyə də bilər).

4. ***Nəticəlilik və ya sonluluq xassəsi.*** Bu xassə onu bildirir ki, alqoritm, qoyulmuş məsələnin həllinə sonlu sayda addımların yerinə yetirilməsi ilə gətirməlidir, yəni alqoritm sonlu sayda addımdan sonra başa çatmalı və verilmiş məsələnin həlli tapılmalıdır.

3.2 Alqoritmlərin tipləri və ifadə formaları

Qurulmuş alqoritmi bir neçə üsulla ifadə etmək olar:

1. *Sözlərlə*
2. *Xüsusi sxemlər – blok-sxemlərlə*
3. *Alqoritmlərin yazılışı üçün xüsusi dildə, alqoritmik dildə.*

Alqoritmin sözlərlə ifadəsindən, alqoritm icraçısı insan olduqda istifadə olunur. Bu zaman alqoritmin addımları nömrələnir ki, onlara müraciət etmək mümkün olsun. Misal üçün Evklid alqoritmini nəzərdən keçirək. Bu alqoritm verilmiş iki natural ədəd üçün ən böyük ortaq böləni (ƏBOB) təyin edir. Verilmiş ədədləri M və N ilə işarə edək, onda

1) Əgər $M<>N$ olarsa, 2 bəndinə, əks halda 5 bəndinə keç,

2) Əgər $M>N$ olarsa, 3 bəndinə, əks halda 4 bəndinə keç,

3) M-dən N-i çıxmalı və bu fərqin nəticəsini M-ə mənimsətməli. 1 bəndinə keçməli.

4) N-dən M-i çıxmalı və bu fərqin nəticəsini N-ə mənimsətməli. 1 bəndinə keçməli.

5) ƏBOB-un M-ə bərabər olduğunu qəbul etmək.

6) Son.

Digər bir misal, verilmiş kvadrat tənliyin həlli üçün alqoritm:

1) Tənliyin a, b, c əmsallarını daxil etmək.

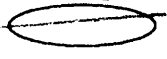
2) $D = b^2 - 4ac$ ifadəsini hesablamaq.

3) Əgər $D < 0$ olarsa, 5 bəndinə, əks halda 4 bəndinə keç.

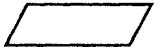
4) $x_1 = \frac{-b + \sqrt{D}}{2a}$, $x_2 = \frac{-b - \sqrt{D}}{2a}$ hesablamalı

5) Hesablamaları qurtarmalı. Son.

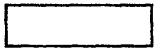
Alqoritmlərin blok-sxemlər vasitəsilə ifadəsi zamanı onlar əyani qrafik formada ifadə olunurlar. Alqoritmin əmrləri onların yerinə yetirilmə ardıcıllığını göstərən oxlarla birləşdirilən bloklar içərisinə yerləşdirilir. Bu blokların qrafik təsviri üçün aşağıdakı standartlar qəbul edilmişdir:



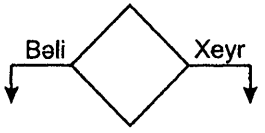
Başlanğıc və sonu göstərmək üçün.



Paraleloqram. Daxil etmə və çıxışı göstərmək üçün.

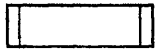


Düzbucaqlı. Hesablamaları göstərmək üçün.



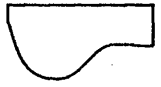
Romb. Şərtin yoxlanılması üçün.

Dövrün başlanğıcını göstərmək üçün.



Alt proqramların göstərilməsi üçün.

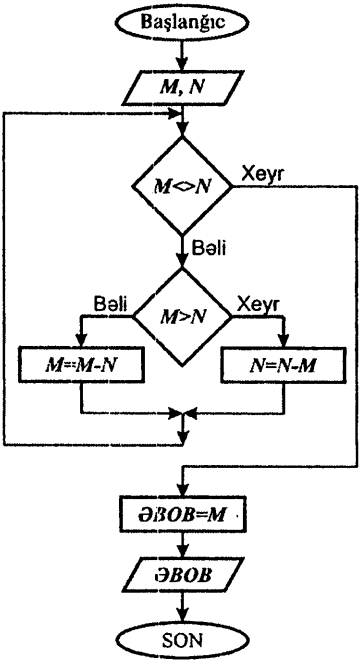
Eyni bir səhifədə yerləşən blokları birləşdirmək üçün.



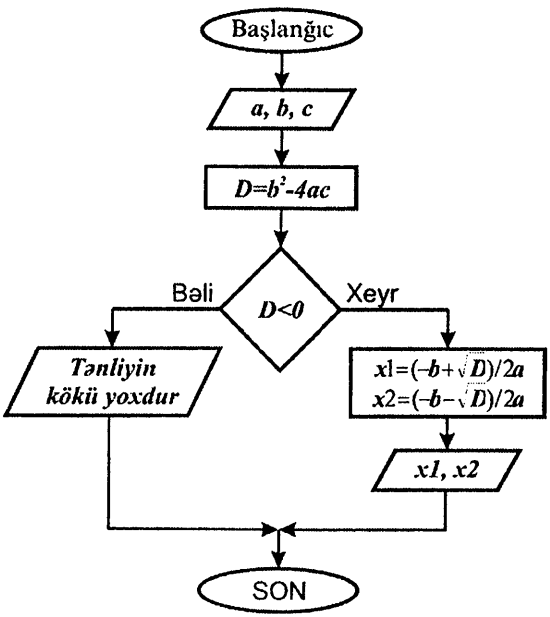
Cavabın çap qurğusuna verildiyini göstərmək üçün.

Müxtəlif səhifələrdə olan blokları birləşdirmək üçün.

Blok-sxemlərdə alqoritmin bloklar daxilində verilən əmrləri daha qısa şəkildə ifadə olunur, belə ki, burada həndəsi fiqurlar sözləri əvəz edir. Məsələn, Evklid alqoritmi üçün uyğun blok-sxem aşağıdakı şəkildə olar.

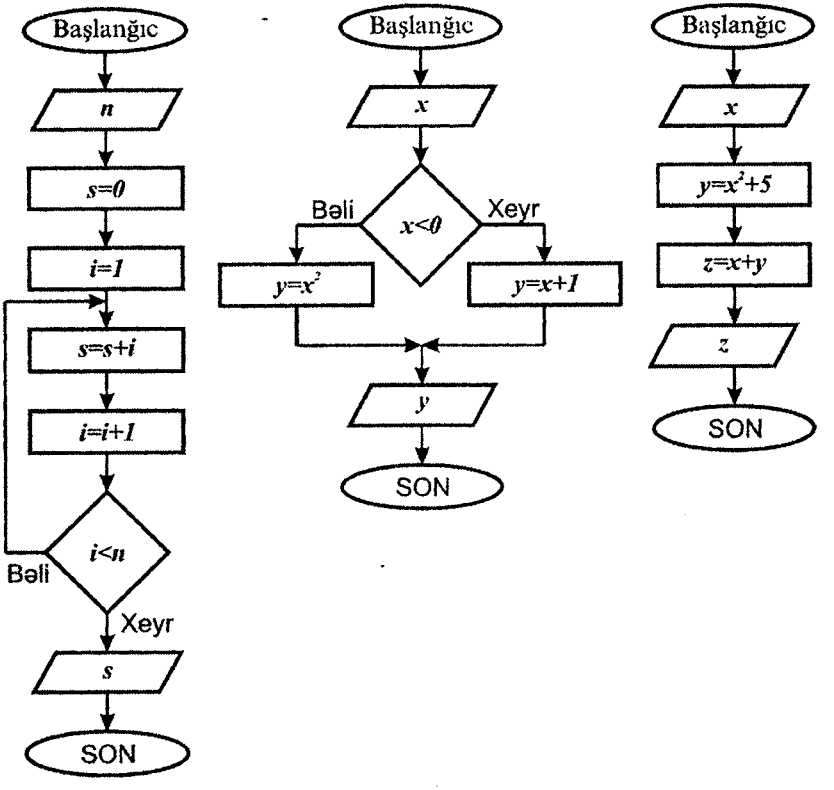


Kvadrat tənliyin həlli üçün alqoritmin blok-sxemi isə aşağıdakı kimi olar:



Alqoritmlər strukturuna görə üç əsas tipə bölünürlər: xətti strukturlu, budaqlanan strukturlu və dövr strukturlu alqoritmlər. Əgər alqoritmin təşkil etdiyi N sayda addımlar, bir-birinin ardınca başlanğıcdan sona qədər ardıcıl yerinə yetirilirsə, belə alqoritmə xətti strukturlu alqoritm deyilir. Əgər alqoritmin addımlarının yerinə yetirilmə ardıcılığı, müəyyən şərtlərin ödənilməsindən asılı olaraq dəyişirsə, belə alqoritm budaqlanan strukturlu alqoritm adlanır. Şərt isə məntiqi ifadə olub, yalnız iki qiymət ala bilər: «hə» - əgər şərt doğrudursa və «yox» - əgər şərt yalandırsa. Əgər alqoritmin müəyyən addımlar, ardıcıllığı, müəyyən şərtin ödənməsindən asılı olaraq bir neçə dəfə təkrarlanırsa, belə alqoritm dövr strukturlu alqoritm adlanır.

Məsələn, bu alqoritmlər üçün aşağıdakı misallara baxaq:



Bu məsələlərin şərtləri uyğun olaraq aşağıdakı kimidir: 1) İlk n natural ədədin cəmini tapmalı; 2) Əgər $x<0$ olarsa, y-ə x^2, əks halda y-ə $x+1$ mənimsətməli ; 3) x ədədi verilib, $y=x^2+5$, $z=x+y$ ifadələrini hesablayıb, z-i çapa verməli.

3.3 Alqoritmlərin qurulma qaydaları

Alqoritmlərin qurulma qaydalarını, aşağıdakı misallar üzərində tədqiq edək:

1) İki natural N və M ədədlərinin K nisbətini və L qalıq həddinin tapılması, harada $L<M$ və K-tam ədəddir (sxem1).

2) $S=\begin{cases}-1, & x<0\\ 0, & x=0\\ 1, & x>0\end{cases}$ funksiyasının hesablanması (sxem2).

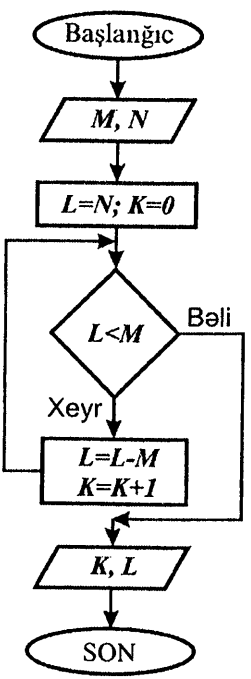
3) x, y, z ədədləri içində maksimalının tapılması (sxem3).

4) $y=f(x)$ funksiyasının x_0-dan x_k-ya qədər h sabit addımı ilə dəyişən x arqumenti üçün qiymətlərinin hesablanması (sxem4).

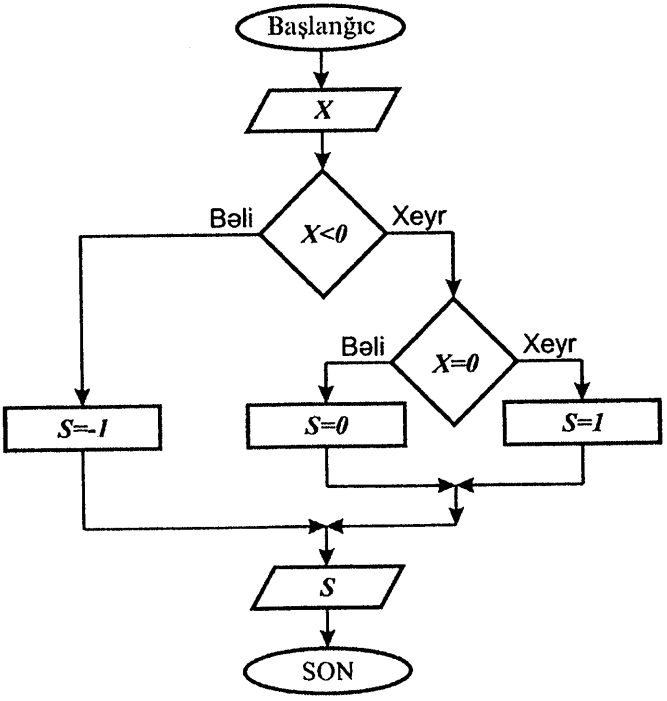
5) $S=\sin x+\sin(3x)/3+\ldots=\sum_{i=1}^{\infty}\frac{\sin(2i-1)x}{2i-1}$ sırasının cəmini verilmiş x üçün ε dəqiqliyi ilə hesablamalı (sxem5).

6) Verilmiş sonlu ədədlər ardıcıllığında maksimal və minimal elementlərin təyini (sxem6).

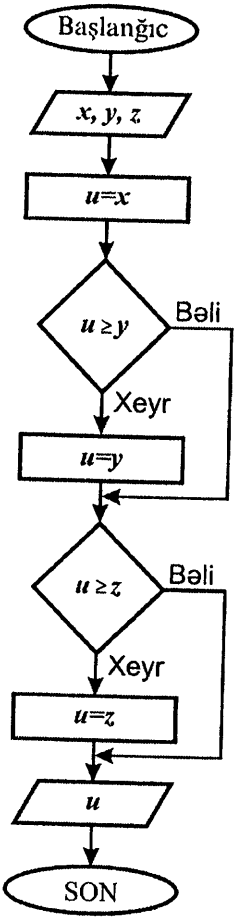
7) Verilmiş müəyyən inteqralı trapeslər düsturu ilə ε dəqiqliyi ilə hesablamalı (sxem7).



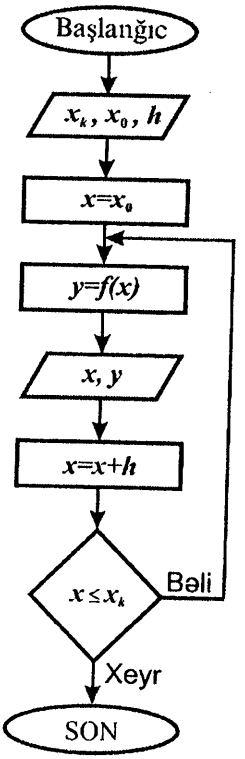
Sxem 1.



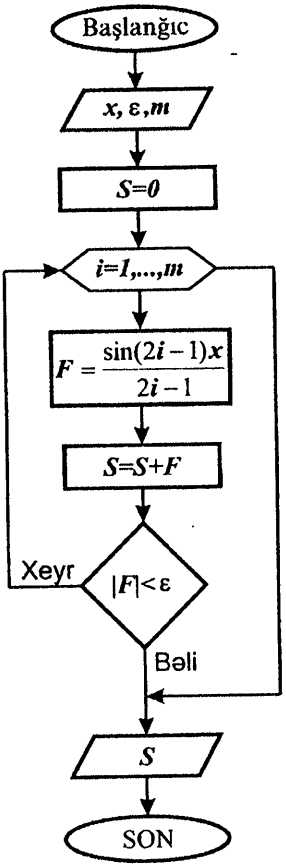
Sxem 2.


Başlanğıc
x, y, z
u=x
u ≥ y
Bəli
Xeyr
u=y
u ≥ z
Bəli
Xeyr
u=z
u
SON

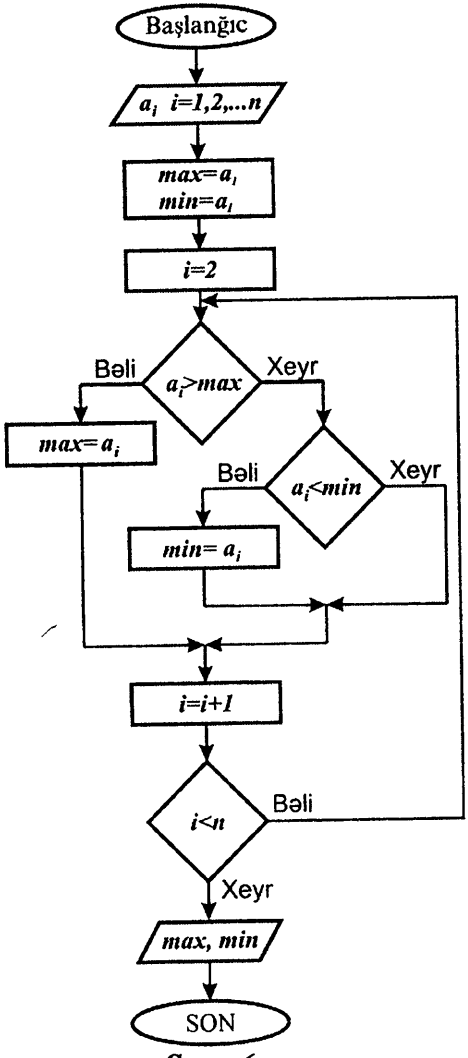
Sxem 3.


Başlanğıc
x_k, x_0, h
$x=x_0$
$y=f(x)$
x, y
$x=x+h$
$x \le x_k$
Bəli
Xeyr
SON

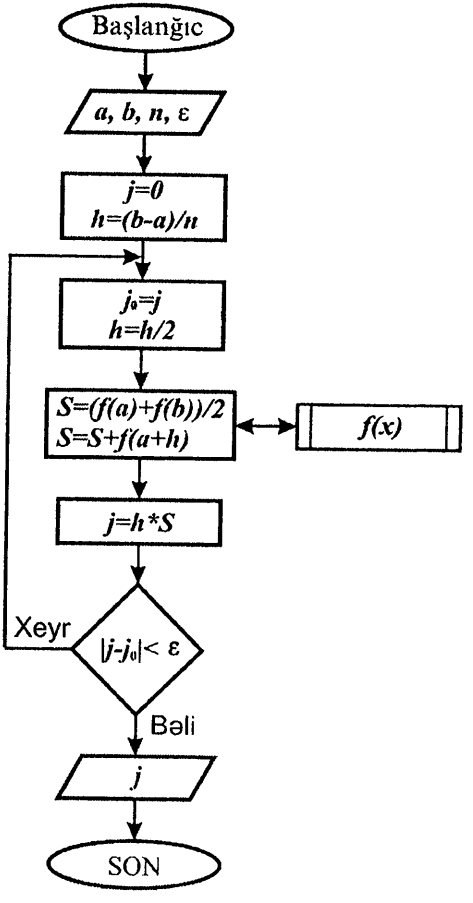
Sxem 4.


Başlanğıc
x, ε, m
S=0
i=1,...,m
$F = \frac{\sin(2i-1)x}{2i-1}$
S=S+F
Xeyr
|F|<ε
Bəli
S
SON

Sxem 5.


Başlanğıc
a_i i=1,2,...n
max=a_i
min=a_i
i=2
Bəli
a_i>max
Xeyr
max=a_i
Bəli
a_i<min
Xeyr
min=a_i
i=i+1
i<n
Bəli
Xeyr
max, min
SON

Sxem 6.



Sxem 7.

3.4. Alqoritmik dillər

Hesablama texnikasının inkişafı proqramlaşdırma dillərinin yaranmasını təmin etdi. Proqramlaşdırma dili - məsələnin EHM-də həlli üçün verilən alqoritminin yazıldığı dildir.

Birinci nəsl EHM-də proqramlaşdırma, yalnız maşın dilində aparılırdı. Maşın dili müəyyən əməliyyatların, əsasən cəbri əməliyyatların ədədi şəkildə kodlaşdırılma qaydalarının ardıcıl-

lığından ibarət olur. EHM-də yerinə yetirilməli olan hər bir əməliyyat, maşın dilində əmrlər şəklində ifadə olunur. Hər bir əmr isə, maşın əməliyyatı adlanan və informasiya emalı prosesinin hər hansı elementar hissəsi olan bir əməliyyatı ifadə edir.

Əmrdə ümumi şəkildə, aparılacaq əməliyyatın məzmunu haqqında məlumat, üzərində maşın əməliyyatı aparılacaq başlanğıc verilənlərin yerləşdiyi yer - ünvan, nəticənin ünvanı və bu əmrdən sonra yerinə yetiriləcək əmr haqqında məlumat verilməlidir.

İkinci nəsl EHM-lərin yaranması, konkret maşının yox, qoyulmuş məsələnin xüsusiyyətlərindən asılı olan dillərə ehtiyac yaratdı. Bu dillərə formal alqoritmik və ya sadəcə alqoritmik dillər deyilir. Bu dillər üzərinə bir çox şərtlər qoyulur. Belə ki, birincisi bu dillər əyani olmalı, bunun üçün isə burada məlum riyazi simvolika və digər asan başa düşüləcək təsviri vasitələrdən istifadə olunmalıdır, ikincisi bu dillərdə ixtiyari alqoritm asanlıqla ifadə edilə bilməlidir, üçüncüsü, bu dillərdə qurulan alqoritm birqiymətli qəbul edilməli və digər məna daşımamalıdır, dördüncüsü, qurulan mürəkkəb alqoritm daha sadə alqoritmlərin vəhdəti şəklində ifadə edilə bilməlidir. Bunlardan əlavə bu dillər insanla maşın arasında ünsiyyət yaratmalıdır, belə ki, alqoritmik dildən maşın dilinə tərcümə EHM tərəfindən xüsusi proqram- translyator vasitəsilə yerinə yetirilir. Proqram dedikdə isə biz, EHM-ə verilən göstərişləri ifadə edən operatorlar yığını başa düşürük. Birinci və kifayət qədər yaxşı alınmış dil 1954-cü ildə IBM firmasının yaratdığı FORTRAN dili idi. Bu dilin adı FORTRAN-FORmulae TRANslation - formulaların tərcüməsi sözündən götürülmüşdür. Bu dil çox sadə struktura malik olduğundan ondan hal-hazırki vaxta qədər istifadə olunur. Fortranda proqram operatorlar ardıcıllığı şəklində yazılır. Bu dildə yazılan proqram bir və ya bir neçə seqmentlərdən (alt proqramlar) ibarət olur. Bütün proqramın işini idarə edən seqment əsas proqram adlanır.

Fortran dili elmi və mühəndis - texniki hesablama sahələrində istifadə edilmək üçün nəzərdə tutulmuşdu. Lakin bu dildə budaqlanan strukturlu məsələlər (istehsal prosesinin modelləşdirilməsi və s.), bəzi iqtisadi məsələlər və redaktə etmə məsələləri (cədvəl, arayış və s. qurulması) üçün proqramlar da qurula bilər. Sonrakı illərdə bu dilin müxtəlif modifikasiyaları yaradılmışdı.

Fortran dilinin əsasında 1966-cı ildə Dartmut kollecinin hesablama mərkəzində Beyzik dili (BASIC- Beginner's All-purpose Symbolic Instruction Code - yeni başlayanlar üçün çoxməqsədli simvolik əmrlər dili) və 1975-ci ildə Digital Equipment Corporation firması tərəfindən Beyzik- plyus (BASIC - PLUS) (Beyzik dilinin genişlənməsi) dili yaradıldı. Hal-hazırda da bu dillər proqramlaşdırma praktikasında alqoritmik dillərdən istifadə etmək vərdişlərinin alınması üçün ən yaxşı dillərdən biridir.

1960-cı ildə Alqol-60 (ALGOritmic Language - alqoritmik dil) dili yaradılmışdı. Fortran, Alqol-60 dilləri əsasən elmi texniki məsələlərin həlli üçün nəzərdə tutulmuşdu və bu dillər heç də bütün mümkün məsələlər üçün yaramırdı. Buna görə də intensiv inkişaf edən elm və texnikanın yeni sahələrinin tələbatını ödəmək üçün yeni proqramlaşdırma dilləri yaradılır. Məsələn, iqtisadi məsələlərin həlli üçün 1959-cu ildə IBM firması tərəfindən Cobol (Common Business Oriented Language) dili yaradılır. Simvol informasiyanın emalını təmin edən dillərdən biri, Massaçuset texnoloji institutunda 1960-cı ildə yaradılmış Lisp dilidir. Digər dil Snobol - isə təbii dildə yazılmış mətnlərin maşın analizi üçün tətbiq olunur.

Üçüncü nəsl EHM-in yaranması, universal alqoritmik dillərin yaradılması məsələsini qarşıya qoydu. Bu cür dillərin yaradılması üçün edilən cəhdlərdən biri nəticəsində IBM firması tərəfindən PL/1 (Programming Language/1-proqramlaşdırma dili-1) dili yaradılır. Bu dil Fortran, Alqol və Cobol dillərinin əsasında yaradılmış və bu dillərin üstünlüklərini özündə birləşdirmişdi.

1971-ci ildə isə dünyada ilk dəfə olaraq ədədləri cəmləməyə imkan verən avtomatik qurğu yaratmış XVII əsr böyük fransız alimi Paskalın şərəfinə adlandırılmış Paskal dili yaradılır. Bu dil Alqol və PL/1 dillərinin varisidir. Paskal dili də Beysik dili kimi çox sadədir, lakin Paskal dili müasir proqramlaşdırma texnologiyasının geniş tətbiqini təmin edir. Bu dil struktur proqramlaşdırma ideyasının, yəni proqramın kiçik, dəqiq təyin edilmiş prosedurlardan tədricən qurulması ideyasının həyata keçirilməsinə imkan yaradır. 70-ci illərin sonunda Ada dili yaradılır. Bu dil çox böyük tətbiq sahələrinə malik-dir. Dil birinci proqramçı qadın Ada Lavlaysın şərəfinə adlan-dırılmışdır. 80-cı illərin

əvvəllərində C/C++ proqramlaşdırma dili (dilin adı bir C latın hərfindən ibarətdir) yaradılmışdır. C/C++ dili - universal dil olub, geniş yayılmış UNIX əməliyyat sistemi ilə sıx bağlıdır, belə ki, UNIX sistemi və onun proqram təminatı C/C++ dilində yazılıb. C/C++ dili, müasir kompüterlərin imkanlarından tam istifadə etməyə imkan verən dildir. Qeyd etdiyimiz dillərdən başqa digər dillər də mövcuddur və dillərin yaradılması prosesi davam etdirilir.

IV FƏSİL

BASIC (QBASIC) ALQORİTMİK DİLİ.

4.1. Dilin əlifbası. Əsas konstruksiyaları. Verilənlər

Dilin ixtiyari konstruktiv elementini yazmaq üçün müəyyən hərflər, rəqəmlər və işarələr ardıcıllığından - dilin əlifbasından istifadə olunur. Dilin əlifbasına aşağıdakılar aiddir:

1) latın əlifbasının (**A - Z**) hərfləri

2) ərəb rəqəmləri (**0,1,...,9**)

3) hesab əməl işarələri: «**+**» – toplama, «**-**» – çıxma, «*****» – vurma, «**/**» – bölmə, «****» – tam bölmə, «**^**» – qüvvətə yüksəltmə, «**MOD**» – modul üzrə bölmə (qalıq həddin təyini).

4) münasibət əməl işarələri: «**=**» – bərabər, «**<**» – kiçik, «**>**» – böyük, «**>=**» – böyük və ya bərabər, «**<=**» – kiçik və ya bərabər, «**<>**» – fərqli.

5) ayırıcılır və digər işarələr: «**.**» – nöqtə, «**,**» – (vergül), «**;**» – nöqtəli vergül, «**:**» – iki nöqtə, «**'**» – apostrof, «**( )**» – yumru mötərizə, «**"**» – dırnaqlar, «**%**» – tam tipli kəmiyyət nişanı, «**?**» – sual işarəsi, «**!**» – həqiqi tipli kəmiyyət nişanı, «**#**» – ikiqat dəqiqilikli kəmiyyət nişanı, «**$**» – mətn tipli kəmiyyət nişanı, «**&**» – uzun tam tipli kəmiyyət nişanı.

Bundan əlavə dilin əmr və operatorları kimi ingilis dilinin aşağıdakı sözlərindən və söz qısaldılmalarından da istifadə olunur: **AUTO, APPEND, CHAIN, CLOSE, COLOR, COMMON, CIRCLE, DELETE, DATA, DEF FN, DIM, END, EDIT, ELSE, FOR, FILE, GOSUB, GOTO, IF, INPUT, INSTR, LET, LINE, LIST, LOCATE, LEFT, LEN, MID, MERGE, NEXT, NAME, NEW, NONAME, ON, OLD, OPEN, PRINT, PRINT USING, POS, READ, REM, RESTORE, RETURN, RUN, RENAME, RND, RIGHT, SAVE, STEP, SCREEN, STOP, SCR, SEG, SQR, STR, SYSTEM, THEN, TAB, TRM, VAL, WEND, WHILE, WIDTH.**

Verilənlər. Verilənlər iki yerə bölünür: sabitlər və dəyişənlər. Verilənlərin tipi, onların EHM-in yaddaşındakı ifadə forması ilə

təyin edilir. Verilənlərin aşağıdakı tipləri mövcuddur: mətn və ya simvol, tam, kəsr hissəyə malik - birqat və ikiqat dəqiqlikli verilənlər.

Sabitlər. Sabitlər - proqramın yerinə yetiriləcəyi müddətdə dəyişməz qalan qiymətlərdir. Sabitlər – mətn və ya simvol sabitlərə və ədədi sabitlərə bölünür.

Mətn sabit və ya sadəcə sətir bir-birinin ardınca gələn və hər iki tərəfdən dırnaq arasına alınmış simvollar ardıcıllığıdır. Məsələn, «**BAKI**», «**BASIC**» və s. Sətir sabit dilin əlifbasının ixtiyari simvollarından ibarət ola bilər, lakin sətrin uzunluğu 255 simvoldan çox olmamalıdır (QBASIC-də 32567 simvoldan çox olmamalı). Ən qısa sətir heç bir simvola malik olmayan sətirdir və boş və ya sıfır uzunluqlu sətir adlanır.

Onluq ədədlər üç ifadə formasına malikdir: tam sabitlər, adi dəqiqlikli sabitlər və ikiqat dəqiqlikli sabitlər. Tam sabitlər, qarşısında + və ya –işarəsi qoyulan rəqəmlər ardıcıllığı kimi yazılan ədədlərdir (müsbət ədəddə + işarəsi verilməyə də bilər). Tam ədəd -32768-dən +32767-dək dəyişə bilər və yaddaşda iki bayt yer tutur. QBASIC-də uzun tam sabitdən də istifadə olunur. Onun diapazonu -2147483647-dən 2147483648-dək dəyişir. Məsələn, 4568884872&. Tam ədəd qeyd olunmuş intervaldan kənara çıxdıqda artıq tam tipə aid olmur.

Adi dəqiqlikli sabitlər iki formada - qeyd olunmuş və sürüşən onluq nöqtə ilə ifadə olunur. Birinci halda sabit tam və kəsr hissədən ibarət olur, tam hissə, kəsr hissədən onluq vergül deyil, onluq nöqtə ilə ayrılır və bu müsbət və ya mənfi ədəddə rəqəmlərin ümumi sayı yeddidən çox deyil. Məsələn: 345.789; +1234.56; -1.23456. Adi dəqiqlikli sabit sürüşən nöqtə ilə və ya eksponensial formada ifadə edilərkən, sabit tam və ya kəsr ədəd olan mantissadan, E hərfindən və tərtibdən ibarət olur. Tərtib bir və ya iki rəqəmli tam ədəd olub, mantissadakı onluq nöqtənin neçə mərtəbə sağa (işarə müsbət olduqda) və neçə mərtəbə sola (işarə mənfi olduqda) hərəkət etdirmək lazım olduğunu göstərir. Belə ki, bu halda ifadə olunan kəmiyyə-tin faktiki qiymətini almaq olur. Məsələn: -1.345678E+02; 17E-1; +22.2E5; -12E+20.

İkiqat dəqiqlikli ədədi sabitlər adi dəqiqlikli sabitlərdən yalnız onunla fərqlənirlər ki, bu ədədlərdə 16 rəqəm iştirak edə bilər. Sürüşən nöqtə ilə ifadəsində E hərfi əvəzinə D hərfindən

istifadə olunur. Məsələn: 123456789.0987654; -1234567.89D+15.

Adi dəqiqlikli ədədlər EHM-in yaddaşında 4 bayt, ikiqat dəqiqlikli ədədlər isə 8 bayt yer tutur. Sürüşən onluq nöqtə ilə verilən sabitlər EHM-in xüsusiyyətlərindən asılı olur, adətən ən böyük mümkün ədəd 10^{38}-dən böyük olmamalı, ən kiçik isə 10^{-38}-dən kiçik olmamalıdır.

Dəyişənlər. Verilənlərin hər bir tipinə - sətir, tam, adi və ikiqat dəqiqliyə müəyyən dəyişən tipi uyğun gəlir. Dəyişən – ona adı (indentifikatoru) üzrə müraciət olunan və müxtəlif qiymətlər ala bilən kəmiyyətdir. Dəyişən adında birinci simvol latın əlifbasının ixtiyari hərfi, sonrakı simvollar isə hərflər, rəqəmlər və ya nöqtə ola bilər. Dəyişənin adı proqramın bir sətrində yerləşmək şərti ilə ixtiyari uzunluqlu ola bilər, lakin EHM dəyişənin adındakı birinci qırx simvolu qəbul edib, digərlərini atacaqdır. Dəyişənin adı kimi dilin əmr, operator və funksiyalarını ifadə edən sözlərdən istifadə etmək olmaz.

Dəyişənin tipi onun adının axırıncı simvolu ilə müəyyən edilə bilər. Belə ki, bu cür simvollar kimi aşağıdakılardan istifadə olunur: **%** - tam tip üçün; **!** - adi dəqiqliq üçün, **#** - ikiqat dəqiqlik üçün, **$** - sətir tip üçün və **&** - uzun tam tip üçün (QBASIC-də). Məsələn: **A%, B!, C#, D$, E&** uyğun olaraq tam tipli, adi və ikiqat dəqiqlikli sətir tipli və uzun tam tipli dəyişənlərin adını ifadə edirlər.

Hər bir verilmiş anda, yalnız bir qiymət alan dəyişənlərə sadə və ya skalyar dəyişənlər deyilir. Yuxarda nəzərdən keçirdiyimiz dəyişən adları – sadə dəyişənlərin adlarıdır. Eyni zamanda bir dəyişən adı ilə massivdə birləşdirilmiş kəmiyyətlərin müəyyən çoxluğu da işarə edilə bilər. Massivin hər bir ünsürü ədədi indekslərin köməyi ilə işarələnir. Beləliklə, massivin ünsürü - indeksli dəyişəndir. İndeksli dəyişəndəki indekslərin sayı, massivin ölçüsünü təyin edir. Belə ki, bir indeks, bir ölçülü massivi - vektoru, iki indeks iki ölçülü massivi - matrisi təyin edir. Daha böyük ölçülü massivlər üçün isə sadə həndəsi obraz tapmaq çətinlik törədir. Ümumiyyətlə isə bir massiv üçün indekslərin maksimal sayı 255-ə bərabərdir.

İndeksli dəyişənlərdə indekslər massivin adından sonra yumru mötərizədə bir-birindən vergüllə ayrılmaqla verilir. Məsələn: **A(5), B(2,3), C(10)** və s. Eyni bir massivin bütün

ünsürləri eyni bir tipə aid olur: tam, adi və ikiqat dəqiqlikli və ya sətir. Qeyd edək ki, Beyzik dilinin bir çox versiyalarında yalnız bir və iki ölçülü massivlər təyin edilir.

4.2. Əməllər, ifadələr, standart funksiyalar

Verilənlər üzərində müxtəlif əməllər aparıla bilər. Ədədi verilənlər üzərində aparılan əməllərə ədədi əməllər deyilir. Hər bir əməl əməl işarəsi ilə ifadə olunur. Əməl işarələri ilə bir sətirdə birləşdirilən və burada göstərilən əməl yerinə yetirildikdən sonra nəticəsi konkret bir ədədi qiymətə bərabər olan ifadələrə - ədədi ifadələr deyilir. İfadələrdə hesabi əməllərdən, münasibət əməllərindən və məntiqi əməllərdən istifadə olunur.

Hesabi əməllər. Hesabi əməllərə aşağıdakı əməl işarələri təyin edilən əməllər aiddir: ^(qüvvətə yüksəltmə), - (işarəni dəyişmə), *(vurma), /(bölmə), MOD(cəbri modul), +(toplama), - (çıxma). Burada işarəni dəyişmə əməlindən başqa qalan əməllərdə ifadədə iki kəmiyyət iştirak edir. Məsələn: `X^2, -Y, a*b, c/d, 2.5, X` və s. Axırıncı iki misal ifadənin xüsusi halıdır, burada yalnız bir kəmiyyət - sabit və ya dəyişən iştirak edir.

Əməllər hesabi ifadədə müəyyən ardıcılıqla yerinə yetirilir. Belə ki, yuxarda qeyd etdiyimiz hesabi əməllər onların yerinə yetirilməsindəki üstünlüyünün azalması boyunca düzülmüşdür, yəni ifadədə birinci növbədə qüvvətə yüksəltmə və s. əməllər yerinə yetirilir. Qeyd edək ki, bu sırada vurma və bölmə, çıxma və toplama əməlləri yerinə yetirilmədə eyni üstünlüyə malikdir. Eyni üstünlüyə malik əməllər ifadədə soldan sağa doğru ardıcıl yerinə yetirilir. Qüvvətə yüksəltmə əməlində isə bu ardıcıllıq sağdan sola doğru aparılır. Əməllərin standart yerinə yetirilmə ardıcıllığı ifadəyə mötərizələr daxil edilməklə dəyişdirilə bilər. Belə ki, ifadədə birinci növbədə mötərizə daxilindəki əməllər yerinə yetirilir. Öz növbəsində mötərizələr bir- birinin daxilində verilə bilər, bu halda birinci növbədə ən daxildə olan mötərizədəki əməliyyat yerinə yetirilir.

Qeyd edək ki, qüvvətə yüksəltmə əməli, əsasın öz-özünə qüvvətdə göstərildiyi qədər hasili kimi tapılır. Əgər qüvvət tam ədəd deyilsə, onda qüvvətə yüksəltmə əməli `X^Y=EXP(Y*LOG(X))` düsturu ilə aparılır, burada `LOG(X)` – `X`-

in natural loqarifmidir.

Cəbri modul əməliyyatı, bölmədən sonra qalan qalıq həddi hesablayır. Məsələn, **10 MOD 3** qalıq həddi 1 və ya **31.82 MOD 6.4** – 32/6 kimi yerinə yetirilir, nəticədə qalıq həddi 2-yə bərabərdir.

Münasibət əmələri. Beyzik dilində aşağıdakı altı münasibət əməli təyin edilib: **=** (bərabərdir), **<>** (bərabər deyil), **<** (kiçikdir), **>** (böyükdür), **<=** (kiçik bərabərdir) və **>=** (böyük bərabərdir). Bu əmələr eyni yerinə yetirilmə üstünlüyünə malikdir, yəni bir ifadədə bir neçə münasibət əməli varsa, onlar soldan sağa doğru yerinə yetirilir (mötərizələrdən istifadə edilmirsə). Məsələn, **X>Y**, **X=Y**, **X<Y**, **X>=Y** və s.

Münasibət əməlində aparılan müqayisənin nəticəsi, bu müqayisənin nəticəsinin doğru olub-olmaması sualına cavab olur, yəni məsələn, **X=Y** ifadəsi **X**-in **Y**-ə bərabər olub-olmamasını soruşur. Əgər sualın cavabı «hə»-dirsə, onda ifadənin nəticəsi 1 qiymətinə bərabər olur və bu cavab «doğru» kimi qəbul olunur, sualın cavabı «yox» olarsa, onda ifadənin nəticəsi 0 qiymətinə bərabər olur və bu cavab «yalan» kimi qəbul olunur.

Məntiqi əmələr. Bu əmələr yalnız «doğru» və «yalan» qiymətləri üçün mənaya malikdir. Bu qiymətlər isə münasibət əmələrinin nəticəsi olduğundan, məntiqi əmələr münasibət əmələri iştirak edən ifadələr üzərində aparılır. Məntiqi əməlin nəticəsi həmişə ya «doğru», ya da «yalan» qiymətləri olur.

Beyzik dilində aşağıdakı altı məntiqi əməl nəzərdə tutulub: **NOT** (məntiqi inkar), **AND** (konyunksiya və ya məntiqi vurma), **OR** (dizyunksiya və ya məntiqi toplama), **XOR** (dizyunksiyanın istisna edilməsi), **EQV** (ekvivalentlik), **IMP** (implikasiya). Bu əmələr birinci növbədə yerinə yetirilmə üstünlüyünün azalması boyunca düzülübdür, yəni ifadədə birinci növbədə **NOT**, sonra **AND** və s. əmələr yerinə yetirilir. Bu məntiqi əmələrin yerinə yetirilməsi nəticəsində alına biləcək cavabların mümkün variantları aşağıdakı cədvəldə gətirilir:

X	Y	NOT X	X AND Y	X OR Y	X XOR Y	X EQV Y	X IMP Y
doğru	doğru	yalan	doğru	doğru	yalan	doğru	doğru
doğru	yalan	yalan	yalan	doğru	doğru	yalan	yalan
yalan	doğru	doğru	yalan	doğru	doğru	yalan	doğru
yalan	yalan	doğru	yalan	yalan	yalan	doğru	doğru

Sətir və ya mətn əməllər. Sətri dəyişənlər + işarəsi ilə ifadə olunan birləşdirmə (konkatenasiya) əməli vasitəsilə sətri və ya simvol ifadə adlanan ifadələrdə birləşdirilə bilər. Bu əməl sətri qiymətləri bir-birinin ardınca birləşdirərək daha böyük bir sətri qiymət əmələ gətirir, burada əməldə solda duran sətri qiymətin axırıncı simvolunun ardınca sağdakı sətri qiymətin birinci simvolu gələcəkdir. Nəticədə, alınan sətrin uzunluğu, bu birləşmədə iştirak edən sətri qiymətlərin uzunluqları cəminə bərabər olacaqdır. Məsələn, əgər **A$** dəyişəni «**Baki-**» qiymətinə, **B$** dəyişəni isə «**Azerbaycan**» qiymətinə malikdirsə, onda **A$+B$** ifadəsinin qiyməti «**Baki-Azerbaycan**» simvolları sətri olacaqdır.

Standart funksiyalar. Beyzik dilində bəzi çox işlənən riyazi funksiyalar və simvollar sətri üzərində aparılan bəzi əməllər, standart funksiyalar şəklində ifadə olunmuşdur. Standart funksiya, onun identifikatoru və ya adı olan müəyyən sözlə başlayır, ondan sonra isə adətən yumru mötərizə daxilində funksiyanın arqumenti gətirilir.

Riyazi funksiyalar aşağıdakı cədvəldə gətirilir:

Riyazi ifadəsi	Beyzikdə ifadəsi	Riyazi ifadəsi	Beyzikdə ifadəsi
$\sin x$	SIN(X)	arqument x-i aşmayan ən böyük tam ədəd	INT(X)
$\cos x$	COS(X)		
$\operatorname{arctg} x$	ATN(X)		
$\ln x$	LOG(X)	təsadüfi ədədlərin seçilməsi	RND
$\lvert x \rvert$	ABS(X)	onluq nöqtədən sonra bütün rəqəmlərin atılması	FIX(X)
$\sqrt{x}$	SQR(X)		
e^x	EXP(X)		

Bu funksiyalardan başqa **CINT(X)** və **CDBL(X)** funksiyaları da var ki, bunlar uyğun olaraq arqumentdəki ədədi ən yaxın tam ədədə qədər yuvarlaqlaşdırır və arqumentin qiymətini ikiqat dəqiqlikli qiymətə çevirir.

Bu funksiyalardan bəziləri üçün misal gətirək. Əgər **X**, 88.65-ə bərabərdirsə, onda **FIX(X)** 88-ə, **INT(X)** də 88-ə, **CINT(X)** isə 89 və nəhayət, **CDBL(X)** 88.65000000000000 bərabər olacaqdır.

Rəqəmlərdən ibarət sətri ədədi qiymətə və əksinə çevirmək üçün **VAL(X)** və **STR$(X)** standart funksiyalarından istifadə olunur. Belə ki, birinci funksiya rəqəmlərdən ibarət sətri ədədi qiymətə çevirir, məsələn, **VAL("-123")** funksiyasının qiyməti **-123** olacaq. İkinci funksiya isə əksinə ədədi qiyməti rəqəmlərdən ibarət sətirə çevirir, məsələn **STR$(123)** funksiyasının nəticəsi "123" sətri olacaq.

4.3. Operatorlar sistemi

Beyzik dilində ixtiyarı proqram operatorlardan təşkil olunur. Operator sətrin nömrəsi ilə başlayır, bundan sonra operatoru ifadə edən söz və operatorun təyinatından asılı olan müxtəlif elementlərdən ibarət siyahı gəlir. Proqramın bir operatorunun (əmrinin) tutduğu sətrində maksimal mümkün simvolların sayı 255-ə bərabərdir. Proqramın hər sətrinin sonunda klaviaturada Enter düyməsi basılır və bununla sətir EHM tərəfindən qəbul edilir. Bir sətirdə bir-birinin ardınca bir neçə operator da verilə bilər, bu halda operatorlar bir-birindən iki nöqtə ilə ayrılır. Burada *Enter* düyməsi sətrin sonunda bir dəfə basılır və bu halda sətrin yalnız birinci operatoru sətir nömrəsinə malik olur.

Proqram, onu təşkil edən operatorların sətir nömrələrinin artımı boyunca yerinə yetirilir. Proqramlaşdırma praktikasında adətən proqramı, son operatoru olan **END** operatoru ilə qurtarırlar. Lakin bu heç də məcburi deyildir. Əgər proqramda alt proqramdan istifadə olunursa, onda əsas proqramın sonunda **END** operatorunu qoymaq vacibdir, çünki bununla əsas proqram alt proqramdan ayrılır, və əsas proqram yerinə yetirildikdən sonra alt proqrama müraciət olmadan o ardıcıl yerinə yetirilmir.

Özünün funksional təyinatına görə Beyzik dilinin operatorları aşağıdakı siniflərə bölünür: təyinetmə və təsviretmə operatorları, mənimsətmə və daxiletmə operatorları, proqramın struktur və alt proqram operatorları, idarəetmə operatorları, çıxış və ya çap operatorları, faylların emalı operatorları, maşın qrafikası operatorları.

Proqrama izahedici mətnləri və qeydləri daxil etmək üçün izah etmə və ya şərh etmə operatorundan istifadə olunur. Bu operator REM sözü və onun ardınca ixtiyari mətn verilməklə ifadə

olunur. *Məsələn*:

10 REM *proqramın əvvəli*
70 REM *tənliyin köklərinin hesablanması*

Beyzik dilində sətirlər adətən 10-dan başlayaraq 10 addımı ilə nömrələnir. Lakin proqramda sətirlər ixtiyari başqa cür də nömrələnə bilər. Sətrin nömrələri arasında 10 addımının verilməsi isə buraxılmış sətirlərin proqrama daxil edilməsi işini yüngülləşdirmək üçün edilir. QBASIC-də sətirlərin nömrələnməsi vacib deyil, yalnız proqramda ona müraciət nəzərdə tutulmuş sətirlər nömrələnir.

4.4. Dəyişənlərin təsviri. Operator - funksiya

Dəyişənin tipinin təsvirinin əsas üsulu artıq qeyd etdiyimiz dəyişənin adının sonunda axırıncı simvolla təsvir etmədir. Lakin bununla yanaşı dəyişənin tipi, onun adının birinci hərfi tiplərin təsviri operatorunda verilməklə də təyin edilə bilər. Bu halda artıq dəyişənin adının sonunda xüsusi simvollardan (**%**, **!**, **#**, **$**) istifadə edilmir. Beyzik dilində təyin olunan tiplərə uyğun olaraq aşağıdakı dörd cür tiplərin təsviri operatoru mövcuddur:

DEFINT *hərflər siyahısı*
DEFSNG *hərflər siyahısı*
DEFDBL *hərflər siyahısı*
DEFSTR *hərflər siyahısı*

Bu operatorlar uyğun olaraq tam, adi dəqiqlikli, ikiqat dəqiqlikli və sətir tipli dəyişənlərin tiplərinin təsviri üçün nəzərdə tutulub. *Məsələn:*

```
10 DEF A,G-L
20 DEF C,D,E
30 DEF M-S
40 DEF B,X,Y,Z
```

burada **A,G,H,I,J,K,L** hərfləri ilə başlayan dəyişənlər tam tipli, **C,D,E** ilə başlayan adi dəqiqlikli, **M,N,O,P,Q,R,S** ilə başlayanlar ikiqat dəqiqlikli və nəhayət **B,X,Y,Z** ilə başlayanlar sətir tipli dəyişən kimi qəbul edilir.

Operator-funksiya. Bu operatorun ümumi şəkli aşağıdakı

kimidir:

DEF FNS(X)=E

burada **S** – funksiyanın adı, **FN** hərfləri funksiya adının (**FNS**) tərkib hissəsidir, **X**-formal parametrlərin siyahısıdır, yəni **S** - funksiyasının yerinə yetiriləcəyi anda təyin ediləcək faktiki qiymətlər üçün EHM-in yaddaşında yer ayıran fiktiv dəyişənlərin adlarıdır, **E** - verilmiş funksiyanın hansı əməlləri yerinə yetirəcəyini təyin edən ifadədir.

Operator-funksiyanın yerinə yetirilməsi, ona müraciət olduğu təqdirdə həyata keçirilir. Müraciət isə **FNS(A)** formasında yerinə yetirilir, harada **FNS** funksiyanın adı, **A** isə faktiki parametrlər siyahısıdır. Bu müraciət yerinə yetirilərkən faktiki parametrlər uyğun formal parametrlərin yerinə qoyulur və E ifadəsi bu qiymətlər üçün hesablanır. Qeyd edək ki, müraciət proqramın ixtiyari bir konstruksiyası daxilindən verilə bilər. Formal və faktiki parametrlər arasında say və tip uyğunluğu olmalıdır. Faktiki parametrlər cəbri ifadələr kimi də verilə bilər. *Məsələn*:

```
10 DEF FNS(X)=2*X-A^2
20 A=7:B=2
30 P=FNS(15)
40 L=FNS(3)+FNS(7)
50 K=FNS(B+2)-FNS(B+8)
```

4.5. Mənimsətmə operatoru. Daxil etmə operatorları

Mənimsətmə operatorunun ümumi görünüşü aşağıdakı kimidir:

LET V=E

burada **LET** sözü verilməyə də bilər, **V** – dəyişənin adı, **E** – isə ifadədir. *Məsələn*:

```
10 LET X=7.2
20 K%=15
30 A$="BASIC"
```

Mənimsətmə operatoru, burada mənimsəmə işarəsi olan bərabərliyin sol tərəfindəki dəyişənə, bu işarədən sağda duran ifadənin qiymətini mənimsədir. Dəyişən və ona mənimsədilən qiymət eyni tipli verilən olmalıdır. Əgər bu şərt ödənmirsə, yəni, məsələn, sətir tipli dəyişənə ədədi qiymət mənimsədilirsə, onda

EHM səhv haqqında məlumat verir.

Adi dəqiqlikli və ya tam olan dəyişənə, ikiqat dəqiqlikli ədədi qiymət mənimsədilirsə, bu qiymət dəyişənin tipinə uyğun yuvarlaqlaşdırılır. Analoji olaraq, əgər tam dəyişənə, adi dəqiqlikli qiymət mənimsədilirsə, onda bu qiymət ən yaxın tam ədədə qədər yuvarlaqlaşdırılır. *Məsələn*:

```
10 A!=1.23456789: B%=1.23456789
```

operatorlarının yerinə yetirilməsi nəticəsində **A!** dəyişəni **1.234567** qiymətini, **B%** isə **1** qiymətini alacaqdır.

Adi və ya ikiqat dəqiqlikli dəyişənlərə tam ədəd mənimsədilirsə, onda tam qiymət onluq nöqtə və onun ardınca sıfırlarla tamamlanır. Analoji olaraq, ikiqat dəqiqlikli dəyişənə adi dəqiqliklə qiymət mənimsədilirsə, çatışmayan rəqəmlər sıfırlarla əvəz olunur, lakin dəyişdirilmiş qiymət yalnız altı dəqiq rəqəmə malik olur. *Məsələn*:

```
10 C!=1:D#=1234567
```

operatorunun yerinə yetirilməsi nəticəsində **C!** dəyişəni **1.000000** qiymətini, **D#** isə **1234567000000000** qiymətini alır.

Dəyişənlərə, qiymətlərin mənimsədilməsi üçün mənimsəmə operatoru ən geniş istifadə olunan operatordur. Lakin bu operator, çoxlu sayda dəyişənlərə qiymətlərini mənimsətmək tələb olunarkən, əlverişli olmur. Bu məqsədlər üçün daha effektiv olan daxiletmə operatorlarından istifadə olunur.

Verilənlər bloku vasitəsilə daxiletmə. READ-DATA daxiletmə operatoru.

Bu operatorun ümumi görüşünü aşağıdakı kimidir:

DATA *sabitlər siyahısı*

burada siyahıda bir-birindən vergüllə ayrılmaq şərtilə, ixtiyari mümkün ədədi və ya sətri sabitlər ola bilər. Bu operator EHM-in yaddaşında verilənlər blokunu formalaşdırır. Burada istifadə olunan sətir sabitlər probellə başlayarsa və ya qurtararsa, eləcədə əgər sətir sabitinə daxil simvollardan biri iki nöqtə və ya vergül olarsa, onda belə sətir sabitləri dırnaq içərisinə alınır. Eləcədə dırnaq arasına alınan sətirdə digər dırnaq işarəsindən istifadə etmək olmaz. *Məsələn*:

```
10 DATA 10,3.14,BASIC,"BAKI",2.5E-2
20 DATA 102.3,-45
```

DATA operatoru proqramın ixtiyari yerində verilə bilər və proqramda bir neçə **DATA** operatoru ola bilər. Bu operatorların sabitlər siyahısı bir verilənlər bloku əmələ gətirir. Bu blok, birinci **DATA** operatorunda (proqramın mətnində birinci gələn) verilmiş sabitlərlə başlayıb, proqramda axırıncı **DATA** operatorunda verilən sabitlərlə qurtarır. Belə ki, yuxarıda verilən misalda bu cür blok 10-la başlayıb, -45-lə qurtarır.

Verilənlər blokundan sabitlərin qiymətləri skalyar dəyişənlərə və massiv elementlərinə aşağıdakı daxil etmə operatoru vasitəsilə mənimsədilir:

READ *dəyişənlərin siyahısı*

burada dəyişənlərin siyahısında qiymətlər mənimsədiləcək dəyişənlər verilir.

READ operatoru yerinə yetirilərkən o, **DATA** operatorlarının formalaşdırdığı verilənlər blokundan birinci qiyməti, verilmiş **READ** operatorunun dəyişənlər siyahısındakı birinci dəyişənə mənimsəyir. **READ** operatorunun ikinci dəyişəninə ikinci qiymət, üçüncüyə üçüncü qiymət və s., o vaxta qədər ki, bu operatorun bütün dəyişənlərinə uyğun qiymətlər mənimsədilsin:

```
10 READ A%, B, C$, D#
20 READ X$, X2%
30 DATA 5,12.2, BASIC
40 DATA 1.2345678, "Bakı ",17
```

Əgər verilənlər blokunda istifadə edilməmiş verilənlər qalarsa, onlar nəzərə alınmır. Lakin, əgər hər hansı bir **READ** operatoru yerinə yetirilərkən məlum olarsa ki, verilənlər blokundakı bütün sabitlərdən istifadə edilib və dəyişənə mənimsədiləcək sabit yoxdur, onda EHM səhf haqqında məlumat verir. **READ** operatorunun dəyişənlər siyahısındakı dəyişənlərin və massiv elementlərinin tipləri onlara mənimsədilən qiymətlərin tipləri ilə eyni olmalıdır. Belə ki, məsələn, dəyişən ədədi ona mənimsədilən qiyməti isə sətri və əksinə ola bilməz. Belə hal qeyd olunduqda isə, səhv barədə məlumat verilir. Əgər dəyişəndə sabit də ədədi olarsa, lakin onların tipləri dəqiqliyinə görə fərqlənirsə, onda **READ** operatoru, **LET** mənimsəmə operatoru üçün yuxarda qeyd etdiyimiz analoji əməliyyatları yerinə yetirir.

READ operatoru yerinə yetirilərkən dəyişənə verilənlər blo-

kundan növbəti qiyməti mənimsədildiyi vaxtda, verilənlər blokundakı cari mövqeyi güdən xüsusi göstəricinin bir addım yerdəyişməsi baş verir. Bu göstəricinin yer dəyişməsinin idarə etmək üçün bərpa etmə operatoru adlanan və sadə halda **RESTORE** görünüşünə malik operatordan istifadə olunur. Bu operator yerinə yetirilərkən göstərici verilənlər blokunda sabitlər siyahısının başlanğıcına qaytarılır və proqramda bu operatordan sonra gələn növbəti **READ** operatoru qiymətləri verilənlər blokundan əvvəldən başlayaraq təkrarən seçəcəkdir. *Məsələn*:

```
10 DATA 5, 10.5, 1.2, 12, 15
20 READ K,A,B,L,M
30 RESTORE
40 READ N,E,F
```

Nəticədə alırıq: **K=5,A=10.5,B=1.2,L=12,M=15,N=5, E=10.5,F=1.2.**

Eyni zamanda **RESTORE** operatoru proqramda olan ixtiyarı **DATA** operatorunun elementlər siyahısını bərpa etməyə imkan verir. Bu halda operatorun ümumi görünüşü **RESTORE** *«sətrin nömrəsi»* kimi olur. Burada sətrin nömrəsi – sabitlər siyahısı bərpa ediləcək **DATA** operatorunun yerləşdiyi sətrin nömrəsidir. *Məsələn*:

```
10 DATA 1, 2, 3, 4
20 DATA 5,6,7
30 READ A,B,C,D,E,F,K
40 RESTORE 20
50 READ X,Y,Z
```

Nəticədə alırıq: **A=1,B=2,C=3,D=4,E=5,F=6,K=7,X=5,Y=6, Z=7** olar.

Verilənlərin klaviaturadan daxil edilməsi. INPUT daxil etmə operatoru. Dəyişənlərin qiymətləri proqrama, tələb olunduğca klaviatura vasitəsilə daxil edilə bilərlər. Bunun üçün daxil etmə operatoru nəzərdə tutulub, həmin operator daxil edilən qiymətləri klaviaturadan qəbul edib, uyğun dəyişənlərə və ya massiv elementlərinə mənimsəyir. Bu operatorun ümumi görünüşü aşağıdakı kimidir: **INPUT** dəyişənlər siyahısı, harada dəyişənlər siyahısında ədədi və sətir dəyişənlərinin adları yerləşir. *Məsələn*:

```
10 INPUT A,B%,C$,D#
```

INPUT operatorunu yerinə yetirərkən, EHM displeyin ekranına sual işarəsi çıxarır. Bu işarənin verilməsi isə o deməkdir ki, istifadəçi **INPUT** operatorunun siyahısındakı hər bir dəyişənin qiymətini klaviaturadan daxil etməlidir. Daxil edilən qiymətlərin sayı və tipi **INPUT** operatorunun siyahısındakı dəyişənlərin sayı və tipinə uyğun gəlməlidir.

Proqramda bir neçə **INPUT** operatoru olduqda onları bir-birindən fərqləndirmək üçün bu operatorda müəyyən şərhedici məlumat vermək olur. Bu məlumatın mətni **INPUT** operatorunda **INPUT** sözündən sonra dırnaq arasında sətir sabiti kimi verilməli və dəyişənin adından nöqtə vergüllə və ya vergüllə ayrılmalıdır. *Məsələn*:

```
10 INPUT «Üçbucağın sahəsi»; S
20 INPUT «Tənliyin əmsalı»; A
```

Nəticədə ekranda üçbucağın sahəsi **?** və tənliyin əmsalı **?** mətnləri və sual işarələri veriləcək. Şərhedici mətndən sonra vergül qoyulduqda isə ekranda mətdən sonra sual işarəsi verilmir.

4.6. Xaricetmə operatorları

Displeyin ekranındakı təsvirin eni, aşağıdakı sətrin uzunluğunu təyin edən operatorla **WIDTH N,W** operatoru ilə proqramlaşdırıla bilər. Burada operatorun siyahısında N - hər hansı qurğunun adı və ya nömrəsini, **W**-isə sətrin simvollarla uzunluğunu bildirir. Operatorda **N** parametri verilməyə də bilər, **W** parametri isə istifadəçinin sətrin uzunluğu üçün verdiyi qiymət ola bilər.

Ekranda 25-ə qədər mətn sətri yerləşdirmək mümkündür. Ekranda verilən informasiyanı silmək üçün **CLS** operatorundan istifadə etmək olar, bu operatoru yığıb, *Enter* düyməsini basdıqdan sonra ekrandakılar silinir və kursor ekranın sol yuxarı ucuna gətirilir.

Dəyişənlərin və massiv elementlərinin qiymətləri, yəni ədədi və sətri sabitlərin ekrana çıxarılışı **PRINT** ***siyahı*** operatoru ilə yerinə yetirilir, burada siyahıda qiymətləri ekrana çıxarılmalı olan dəyişənlərin adları və ya çap ediləcək sətir sabitləri dırnaq işarəsi arasında verilir. Bu siyahıda ifadələr də iştirak edə bilər.

PRINT operatorunda siyahı olmayada bilər, bu halda ekrana boş sətir çıxarılır. Operatorun siyahısında elementlər arasında ayırıcı kimi vergüldən və nöqtə vergüldən istifadə olunur. Birinci halda qiy-mətlər ekrana bir-birindən müəyyən aralıqda çıxarılır, ikinci hal-da isə onlar bilavasitə bir birinin ardınca çıxarılacaq.

Müsbət ədədlər çıxarılarkən qarşısında bir boş yer, mənfi ədədlərin qarşısında isə mənfi işarəsi verilir. Ekranda sətir tamamilə qiymətlərlə doldurulduqda, çıxarış növbəti sətirdən veriləcəkdir. *Məsələn*:

```
10 PRINT "A,B,C amsallarini daxil et:"
20 INPUT A,B,C
```

Nəticədə:

```
A,B,C amsallarini daxil et:
? 1 2 3
```

```
10 A=5: B=3.14: C$= "BASIC"
20 PRINT A,B,C$
```

Nəticədə:

```
5 3.14 BASIC
```

```
10 S=12.5
20 PRINT "Ucbucagin sahesi=";S
```

Nəticədə:

```
Ucbucagin sahesi=12.5
```

PRINT operatorunda siyahıda ayırıcı kimi vergüldən istifadə etdikdə, qiymətlər ekrana sütunlarla standart cədvəl formasında çıxarılır. Belə ki, ekran sətrin 40 simvol eni halında 14 simvoldan ibarət 2 və 12 simvoldan ibarət bir zonaya bölünür. Sətrin eni 80 simvol olduqda isə 14 simvoldan ibarət 5 zonaya və 10 simvoldan ibarət 1 zonaya bölünür. Əgər **PRINT** operatorunda siyahının elementi qarşısında vergül işarəsi dursa, onda bu elementin ekrana çıxarılması zamanı kursor növbəti zonanın əvvəlinə keçəcəkdir və qiyməti oradan çıxaracaqdır. Element qarşısında iki vergülün olması növbəti zonanın buraxıldığını, siyahının sonunda vergülün olması isə növbəti **PRINT** operatorunun siyahısındakı elementlərin həmin sətirdə növbəti zonadan başlayaraq ekrana çıxarılacağını bildirir.

Ədədi qiymətlər ekrana ədəddəki rəqəmlərin sayından asılı

olaraq qeyd olunmuş və ya sürüşən onluq nöqtə ilə çıxarılırlar. Belə ki, adi və ikiqat dəqiqlikli ədədlər ekrana həmişə sürüşən onluq nöqtə ilə çıxarılırlar. Məsələn, 12345678912 və 0.001234567890123456789 qiymətləri ekrana 1.2345678E+10 və 1.234567890123457E-03 kimi, 1234.56789 və 123456789.123456789 qiymətləri isə ekrana 1234.568 və 123456789.1234568 kimi çıxarılacaqlar.

Formatlı çıxarış. Bu operatorun ümumi görünüşü aşağıdakı kimidir:

PRINT USING *«şablon»; siyahı.*

Bu operator informasiyanın displeyin ekranına, xüsusi sətir qiymət - şablonla təyin edilmiş formada (formatda) çıxarılışını təmin edir. Şablonun hər bir simvolu müəyyən məna daşıyır və çıxarılan qiymətin xüsusiyyətlərini təyin edir. Bu operatorun siyahısı **PRINT** operatorunda siyahı üzərinə qoyulan şərtləri ödəyir.

Sətir qiymətlərin çıxarılışı zamanı şablonda aşağıdakı simvollardan (sətir formatdan) istifadə olunur:

! - bir simvolun çıxarılışını bildirir;

**** - qeyd olunmuş sayda simvolların çıxarışını bildirir: iki və daha artıq (xətlər arasında buraxılan boş yerlərin sayından asılı olaraq);

& - sətir qiymətin tamamilə çıxarılmasını bildirir.

Məsələn: əgər **A$** sətir dəyişəni «**BASIC**» qiymətini alırsa, onda

```
100 PRINT USING "!"; A$
110 PRINT USING "\\"; A$
120 PRINT USING "\   _  _\"; A$
130 PRINT USING "&"; A$
```

operatorlarının yerinə yetirilməsi nəticəsində alarıq:

```
B
BA
BASI
BASIC
```

Beləliklə, əgər sətir qiymətin uzunluğu şablonla verilən sahəyə sığmırsa, onda «artıq» qalan sağ simvollar atılır, sətri qiymətin uzunluğu şablonun verdiyi sahədən kiçikdirsə, onda boş

qalan yerlər probellərlə doldurulur.

Ədədi qiymətlərin çıxarışı zamanı şablonda aşağıdakı simvollardan (ədədi formatdan) istifadə olunur:

– çıxarılan qiymətin bir mərtəbəsinin çıxarışı;

+ – qiymət qarşısında + və - işarəsinin verilməsi;

- – qiymət müsbət olduqda qarşısında boş yer, mənfi olduqda – işarəsinin verilməsi.

.- sahənin verilmiş mövqeyində onluq nöqtənin çıxarılması;

^^^^ - qiymətin sürüşən onluq nöqtə ilə verilməsi;

, - hər üç mərtəbədən bir qiymətdə vergülün verilməsi;

**- ayrılmış sahədə soldan boş qalan yerlərin ulduz işarələri ilə doldurulması;

$$ - qiymət qarşısında bir $ simvolunun çıxarılışı;

**$ - soldan boş qalan yerlərin ulduzlarla doldurulması və qiymət qarşısında bir $ simvolunun çıxarılışı. *Məsələn*:

```
10 A=123: PRINT USING "###"; A
     123
20 B= -456: PRINT USING "#####"; B
     -456
30 C=78901: PRINT USING "##"; C
    %78901
```

Şablonda + işarəsi həm birinci # simvolundan əvvəl, həm də axırıncı # simvolundan sonra verilə bilər, - işarəsi isə həmişə axarıncı # simvolundan sonra verilir. Şablondakı onluq nöqtə əvvəlcədən ədəddə tam və kəsr hissəyə ayrılacaq yerlərin sayını təyin edir. Şablonda qiyməti sürüşən onluq nöqtə ilə vermək üçün istifadə olunan ^ simvolları ədədi şablonun axırında gətirilir. Böyük ədədlərin oxunmasını sadələşdirmək üçün çox vaxt hər üç rəqəmdən sonra vergül işarəsinin qoyulması əlverişli olur, məsələn: 123,456, 789, 912. Vergül işarəsi şablonda onluq nöqtəyə qədər ixtiyari # işarəsindən sonra qoyula bilər. *Məsələn*:

```
40 A=-32.17:B=12.329
45 PRINT USING "+##.##";A;B
-32.17    +12.33
```

```
50 PRINT USING "##.##+";A;B
12.33+
60 PRINT USING "##.##-";A;B
12.33
70 x=-0.12345:PRINT USING "###.##^^^^";x
-12.35E-02
80 y=12345E7
90 PRINT USING "#,###########.##";y
12,345,000,000.00
100 z=123.45:PRINT USING "**######.##";z
**** 123.45
110 PRINT USING "$$###.##";z
_ $ 123.45
120 PRINT USING "*$##.##";z
*$123.45
```

Kağız üzərinə çıxarış. Verilənlərin çap qurğusu vasitəsilə kağız üzərində çap edilməsi üçün proqramda **LPRINT** və **LPRINT USING** operatorlarından istifadə etmək lazımdır. Bu operatorlar artıq nəzərdən keçirdiyimiz **PRINT** və **PRINT USING** operatorlarına uyğundur və onların siyahıları eyni struktura malikdir.

PRINT operatorunda TAB funksiyası. Bu funksiya ekran sətrinin verdiyimiz mövqeyində qiymət çıxarmağa imkan verir. Məsələn,

PRINT TAB(40);A

operatoru ekran sətrinin 40-cı mövqeyində **A**-nın qiymətini çıxaracaqdır. Funksiyanın ümumi şəkli **TAB(N)** formasındadır, burada **N** - tam ədəd və ya tam tipli hesabi ifadədir. **PRINT** operatoru ilə bu funksiyanın ümumi şəkli **PRINT TAB(N);X** formasındadır, bu operator **X**-in qiymətini sətrin **N**-ci mövqeyinə çıxarır. Bu funksiyanın köməyilə ekrana müxtəlif qrafiklər, cədvəllər, histoq-ramlar və s. çıxarmaq olur.

4.7. Keçid operatorları

Praktiki olaraq bütün proqramlarda operatorların sətirlərin sıra nömrəsinin artımı boyunca təbii yerinə yetirilməsi bu və ya digər səbəblərə görə dəyişdirilir. Bu ona görə edilir ki, bəzi opera-

torlar təkrarən yerinə yetirilməli və ya yerinə yetirilmədən buraxılmalıdırlar.

Şərtsiz keçid operatoru. GOTO operatoru. Operatorların yerinə yetirilmə ardıcıllığının dəyişdirilməsi və idarəetmənin proqramının ixtiyari əvvəlcədən təyin edilmiş sətrinə verilməsi üçün aşağıdakı şərtsiz keçid operatorundan istifadə olunur:

GOTO N ,

burada **N**- keçid yerinə yetiriləcək sətrin nömrəsidir. Sətrin **N**-nömrəsi operatorda ədədi sabit şəklində verilir. Əgər bu ədədi sabitdə onluq nöqtə və kəsr hissə varsa, onlar nəzərə alınmır. Sətrin nömrəsi sürüşən onluq nöqtə formasında verilə bilməz. *Misal*:

```
10 INPUT "Dairənin radiusu"; R
20 PRINT "R=";R
30 S=3.1416*R^2
40 PRINT "Dairənin sahəsi"; S
50 GOTO 10
```

Hesablanan keçid operatoru. ON – GOTO operatoru. Əgər əvvəlcədən məlumdur ki, proqram yerinə yetirilən zaman yaranmış vəziyyətdən asılı olaraq, idarəetmə proqramın müxtəlif hissələrinə verilməlidir, onda aşağıdakı hesablanan keçid operatorundan istifadə olunur:

ON E GOTO *sətir nömrələrinin siyahısı*

burada **E** - ədədi ifadədir, siyahı isə (bir-birindən vergüllə ayrılan) proqramın ixtiyari sayda sətirlərinin nömrələrindən ibarətdir. Bu operator yerinə yetirilərkən **E** ədədi ifadəsi hesablanır və idarəetmə operatorun siyahısındakı sətir nömrələrindən qiyməti **E** ifadəsinin cari qiymətinə bərabər olan sıra nömrəli sətirə verilir. Beləliklə, **E** ifadəsinin ala biləcəyi qiymətlər vahidlə, operatorun siyahısındakı sətir nömrələrinin sayına bərabər olan ədəd arasında olmalıdır. E ifadəsinin kəsr hissəyə malik qiymətləri, ən yaxın tam ədədə qədər yuvarlaqlaşdırılır. Əgər **E** ifadəsinin aldığı qiymət sıfır və ya siyahıdakı elementlərin sayından böyük olarsa, onda bu operator yerinə yetirilmir və idarəetmə onun ardınca gələn növbəti operatora verilir və nəhayət **E** ifadəsi mənfi qiymət alarsa, EHM səhf olduğu barədə məlumat verəcəkdir. *Misal*:

```
10 PRINT "Asagidaki fiqurlarin sahelerini
```

```
    hesablamaq ucun daxil edin:"
20 PRINT "Kvadrat-1, Daire-2, Ucbucaq-3"
30 INPUT K
40 INPUT X
50 ON K GOTO 60,70,80
60 PRINT "Kvadratin sahesi"; X^2: GOTO 90
70 PRINT "Dairenin sahesi";3.14*X^2:GOTO 90
80 PRINT "Ucbucagin sahesi"; X^2*SQR (3)/4
90 END
```

Şərti keçid operatoru. Şərti keçid operatorlarında, bu və ya digər əməliyyatların yerinə yetirilməsini təyin edən şərt, bilavasitə operatorun konstruksiyasına daxildir. Beyzik dilində iki cür şərti keçid operatoru mövcuddur; şərt operatoru (**IF - THEN** operatoru) və tam şərt operatoru (**IF - THEN - ELSE** operatoru).

Şərt operatoru. Hər hansı bir əməliyyatın, yalnız operatorda verilən şərt ödəndikdə yerinə yetirilməsinə göstəriş verən bu operatorun ümumi görünüşü aşağıdakı kimidir:

IF L THEN S,

burada **L** – məntiqi ifadə olub, «doğru» (sıfırdan fərqli) və ya «yalan» (sıfır) məntiqi qiymətlərini ala bilir, **S**-isə hər hansı bir operator və ya bir -birindən ikinöqtə ilə ayrılan operatorları bildirir.

Bu operator yerinə yetirilərkən **L** məntiqi ifadəsi hesablanır, əgər **L** - «doğru» qiymətini alırsa və ya **L** şərti ödənirsə, idarəetmə **S** operatoruna verilir və o yerinə yetirilir, əks halda, yəni **L** - «yalan» qiyməti alırsa və ya **L** şərti ödənmirsə, onda **S** operatoru yerinə yetirilmir və idarəetmə şərt operatorundan sonra gələn birinci operatora verilir.

Misallar:

```
10 T=1
20 PRINT T,T^2
30 T=T+2
40 IF T<=10 GOTO 20
50 END
```

```
10 x=1
20 y=SIN (x)
30 PRINT x,y
40 x=x+0.1
50 IF x<=2 THEN 20
```

Qeyd edək ki, əgər **S** operatoru **GOTO** şərtsiz keçid operatorudursa, onda ya **THEN** sözünü, ya da **GOTO** sözünü yazmamaq olar. *Misal*:

```
10 INPUT N
20 F=1:I=1
30 F=F*I
40 I=I+1
50 IF I<=N  THEN 30
60 PRINT "N=";N,"F=";F
70 END
```

Bu proqram **N** natural ədəd üçün faktorialın hesablanmasını təmin edir.

Tam şərt operatoru. Şərt operatoru **ELSE** sözü ilə tamamlana bilər, bu sözdən sonra **L** məntiqi ifadəsinin aldığı qiymət «yalan» olduqda yerinə yetiriləcək əməliyyat göstərilir. Nəticədə, tam şərt operatoru adlandırılan aşağıdakı operatorda **L** ifadəsinin ala biləcəyi ixtiyari qiymətdə, yerinə yetiriləcək bütün əməliyyatlar göstərilir:

```
IF L THEN S ELSE S1
```

Bu operator yerinə yetirilərkən **L** şərti yoxlanılır. **L** ödənilirsə, yəni **L** ifadəsi «doğru» qiymətini alırsa **S** operatoru, əks halda yə'ni **L** «yalan» qiymətini aldıqda **S1** operatoru yerinə yetirilir. Qeyd edək ki, **S** və **S1**-in yerində proqramın ixtiyari sətrinin nömrəsi də dura bilər.

Misal: $S=\begin{cases} x^2, & x<0 \\ 1+x, & x\geq 0 \end{cases}$ funksiyasının qiymətini tapaq.

```
10 INPUT X
20 IF X <0 THEN S= X ^2 ELSE S=1+ X
30 PRINT S
40 END
```

Bu operator bir-birinin daxilində də verilə bilər. *Məsələn*:

```
50 IF A=B THEN IF B=C THEN PRINT "A=C"
ELSE PRINT   "A<>C"
```

Digər misallara baxaq:

```
10 INPUT X,Y
20 IF X<Y THEN A=X: B=Y
   ELSE A=Y: B=X
30 PRINT "A=";A,"B=";B
```

```
10 INPUT X
20 Y=COS(X):Z=SIN(X)
30 IF (2<=X) AND
(X<=10) THEN
Y=SIN(X):Z=Y*X
40 R=Y+Z^2
50 PRINT R
```

```
10 X=-5
20 PRINT TAB(X*X);"*"
30 X=X+1
40 IF X<5 GOTO 20
```

```
10 X=0
20 PRINT TAB(X);"*"
30 X=X+1
40 IF X<=15 THEN 20
```

4.8. Dövr operatorları. FOR - NEXT dövr operatoru

Bu operatorun ümumi görünüşü aşağıdakı kimidir:

FOR I=E1 TO E2 STEP E3
<dövrü təşkil edən operatorlar>
NEXT I

Burada **I**–dövrün parametri adlanan sadə ədədi dəyişəndir, **E1,E2,E3** isə ədədi ifadələr olub, uyğun olaraq dövr parametrinin başlanğıc qiymətini, son qiymətini və dəyişmə addımını təyin edirlər, **TO** və **STEP** sözləri isə ayırıcı rolunu oynayırlar.

FOR operatorundan sonra dövrün gövdəsini təşkil edən operatorlar yerləşir, yəni burada dövr parametrinin hər bir qiyməti üçün yerinə yetirilən operatorlar ardıcıllığı yerləşir. Dövrün gövdəsi **NEXT I** operatoru ilə tamamlanır.

FOR - NEXT operatoru yerinə yetirilərkən, birinci növbədə (əgər ehtiyac varsa) dövr parametrinin başlanğıc qiyməti, son qiyməti və dəyişmə addımı hesablanır və yaddaşda saxlanılır. Sonra dövr parametrinə **E1** , başlanğıc qiyməti mənimsədilir: **I=E1**. Dövrün gövdəsini təşkil edən proqram operatorları yerinə yetirilir. Sonra dövr parametrinin qiyməti dəyişmə addımı qədər dəyişdirilir (əgər **E3**>0 olarsa, **E3** addımı qədər artırılır: **I=I+E3**, əgər **E3**<0 olarsa **E3** qədər azalır: **I=I-E3**) və **I** dövr parametrinin cari qiymətini **E2** qiymətindən böyük olub-olmaması (**E3**

addımı müsbət olduqda) şərti və ya **I** dövr parametrinin cari qiymətinin **E2**-dən kiçik olub-olmaması (**E3** addımı mənfi olduqda) şərti yoxlanır. Əgər **I** dövr parametrinin qiyməti **E1** başlanğıc qiyməti ilə **E2** son qiymətləri intervalına daxildirsə, onda dövrün gövdəsini təşkil edən operatorlar təkrarən yerinə yetirilirlər, yox əgər **I** parametri bu intervala daxil deyilsə, onda idarəetmə proqramda **NEXT I** operatorundan sonra gələn birinci operatora verilir, yəni dövrdən çıxış yerinə yetirilir. Əgər dəyişmə addımı birə bərabərdirsə, onda **STEP 1** konstruksiyası verilməyə də bilər.

Misal. Birinci N natural ədədin cəmini və hasilini tapaq:

```
10 INPUT N                    10 INPUT N
20 S=0                        20 P=1
30 FOR I=1 TO N               30 FOR I=1 TO N
40 S=S+I                      40 P=P*I
50 NEXT I                     50 NEXT I
60 PRINT S                    60 PRINT P
70 END                        70 END
```

Dövr operatorları bir-birinin daxilində də verilə bilər. Bu halda daxildəki dövr tamamilə xaricdəki dövrün gövdəsində yerləşməlidir. Dövrlər bir-birinin daxilində verilərkən, əvvəlcə daxili dövr, sonra isə xarici dövr **NEXT** operatoru ilə bağlanmalıdır və bu dövrləri bağlamaq üçün bir **NEXT** operatorundan istifadə edib, onun siyahısında (bir-birindən vergüllə ayırmaqla) dövr parametrlərini ardıcıl vermək olar. Məsələn:

```
10 INPUT N,M
20 DIM A(N,M)
30 FOR I=1TO M
40 FOR J=1 TO M
50 INPUT A(I,J)
60 NEXT J,I
```

WHILE - WEND dövr operatoru (QBASIC). Bu operatorun ümumi görünüşü aşağıdakı kimidir:

WHILE L
< dövrün gövdəsini təşkil edən operatorlar >
WEND

Burada **L** - hər hansı məntiqi ifadədir. Operator yerinə yeti-

rilərkən **L** ifadəsi hesablanır, əgər o, «doğru» qiymətini alırsa, yəni şərt ödənirsə, onda bu operatorda **WHILE** ilə **WEND** sözləri arasında yerləşən və dövrün gövdəsini təşkil edən operatorlar yerinə yetirilir. İdarəetmə **WEND** sözünə çatan kimi yenidən **WHILE** operatoruna qaytarılır. Bundan sonra **WHILE** operatoru təkrarən yerinə yetirilərkən yenə də **L** ifadəsi hesablanır, əgər o, yenə də «doğru» qiymətini alırsa onda dövrün gövdəsindəki operatorlar təkrarən hesablanır. **WHILE** operatorundakı **L** ifadəsi «yalan» qiyməti aldıqda idarəetmə **WEND** sözündən sonra gələn birinci operatora ötürülür, yəni dövrdən çıxış yerinə yetirilir. *Məsələn:*

```
10 X=2
20 WHILE X<12
30 Y=X^3+3
40 PRINT "X=";X, "Y=";Y
50 X=X+2
60 WEND
70 END
```

4.9. Alt proqramlar

Bəzən proqramda eyni bir hesablamalara, müxtəlif başlanğıc verilənlərlə dəfələrlə müraciət etmək lazım gəlir. Bu cür əməliyyatları təyin edən operatorları əsas proqramdan ayırıb, onlara ehtiyac olduqda idarəetməni əsas proqramdan bu operatorlar qrupuna vermək daha əlverişlidir. Aydındır ki, ayırdığımız bu operatorlar yerinə yetirildikdən sonra, idarəetmə əsas proqramda bu operatorlar qrupuna keçidi təmin edən keçid operatoru olan yerə qaytarılmalıdır.

Bu cür operatorlar qrupuna alt proqramlar deyilir. İdarəetmənin alt proqrama verilməsi isə alt proqramın çağırılması adlanır. Alt proqramda axırıncı operator olan və idarəetməni yenidən əsas proqrama qaytaran operatora qayıdış operatoru deyilir.

Alt proqramın çağırışı aşağıdakı operatorla yerinə yetirilir:

GOSUB N

burada **N** - alt proqramın ilk yerinə yetirilən operatorunun nömrəsidir. Bu operator yerinə yetirilərkən, idarəetmə **N** nömrəli

sətirdəki birinci operatora verilir, yəni **GOTO** şərtsiz keçid operatorunda olduğu kimi, lakin burada çağırılmış alt proqram yerinə yetirildikdən sonra idarəetmənin qaytarılmalı olacaq çağırış nöqtəsi yaddaşda saxlanılır. Alt proqramın işi **RETURN** qayıdış operatoru ilə tamamlanır. Bu operator yerinə yetirilərkən idarəetmə, əsas proqramda alt proqramı çağırmış **GOSUB** operatorundan sonra gələn operatora verilir. Alt proqramları əsas proqramdan fərqləndirmək üçün adətən altproqramın sətirlərini 1000-dən başlayaraq, 10 addımı ilə nömrələyirlər. Proqramda alt proqramdan istifadə edildikdə əsas proqramın sonunda **END** son operatorundan istifadə edilməlidir. Bir alt proqramın daxilindən digər alt proqramlara da müraciyət ola bilər.

Qeyd edək ki, alt proqramlarda qayıdış operatorunun **RETURN N** formasından istifadə etmək olar. Bu halda altproqram yerinə yetirildikdən sonra idarəetmə əsas proqramda **RETURN** operatorunda göstərilmiş **N** nömrəli sətirə veriləcəkdir.

Misal. Verilmiş iki kvadrat tənliyi həll etməli və $C_n^m = \frac{n!}{m!(n-m)!}$ kombinezonun qiymətini tapmalı.

```
10 N=1
20 INPUT A,B,C
30 GOSUB 1000
40 PRINT
"X1=";X1,"X2=";X2
50 N=N+1
60 IF N>2 THEN 70 ELSE 20
70 END
1000 D=B^2-4*A*C
1010 IF D>=0 THEN 1040
1020 PRINT "tənliyin kökü
yoxdur"
1030 STOP
1040 X1=(-B+SQR(D))/(2*A)
1050 X2=(-B-SQR(D))/(2*A)
1060 RETURN
```

```
10 INPUT N,M
20 K=N:GOSUB 1000
30 P1=P
40 K=M:GOSUB 1000
50 P2=P
60 K=N-M:GOSUB
1000
70 P3=P
80 C=P1/(P2*P3)
90 PRINT C
100 END
1020 P=P*I
1030 NEXT I
1040 RETURN
```

4.10. Massivlər

Massiv – tam tipli indeksli eyni bir adla işarələnən bircins kəmiyyətlərin nizamlanmış külliyatıdır. BASIC dilində bir və ikiölçülü (QBASIC-də isə hətta səkkizölçülü massivlər mümkündür) massivlərdən istifadə olunur. Onlar da sadə dəyişənlər kimi tam, mətn və s. tipli ola bilər. Massiv elementinin ümumi şəklini aşağıdakı kimi vermək olar:

1) birölçülü massivin elementi : *<ad>*(**K**);

2) ikiölçülü massivin elementi: *<ad>*(**I,J**),

burada *<ad>* massivin adı, *k* ($k \geq 0$), birölçülü massiv elementinin indeksi, *i, j* ($i \geq 0, j \geq 0$) isə ikiölçülü massiv elementinin indekslərini (kəsişməsində durduğu sətir və sütunun nömrələri) bildirir. QBASIC-də *k, i, j* üçün başlanğıc qiymətləri vahid qəbul etdirmək olar. Bu indekslər ixtiyari hesabi ifadə ilə verilə bilər. Bu ifadənin hesablanması zamanı QBASIC-də nəticə ən yaxın tam ədədə qədər yuvarlaqlaşdırılır. Məsələn, **P$(0)**, **C2(101)**, **X(46,5*K+1)**, **T%(N/2,M)** və s. Proqramda massivlərdən istifadə edildikdə, onlar qabaqcadan elan edilməlidirlər, yəni EHM-ə bu massivin tipi və ölçüləri haqqında qabaqcadan məlumat çatdırılmalıdır. Bu DIM operatoru vasitəsilə yerinə yetirilir. Operatorun şəkli aşağıdakı kimidir:

1) birölçülü massiv halında **DIM** <ad> (*l*)

2) ikiölçülü massiv halında isə **DIM**<ad>(*n, m*), burada **DIM** operatoru işarə edən işçi ad, *l, n, m* isə massivlərin ölçülərini göstərən qiymətlər, yəni *l* – birölçulu massivin sonuncu elementinin nömrəsi, *n (m)* isə ikiölçülü massivin sonuncu sətrinin (sütununun) nömrəsidir. DIM operatorunda verilən informasiyaya əsasən EHM yaddaşda buradakı hər bir massiv üçün tələb olunan ölçüdə yer ayırır. Massivin ölçüsü BASIC dilinin bir çox versiyalarında (o cümlədən QBASIC-də) tam ədəd və ya tam dəyişənlə ifadə olunur. Bir DIM operatorunda ixtiyari sayda massivi elan etmək olar. Bu operatoru proqramın əvvəlində yerləşdirmək məqsədə uyğundur. Proqramda eyni bir adla həm sadə dəyişəni, həm də massivi işarə etmək olmaz. Misallara baxaq:

1) $S = \sum_{i=1}^{n} a_i$

```
10 INPUT N
20 DIM A(N)
30 FOR I=1 TO N
40 INPUT A(I)
50 NEXT I
60 S=0
70 FOR I=1 TO N
80 S=S+A(I)
90 NEXT I
100 PRINT S
```

$b_i = \sum_{j=1}^{8} a_{ij}, \; i = \overline{1,5}$

```
10 DIM A(5,8), B(5)
20 FOR I=1 TO 5
30 FOR J=1 TO 8
40 INPUT A(I,J):NEXT J,I
50 FOR I=1 TO 5
60 S=0
70 FOR J=1 TO 8
80 S=S+A(I,J)
90 NEXT J
100 B(I)=S: PRINT B(I)
110 NEXT I
```

4.11. Fayllar

Adətən bir proqramda çox böyük həcmdə verilənlərdən istifadə olunur. Bu verilənləri bir yerdə cəmləyib EHM-in operativ yaddaşından kənarda saxlamaq əlverişli olur. Bu cür verilənlər yığımına fayllar deyilir. Fayllar isə disklərdə yerləşdirilirlər. Fayllar öz növbəsində yazılışlardan ibarətdir. Yazılış isə verilənlərin bir və ya bir neçə qiymətindən ibarət olur. Məlumat yazılışının hər bir elementinin qiymətinə sahə deyilir.

Fayllardakı lazımi yazılışların tapılması üçün iki üsul vardır. Birinci üsul ardıcıl müraciət üsuludur, bu üsula görə faylın əvvəlindən başlayaraq, hər bir yazılışın, lazımi yazılış tapılana qədər yoxlanılması aparılır. İkinci üsul birbaşa müraciət üsuludur, bu üsulda isə yazılışa onun nömrəsi üzrə müraciət olunur. İxtiyari fayl, ya birbaşa müraciət faylı, ya da ardıcıl müraciət faylı kimi qurula bilər.

Ardıcıl müraciət faylını qurmaq daha sadədir və bu cür fayl EHM-in yaddaşında daha az yer tutur, lakin bu faylda lazımi yazılışını tapmaq üçün daha çox vaxt tələb olunur. Birbaşa müraciət faylının qurulması daha mürəkkəbdir və bu fayl yaddaşda çox yer tutur, lakin bu cür faylda lazımi yazılışı çox asanlıqla və tez tapmaq olur.

Hər bir fayl, birdən az olmayan və səkkizdən çox olmayan

simvola malik adla işarələnir. Faylın adından sonra nöqtə qoyulur və ardınca üç simvoldan artıq olmayan söz gəlir. Buna fayl adının genişlənməsi deyilir və o, faylın tipini işarə etməyə imkan verir. Faylda ad genişlənməsindən istifadə etməmək də olar. Faylın adında hərflərdən, rəqəmlərdən və bəzi xüsusi işarələrdən istifadə etmək olar. Məsələn, **FILE F1.BAS, F2.DAT** fayl adlarının genişlənməsi birinci faylda Beyzik dilində qurulmuş proqram, ikincidə isə verilənlər külliyatı yerləşdiyini bildirir. Faylın hansı diskdə olduğunu bildirmək üçün onun adının qarşısında uyğun diskin nişanı verilə bilər. Məsələn: **B:PROG.BAS** fayl adı, **PROG.BAS** faylının **B** diskində olduğunu bildirir.

Verilənlər faylı diskdə adları üzrə tapılsa da, proqramda fayla müraciət onun nömrəsi üzrə həyata keçirilir. Faylın nömrəsi, fayl açılarkən onun adı ilə birlikdə verilir. Fayllar isə aşağıdakı operatorla açılır:

OPEN *"faylın adı"* **AS #N,**

burada **AS**- ayırıcıdır, **N**- faylın nömrəsi olub, adətən 1,2,3 qiymətlərini alan ixtiyari ədədi ifadə, dəyişən və ya ədədi sabitdir. Qeyd edək ki, əgər EHM-də bir neçə disk varsa, faylın adında faylın yerləşdiyi disk də göstərilməlidir.

Məsələn:

```
100 OPEN "C:F1.BAS" AS #1
```

Faylları bağlamaq üçün aşağıdakı operatordan istifadə olunur:

CLOSE # N

Bu operator göstərilən **N** nömrəli faylı bağlayır və əgər ondan sonra gələn növbəti operator digər **OPEN** operatordursa, fayl nömrəsini, ondan təkrarən istifadə etmək üçün azad edir. Məsələn: **200 CLOSE #1** operatoru **F1.BAS** adlı və 1 nömrəli faylı bağlayır.

OPEN operatorunun aşağıdakı formalarından istifadə etmək olar:

OPEN «faylın adı» **FOR OUTPUT AS #N**
OPEN «faylın adı» **FOR APPEND AS #N**
OPEN «faylın adı» **FOR INPUT AS #N**
OPEN «faylın adı» **AS #N LEN =L**

burada **N**- faylın nömrəsi, **L**- fayldakı yazılışların (baytlarla) uzunluğudur.

Birinci üç operator ardıcıl müraciətli fayllara aiddir. Onlardan birincisi verilmiş adlı fayl yaradır və yazılışları onun əvvəlindən başlayaraq yerləşdirir. Əgər bu adlı fayl artıq var idisə, o silinir və onun yerində elə bu adlı başqa fayl açılır. İkinci operator verilmiş adlı faylı, digər fayllar arasında axtarır və tapdıqda yeni yazılışları bu faylda artıq mövcud yazılışlar varsa, onların ardınca daxil edir. Əgər bu operator verilmiş adlı fayl tapmazsa, onda dərhal həmin adlı fayl yaradır. Üçüncü operator isə verilmiş adlı faylı axtarır, bu cür fayl olmadıqda isə səhv barədə məlumat verir. Nəhayət, axırıncı operator isə həm fayla verilənləri daxil etməyə, həm də verilənləri götürməyə imkan verən birbaşa müraciətli fayl açır. Burada daxil edilən və çıxarılan yazılışların uzunluğu **LEN** operatoru ilə təyin edilir. Bu cür faylda bütün yazılışlar eyni uzunluqlu olub, adətən 128 bayta bərabər olur.

PRINT # və **PRINT USING #** sözləri olan operatorlardan qiymətləri fayla yazmaq üçün, **INPUT #** operatorundan isə qiymətləri fayldan oxuyub uyğun dəyişənlərə mənimsəmək üçün istifadə olunur. Bu operatorlar **PRINT**, **PRINT USING** və **INPUT** operatorları kimi işlədilir. Məsələn: **100 PRINT #1,150** operatoru 150 qiymətini 1 nömrəli fayla yazır, **200 INPUT # 1,R** isə 1 nömrəli fayldan qiyməti oxuyub, **R** dəyişəninə mənimsədir.

Birbaşa müraciət faylı açıldıqdan sonra proqramda fayldakı yazılışların strukturu elan edilməlidir. Bu aşağıdakı operator vasitəsilə yerinə yetirilir:

FIELD # N; sahələrin siyahısı

burada **N**- faylın nömrəsi, sahələrin siyahısında isə faylın yazılışlarında yerləşmə ardıcıllığına uyğun olaraq sahələr sadalanır. Onlar **L AS V** formatlı ifadə şəklində göstərilir, bu ifadə hər bir sahə üçün L uzunluğunu təyin edir və verilmiş sahəyə proqramdan müraciət üçün **V** dəyişən adını verir. *Məsələn*:

100 FIELD # 1, 10 AS P$, 4AS K$

FIELD operatorunda verilən sahələrin ümumi uzunluğu, **OPEN** operatorunda bu fayl üçün verilmiş faylın uzunluğundan böyük olmamalıdır. **FIELD** operatorunda bütün dəyişənlər sətir

tipli olmalıdır, belə ki, onların köməyi ilə qiymətlər, birbaşa müraciət fayllarına daxil edilir və oradan oxunur.

Birbaşa müraciət faylında verilmiş sahəni sətri qiymətlə doldurmaq üçün aşağıdakı operatorlardan istifadə olunur:

```
LSET V=E
RSET V=E
```

burada **V**- sahənin sətir dəyişəni, **E**- isə ona mənimsədilən sətir qiymətidir. **LSET** operatoru sətir qiymətlərini sahənin sol mövqeyindən başdayaraq **RSET** isə sağ mövqeyindən başlayaraq yerləşdirir. Sahənin boş qalan yerləri (uyğun olaraq sağ və sol tərəfdən) boş yerlərlə doldurulur.

Hazır yazılışı birbaşa müraciət faylına yazmaq üçün aşağıdakı operatordan istifadə olunur:

```
PUT # N,M
```

burada **N**-faylın nömrəsi, **M**-yazılışın nömrəsidir.

Yazılışın birbaşa müraciət faylından oxunması isə aşağıdakı operatorla yerinə yetirilir:

```
GET # N,M
```

burada **N**-faylın, **M**-isə yazılışın nömrəsidir.

Proqramda fayllarla iş üçün **EOF(N)** funksiyasından istifadə etmək əlverişlidir. **EOF(N)** funksiyası proqramda N nömrəli faylın sonuna çatdıqda «doğru», əks halda «yalan» qiymətlərini alır (burada **EOF** – «End of File» sözlərinin baş hərflərindən götürülüb). Bu funksiyadan faylın uzunluğu dəqiq məlum olmadığı halda oradan verilənlərin oxunması üçün istifadə olunur. Aşağıdakı misallara baxaq:

1) **N** sayda ədədi **E** diskindəki **D** faylına yazmalı;

2) **E** diskindəki **D** faylının cüt nömrəli elementlərini, bu diskdəki D1 faylına yazmalı;

3) **E** diskinin **D** faylındakı **K**-cı elementi çapa verməli.

1)
```
10 OPEN "E:\D" FOR OUTPUT AS #1
20 INPUT N
30 FOR I=1 TO N:INPUT P:PRINT #1,P
40 NEXT I
50 CLOSE #1
```

2)
```
10 OPEN "E:\D" FOR INPUT AS #1
20 OPEN "E:\D1" FOR OUTPUT AS #2
```

```
30 WHILE NOT EOF(1):INPUT #1,A,B;
40 PRINT #2,B:PRINT A;B
50 WEND:CLOSE
60 OPEN "E:\D1" FOR INPUT AS #2
70 IF EOF(2) THEN GOTO 100
80 INPUT #2,B:PRINT B;
90 GOTO 70
100 CLOSE:END
```

3)
```
10 OPEN "E:\D" FOR INPUT AS #3
20 INPUT K
30 FOR I=1 TO K:INPUT #3,T : NEXT I
40 PRINT T : CLOSE #3 : END
```

4.12. Əmrlər sistemi

EHM-də proqramın yerinə yetirilməsi əmrlər vasitəsilə icra edilir. Bu əmrlərlə proqram EHM-ə daxil edilir, yoxlanılır, səhvlər varsa düzəldilir, proqramın mətni EHM-in yaddaşında saxlanılır, proqramın mətni fayldan çıxarılır, proqramda hesablamalar yerinə yetirilir.

Proqramı, klaviaturadan EHM-ə daxil etməmişdən qabaq **NEW** əmri verilir. Bu əmr cari yaddaşda olan proqramı silir və yeni proqram üçün EHM-in yaddaşını boşaldır. Bu əmr eləcə də bütün faylları bağlayır.

AUTO əmri proqramın sətirlərinin avtomatik nömrələnməsini təmin edir. Bu əmrin ümumi şəkli aşağıdakı kimidir:

AUTO N, SN,

burada **N**-proqramın nömrələnəcək sətirlərinin başlanğıc nömrəsi, **SN** isə nömrələrin dəyişmə addımını göstərir. Məsələn: **AUTO 10,5** əmri proqramın, birinci operatoruna **10** nömrəsini, ikinciyə 15 nömrəsini və s. mənimsədir. Əgər **AUTO** əmrinin siyahısı, yəni **N** və **SN** parametrləri verilməzsə, onda belə əmr **AUTO 10,10** əmrinə ekvivalent olacaqdır. Əgər əmrin siyahısında SN parametri verilməyib, ancaq **N** parametrindən sonra vergül verilibsə, onda dəyişmə addımı kimi əvvəlki **AUTO** əmrində verilmiş addım qəbul edilir. Əmrdə **N** parametri verilmədikdə, **SN** addım təyin edildikdə, nömrələmə sıfırdan başlayaraq verilmiş

addımla aparılır.

Proqramda sətirlərin nömrələrini dəyişdirmək üçün **RENUM NN,NO,SN** əmrindən istifadə olunur. Bu əmrin siyahısındakı **NN** parametri birinci mənimsədiləcək yeni sətir nömrəsini, **NO** dəyişdiriləcək birinci köhnə nömrəni və **SN**-sətir nömrələrinin dəyişmə addımını bildirir. Məsələn: **RENUM 100,10,10** əmri proqramın **10** nömrəli sətrindən başlayaraq, bu sətrin nömrəsini yeni **100** nömrəsilə, növbəti sətrin nömrəsini **110** ilə və s. dəyişəcəkdir.

RENUM əmrində verilən ixtiyari parametri verməmək də olar. Belə ki, **NN** parametri verilmədikdə onun qiyməti 10-a bərabər, **NO** verilmədikdə onun qiyməti proqramdakı birinci sətrin nömrəsinə bərabər və nəhayət **SN** verilmədikdə dəyişmə addımı **10** qəbul edilir.

Proqramın adını dəyişdirmək üçün **RENAME** yeni ad əmrindən istifadə edilir.

Proqramı EHM-ə daxil etdikcə ekrana sığışmayan sətirlər avtomatik olaraq birincidən başlayaraq yuxarıya doğru hərəkət etdirilir və nəzərdən itirilir. Proqram yığıldıqdan sonra bu nəzərdən itirilmiş sətirlərə baxmaq və ümumiyyətlə yığılmış proqramı yenidən displeyin ekranına çağırmaq üçün **LIST** əmrindən istifadə edilir. Bu əmr yalnız **LIST** sözündən ibarətdirsə, onda ekrana bütün cari proqram çağırılır. Eləcə də bu əmrlə proqramda bizə lazım olan sətirləri də ekrana vermək olar. Bu halda əmrin **LIST N1-N2** formatından istifadə olunur.Onda ekrana proqramın **N1** nömrəli sətrindən başlayaraq **N2** nömrəli sətrinə qədər olan hissəsi verilir. Əmr **LIST N1**-kimi verilərsə, ekrana proqramın **N1** nömrəli sətrindən sonuna qədər olan hissəsi, **LIST-N2** kimi olduqda proqramın ən kiçik nömrəli sətrindən **N2** nömrəli sətrinə qədər olan hissəsi çıxarılır. **LIST N** əmri proqramın **N** nömrəli sətrini ekrana çıxarır.

Proqramın ixtiyari sayda sətri çap qurğusu vasitəsilə kağız üzərində çap oluna bilər. Bunun üçün **LLIST** əmrindən istifadə edilir. Bu əmrdən istifadə qaydaları **LIST** əmrində olduğu kimidir.

Proqram EHM-ə daxil edilərkən, proqrama yeni sətirlər daxil etmək və ya bəzi sətirləri çıxarmaq lazım gəlir. Mövcud sətri

yeni sətirlə əvəz etmək üçün klaviaturadan həmin nömrəli yeni sətir yığıb, **ENTER** düyməsini basmaq kifayətdir, onda EHM yeni yığılmış sətri həmin nömrəli sətrin yerinə qoyacaq, köhnə sətir isə silinəcəkdir. Əgər iki mövcud sətir arasına yeni sətir daxil etmək lazımdırsa, onda bu sətirə həmin iki sətrin nömrələri arasında yerləşən nömrə verib, klaviaturadan daxil etmək lazımdır. Sonra **ENTER** düyməsini basdıqda bu sətir həmin sətirlər arasında yerləşdiriləcəkdir. Bu əməliyyatları yerinə yetirmək asan olsun deyə, proqram tərtib edilərkən sətirlərin nömrələrini müəyyən intervalla (məsələn 5 və ya 10 intervalı ilə) vermək lazımdır. Müəyyən nömrəli sətri proqramdan çıxarmaq üçün həmin sətrin nömrəsini yığıb, **ENTER** düyməsini basmaq lazımdır. Onda həmin nömrəli sətir proqramdan çıxarılacaqdır.

Proqramdan birdən çox sayda sətri çıxarmaq lazım gəldikdə isə **DELETE** əmrindən istifadə olunur. Bu əmrin formatı **LIST** əmrində olduğu kimidir, yəni **LIST** əmri üçün dediklərimizi **DELETE** əmrinə də aid etmək olar.

Cari proqramın yerinə yetirilməsi **RUN** əmrinin verilməsi ilə başlanır. Bu əmr verildikdən sonra proqramın operatorları onların sətir nömrələrinin artımı boyunca yerinə yetirilməyə başlanır. Bu əmr proqramın ən kiçik nömrəli sətrindən başlayaraq yerinə yetirilməsini təmin edir. Əgər proqramın onun hər hansı **N** nömrəli sətrindən başlayaraq yerinə yetirilməsi tələb olunarsa, onda **RUN N** formalı əmrdən istifadə olunur.

Proqramın yerinə yetirilməsi proqramda **STOP** operatoru verilməklə dayandırıla bilər. Bu operator verilərkən, proqramın yerinə yetirilməsi dayandırılır və dayanma baş vermiş sətrin nömrəsi ekrana verilir. Proqramın işi klaviaturadakı uyğun düymənin basılması ilə də dayandırıla bilər. Hər iki halda proqramın işini davam etdirmək üçün **CONT** əmrindən istifadə etmək olar.

Proqramın normal sona çatması isə proqramın və ya proqramda alt proqramdan istifadə edilirsə, əsas proqramın sonuncu operatoru olan **END** operatoruna çatdıqda baş verir. Lakin bu operator yerinə yetirildikdən sonra artıq ekrana axırıncı yerinə yetirilmiş sətrin nömrəsi haqqında heç bir məlumat verilmir.

Proqramın işinin dayandırılması, proqramda səhv tapıldıqda da baş verir. Bu halda ekrana səhv barəsində məlumat verilir.

Proqramın ayrıca bir sətrinin redaktə edilməsi **EDIT** əmri

ilə həyata keçirilir. Bu əmrin ümumi şəkli **EDIT N** kimidir. Burada **N** redaktə edilən sətrin nömrəsidir. Bu əmr göstərilən nömrəli sətri redaktə etmək üçün ekrana çıxarır və bu zaman kursor sətrin nömrəsindən sonra gələn birinci simvolun durduğu mövqeydə durur. Sonra sətri redaktə etmək (simvolları dəyişdirmək, çıxarmaq və ya əlavə etmək) olar.

Adətən istifadəçi eyni zamanda bir neçə proqramla işləmir və operativ yaddaşda cari anda yalnız bir proqram olur. Buna görə də digər proqramları xarici yaddaş qurğularında saxlamaq və lazım olduqca onları aradan çıxarmaq lazım gəlir. Beyzik dilində proramları fayllara yazıb saxlamaq və lazım olduqca onları xarici yaddaşdan çıxarmaq imkanları var.

Proqramı operativ yaddaşdan verilmiş fayla göndərmək üçün (yazmaq üçün) **SAVE** və **LIST** əmrindən istifadə olunur. Əgər bu əmrlərin siyahısında heç bir şey qeyd edilmirsə, onda diskdə cari proqram öz adı ilə saxlanılacaqdır. Bu əmrlərdə proqramın yazılacağı faylın adı dırnaq içərisində də verilə bilər. Bu halda əgər artıq həmin adlı fayl var idisə, onda həmin köhnə fayl silinir, onun yerinə həmin adlı yeni fayl yazılır. *Məsələn*:

```
LIST
SAVE
LIST «C:COEF.BAS»
SAVE «C:COEF.BAS»
```

LIST əmri ilə proqramın bütün simvollarını displeyin ekranına çıxarmaqla yanaşı, ixtiyari xarici qurğuya da göndərmək olar. Xüsusi halda **LIST** əmri ilə proramın ayrı-ayrı hissələrini də fayla yazmaq olar. Bunun üçün əmrdə proqramın fayla yazılacaq hissəsinin operatorlarının sətir nömrələrinin uyğun dəyişmə diapazonunu göstərmək lazımdır. Məsələn: **LIST 10-100, «COEF»**

SAVE əmri yerinə yetirilərkən proqram həmişə tamamilə fayla yazılır. Adətən bu zaman proqramın mətni sıxılır və fayl ikilik formatda saxlanılır. Bu sıxma əməliyyatını ləğv etmək üçün **SAVE** əmrinin sonunda vergül qoyub, **A** simvolunu əlavə etmək lazımdır: **SAVE «COEF», A**

SAVE əmri proqramdan başqalarının istifadə etməsinin qarşısını almaq üçün onun fayla şifrlənmiş formada yazılışını təmin edir. Əgər şifrlənmiş proqram yenidən operativ yaddaşa qaytarılarsa, onda bu proqrama artıq **LIST** və **EDIT** əmrlərini tətbiq

etmək və proqram üzərinə qoyulmuş müdafiəni heç bir üsulla götürmək mümkün olmayacaqdır. Proqramı şifrləmək üçün **SAVE** əmrinin sonunda vergül işarəsi qoyub, **P** simvolunu daxil etmək lazımdır. Məsələn: **SAVE «COEF», P.**

İxtiyari xarici qurğuda yerləşdirilmiş proqram, məsələn, diskdəki fayla **LIST** və **SAVE** əmrləri ilə yazılmış proqram, **LOAD** əmri vasitəsilə fayldan çıxarılıb, operativ yaddaşa yüklənir.

Bu əmr yerinə yetirilərkən, EHM-in yaddaşı təmizlənir və yeni proqram, bu əmr yerinə yetirilənə qədər yaddaşda olan proqramın yerini tutur. Əmrdə **LOAD** sözündən sonra dırnaq içərisində faylın adı verilir. Əgər əmrdə faylın adında diskin adı verilməzsə, cari diskovod istifadə olunur. Məsələn: **LOAD «COEF. BAS».**

LOAD əmri proqramı yalnız operativ yaddaşa yükləyir, lakin onu yerinə yetirmir. Fayldan çıxarılan proqramın yerinə yetirilməsi üçün **LOAD** əmrində faylın adından sonra vergül qoyub, **R** parametrini yazmaq lazımdır. Məsələn: **LOAD «COEF. BAS», R.** Bu halda proqram birinci operatordan başlayaraq yerinə yetirilir. Proqramın yaddaşa yüklənməsi və dərhal yerinə yetirilməsi üçün həmçinin **RUN** əmrindən də istifadə edilir. Bunun üçün **RUN** sözündən sonra faylın adını da vermək lazımdır. Məsələn: **RUN «COEF. BAS»**

Proqramı fayldan yaddaşa yükləyərkən, yaddaşda olan əvvəlki proqramı saxlamaq da olar. Belə ki, bunun üçün **MERGE** əmrindən istifadə olunur. Bu əmr yerinə yetirilərkən verilmiş fayldakı proqramın sətirləri, cari anda yaddaşda olan proqramın mətninə birləşdirilirlər (əlavə olunurlar). Məsələn, əgər yaddaşda **PROGRAM 1** proqramı var idisə, **MERGE «PROGRAM 2»** əmri **PROGRAM 2** proqramının sətirlərini **PROGRAM 1** proqramının sətirlərinə əlavə edir.

Əgər **LOAD**, **RUN** və ya **MERGE** əmrləri yerinə yetirilərkən, EHM göstərilən adlı faylı tapa bilməzsə, onda o, səhv barəsində məlumat verəcəkdir.

Bəzi hallarda diskdə olan faylların adları haqqında məlumat almaq tələb olunur. Yalnız bir **FILES** sözündən ibarət əmrin köməyi ilə cari diskdə olan bütün fayl adları ekrana çıxarılır. Lakin eyni zamanda seçdiyimiz ixtiyari diskdəki fayl adları və hətta hər

hansı konkret bir fayl haqqında məlumat almaq olar. Bunun üçün FILES əmrində diskin adını, faylın adını və ya hər ikisini dırnaq arasında vermək lazımdır. Bu əmrdə fayl adında «*» işarəsindən istifadə etmək olar, həmin işarə ilə ixtiyari fayl adı və ya ad genişlənməsi verilə bilər. Məsələn: **FILES «*.BAS»** əmri cari diskdəki **BAS** ad genişlənməsinə malik bütün fayl adlarını çıxarır, **FILES «B:»** və ya **FILES «B:*»** əmri isə **B** diskində olan və ad genişlənməsi olmayan bütün fayl adlarını çıxarır: **FILES «B:*.*»** əmri **B** diskində olan ixtiyari adlı bütün fayl adlarını çıxarır; **FILES «*.*»** əmri isə bütün mümkün fayllar haqqında məlumat verilməsini təmin edir.

Diskdəki faylı ləğv etmək üçün **KILL** «faylın adı» əmri istifadə olunur. Əmrdə faylın adında diskin adı da verilə bilər. Məsələn: **KILL «A:COEF.BAS»**. Bu əmr ixtiyari tipli faylların ləğv edilməsi üçün yarayır.

İxtiyari faylın adı yeni adla əvəz edilə bilər. Bunun üçün aşağıdakı əmrdən istifadə olunur: **NAME** «köhnə ad» **AS** «yeni ad», burada «köhnə ad» və «yeni ad» - faylın köhnə və yeni adlarını göstərmək üçün istifadə olunan sətir ifadələridir. Diskdə köhnə adlı fayl olmalı yeni adlı fayl isə olmamalıdır. Əks halda səhv olduğu barədə məlumat veriləcək. Əgər adı dəyişdirilən fayl cari diskdə deyilsə, onun köhnə adı qarşısında diskin adı verilə bilər. Faylın yeni adı qarşısında isə heç bir ad verilə bilməz. Məsələn: **NAME «A: COEF.BAS» AS «ACTS.BAS»**. Bu əmrin yerinə yetirilməsi nəticəsində, həmin fayl **A** diskində yeni **ACTS. BAS** adı ilə qalacaqdır.

4.13 Mətn informasiyanın emalı.

Mətn informasiya üzərində **BASIC** dilində aşağıdakı əməliyyatlar aparıla bilər.

1) Birləşdirmə (konkanetasiya) əməliyyatı;
2) Mətni ifadənin mənimsədilməsi əməliyyatı;
3) Mətni kəmiyyətlərin müqayisəsi;
4) Mətni funksiyaların tətbiqi;

Birləşdirmə əməliyyatı A+B formasında yazılır, burada A, B – ixtiyari mətni kəmiyyətlərdir. Əməliyyat nəticəsində B-nin

qiyməti, A-nın arxasına birləşdiriləcək. *Məsələn,* «proqram» + « laşdırma» → «proqramlaşdırma».

Mətni ifadələr, mətni sabitlər, dəyişənlər, funksiyalar və birləşdirmə əməliyyatlarının köməyi ilə yaradılır. *Məsələn,*

```
10 A$="ALQO":B$="RITM"
20 C$=A$+B$:PRINT C$:END
```

Buradan nəticə **ALQRITM** olacaqdır.

Simvol (mətni) tipli kəmiyyətlər üzərində aşağıdakı müqayisə əməliyyatları aparıla bilər: =, < >, <, >, <=, =>. *Məsələn,* **"A"<"D","BASIC">="PASCAL"** və s. Belə ki, EHM-də hər bir simvol müəyyən bir kodla ifadə olunub, buna görə də iki simvol tipli kəmiyyətin müqayisəsi onların simvol kodlarının müqayisəsinə gətirilir. BASIC-də münasibət əməliyyatını ixtiyari uzunluqlu iki mətni sabit (dəyişən) üzərində də aparmaq olar. Bu zaman daha kiçik uzunluqlu sətir sağdan probellərlə doldurulur. Sətirlərin müqayisəsi sağdan sola doğru hər bir simvol üçün yerinə yetirilir. Əvvəlcə sətirlərin birinci simvolları müqayisə olunur, əgər onlar bərabərdirsə, ikinci, üçüncü, və s. simvollar müqayisə olunur. Bu müqayisədə birinci bərabər olmayan simvollar həmin müqayisənin cavabını təyin edir. Misal. Daxil edilən simvolun latın əlifbasının böyük hərfi olub olmadığını təyin etməli:

```
10 INPUT C1$
20 IF "A"<=C1$ AND C1$<="Z" THEN B1$="TRUE"
   ELSE B1$ = "FALSE"
30 PRINT B1$ :END
```

İndi isə mətni standart funksiyalara baxaq:

1) **ASC(X$)** funksiyası simvolun ikilik kodunu onluq ədədə çevirir.

2) **LEN(X$)** funksiyası **X$** mətni dəyişənin uzunluğunu, yəni onun daxilindəki simvolların sayını təyin edir. *Məsələn,*

```
10 X$="BAKI":C=LEN(X$):PRINT "C =";C
```

Cavab **C=4** olacaqdır.

3) **INSTR(N,X$,Y$)** funksiyası **Y$** mətnin **X$** mətninə daxil olduğu birinci mövqeyin nömrəsini təyin edir. Axtarış **X$** mətnin N nömrəli sətrindən başlayır. Əgər **Y$** mətni, **X$** mətninə daxil deyilsə, onda funksiyanın qiyməti sıfra bərabərdir. Burada **X$,Y$** mətni sabit və ya dəyişənlərdir, **N** isə ədəd, dəyişən və ya

hesabi ifadədir.

Məsələn,

```
10 X$="FUTBOL":Y$="BOL"
20 Z=INSTR(1,X$,Y$):PRINT "Z=";Z
```

Nəticə **Z** = 4 olacaqdır.

4) **MID$(X$,N,K)** funksiyası **X$** mətnin **N**-ci mövqeyindən başlayaraq K sayda simvolunu çıxarır. Burada **X$** mətni sabit və ya dəyişən, **N, K** isə ədəd, dəyişən və ya hesabi ifadədir.

Məsələn,

```
10 X$="FUTBOL":Y$=MID$(X$,4,3)
20 PRINT "C ="; C$
```

Nəticə **C=BOL** olacaqdır.

5) **LEFT$(X$,K)** funksiyası **X$** mətnin soldakı **K** sayda simvolunu çıxarır.

6) **RIGHT$(X$,K)** funksiyası isə **X$** mətninin sağdakı **K** sayda simvolunu çıxarır. Buradan **X$** mətni sabit və ya dəyişən, **K** isə ədəd, dəyişən və ya hesabi ifadədir. *Məsələn,*

```
10 X$="ALIYEV_A.Y.":L=LEN(X$)
20 Y$=RIGHT$(X$,4):Z$=LEFT$(T$,L-5)
30 T$=Y$+"_"+Z$:PRINT T$:END
```

Nəticə **T = A.Y. ALIYEV** olacaqdır. Daha bir misala baxaq: **N (N< 255)** sayda simvoldan ibarət **T$** mətnində hər bir cümləni ayırıb, yeni sətirdən verməli.

```
10 INPUT T$:B$=".":A=0
20 IF INSTR(A+1,T$,B$)=0 THEN GOTO 60
30 Y=INSTR(A+1,T$,B$):R$=MID$(T$,A+1,Y-A)
40 PRINT R$:A=Y
50 GOTO 20
60 P$="SON"
70 PRINT P$:END
```

Qeyd edək ki, QBASIC-də **WHILE–WEND** dövr operatorundan da istifadə etmək olar, onda 20 və 50-ci sətirlər uyğun olaraq **20 WHILE INSTR (A+1, T$, B$) > 0** və **50 WEND** olardı.

4.14. Dilin qrafik imkanları

Məlumdur ki, informasiya yalnız mətni formada deyil, həm də qrafik formada, yəni qrafik, sxem, şəkil, dioqram və s. formada verilə bilər. Qrafik imkanlardan istifadə edərkən birinci növbədə istifadə olunan monitorun xarakteristikalarını, yəni buradakı koordinat sisteminin istiqamətini OX və OY oxları boyunca nöqtələrin maksimal sayını və istifadə edilə biləcək rənglərin sayını bilmək lazımdır. IBM PC tipli fərdi kompüterlərdə (0,0) nöqtəsi koordinat sisteminin mərkəzi ekranın yuxarı sol küncündə yerləşir. OX oxu soldan sağa doğru üfiqi istiqamətdə, OY oxu isə yuxarıdan aşağıya doğru şaquli istiqamətdədir. OX×OY oxları boyunca nöqtələrin sayı monitorun tipindən asılı olaraq dəyişir, məsələn, 640×200 nöqtə və s. Hər bir EHM-də qeyd olunmuş rənglər çoxluğundan istifadə olunur və bu çoxluqda hər rəng öz nömrəsinə malikdir. Rəngli iş rejimində iki əsas anlayışdan: əsas rəng-təsvirlərin (hərf, xətt, kontur və s.) rəngi və fon (ekranın rəngi) rəngindən istifadə edilir. Qrafik əməliyyatların yerinə yetirilməsi üçün müasir proqramlaşdırma dillərində (eləcə də BASIC (QBASIC) dilində) müəyyən həndəsi fiqur və onların elementlərinin təsvirini təmin edən xüsusi operatorlar var. Hər bir belə operatorda fiqurun (elementin) forma və parametrləri yəni ölçüsü, ekrandakı vəziyyəti və s. göstərilir. Operator bu informasiyanı monitor adapteri üçün əmrlər ardıcıllığına çevirir. Bu ardıcıllığın yerinə yetirilməsi isə monitorun ekranında fiqurun təsvirinə gətirib çıxarır. Ekrana çıxarıla biləcək nöqtələrin və rənglərin sayı iş rejimi ilə müəyyən edilir. İlk fərdi kompüterlərdə monitor yalnız üç iş rejimində; mətni, monoxrom (ağ-qara) qrafik və rəngli qrafik rejimlərdə işləyə bilirdi. Müasir IBM PC tipli kompüterlərdə iş rejimləri daha çoxdur. Eyni bir adapterin müxtəlif iş rejimlərində müxtəlif imkanlar olur. Müxtəlif adapterlər üçün iş rejimlərinin sayı da müxtəlifdir, lakin 13-ü aşmır. Adpterin hər bir iş rejimi nömrə ilə təyin edilir. Burada 0 nömrəli rejim mətni, qalanları isə qrafik iş rejimləridir. Çox saylı qrafik iş rejimlərindən 1, 7 və 8 nömrəli rejimlərə baxaq. İxtiyari qrafik rejimdə işləyərkən iki rəngi – əsas və fon rənglərini vermək lazımdır. Baxdığımız qrafik iş rejimində ixtiyari adapterdə (SVGA istisna olmaqla) öz nömrəsilə təyin edilən 16 rəngdən istifadə edilir: 0 – qara, 1 – göy, 2 –

yaşıl, 3 – mavi, 4 – qırmızı, 5 – al qırmızı, 6 – qəhvəyi, 7 – açıq boz, 8 – tünd boz, 9 – parlaq göy, 10 – parlaq yaşıl, 11 – parlaq mavi, 12 – parlaq qırmızı, 13 – parlaq al qırmızı, 14 – sarı, 15 – ağ. Qrafik iş rejimində fonun rəngi onun nömrəsi ilə təyin edilir və yuxarıdakı siyahıdan ixtiyari rəng ola bilər.

Əvvəlcə rejim 1-ə baxaq. Bu CGA adapterli fərdi kompüterlər üçün əsas qrafik iş rejimidir. CGA, EGA, VGA adapterləri bu iş rejimində əsas rəng kimi 6 rəngdən, daha doğrusu, palitra adlanan iki rənglər qrupundan istifadə edir:

Palitra 0 (1): 1) yaşıl, 2) qırmızı, 3) qəhvəyi;

Palitra 1 (0): 1) mavi 2) al qırmızı 3) ağ.

Burada mötərizə daxilində CGA-da palitranın aldığı nömrələr göstərilib. Qeyd edək ki, 0 nömrəli rəng fonun rəngi ilə üstüstə düşür və təsviri görünməz edir. Zamanın bir anında yalnız bir palitraya müraciət etmək olar və əsas rəngi təyin etmək üçün palitranın nömrəsini (**COLOR** operatorunda) və palitradakı rəngin nömrəsini (qrafik operatorlarda) vermək lazımdır.

COLOR N1,N2 – rəng operatorudur. Burada **N1** – fonun rənginin nömrəsi ($0 \leq$ **N1** ≤ 15), **N2** isə palitranın nömrəsidir (0 və ya 1).

İndi isə 7 və 8 iş rejimlərinə baxaq. Bu iş rejimləri EGA və VGA adapterlərində verilə bilər. Burada palitra 16 rəngdən ibarətdir və yuxarıda verdiyimiz 0 – 15 nömrəli rənglərə uyğundur. Onlar üçün rəng operatoru aşağıdakı şəkildədir:

COLOR N,N1

Burada **N** – əsas rəngin (təsvir rənginin) nömrəsidir, **N1** isə fon rəngidir ($0 \leq$ **N,N1** ≤ 15).

Nəhayət, 0 iş rejiminə baxaq. Bu mətni iş rejimidir. Burada 7 və 8 rejimlərində olan formalı **COLOR** operatorundan istifadə edilir, lakin **N1** ≤ 7 olur. Əgər **N** əsas rəng nömrəsinə 16 rəqəmini əlavə etsək, simvolun rəngi bərq vuracaqdır. Bu iş rejimi susmaqla verilmiş fon və simvol rənglərini dəyişdirməyə imkan verir.

Ekranın iş rejimi aşağıdakı operatorla seçilir:

SCREEN R,R1

Burada **R** – tam tipli sabit və ya ifadə olaraq ekran rejiminin nömrəsini göstərir. Bizim halda bu **R**=0, 1, 7 və ya 8 ola bilər. **R1** isə sabit və ya ifadə olmaqla təsvirin rəngli və ya ağ-qara çıxa-

cağını təyin edir. Belə ki, **R1** < 0 olduqda təsvir ağ-qara, **R1**=0 olduqda isə təsvir rəngli dir. Burada 0 rejimində **R1** əks qiymətlər alır. **R**≥2 rejimlərində isə **R1**-in qiyməti nəzərə alınmır. Məsələn, **SCREEN 1,0**. Qeyd edək ki, baxdığımız 1, 7, 8 qrafik iş rejimlərində **PRINT** və **INPUT** operatorlarından, standart funksiyalardan və mətn emalının digər vasitələrindən adi qaydada istifadə etmək olar, mətn simvollarının rəngi isə **COLOR** operatoru ilə müəyyən olunur.

Dilin əsas qrafik operatorlarına baxaq. Bütün bu operatorlarda təsvir olunan elementin rəngini təyin edən **R** parametri iştirak edir. Onun QBASIC-də rejim 1-də ala biləcəyi qiymətlər **R**=1, 2 və ya 3 palitralarındakı rənglərin nömrəsidir, rejim 7 və 8-də isə **R**=0, 1, 2, ..., 15 qiymətləri ala bilər ki, bu da yuxarıda verdiyimiz rəng nömrələrinə uyğundur. **R** parametri verilməyə də bilər, lakin onun operatordakı yerini müəyyən edən vergül saxlanılır. Bu halda elementin rəngi **COLOR** operatoru ilə təyin edilir. **COLOR** operatoru və operatorlardakı **R** parametri verilmədikdə isə ekranın iş rejimi monoxrom olur. Ən geniş istifadə edilən operatorlar aşağıdakılardır:

1) **CLS** operatoru ekranı təmizləyir, yəni bütün ekranı fonun rəngi ilə rəngləyir.

2) **PSET(X,Y),R** operatoru **R** rəngli, **X** və **Y** koordinatlı nöqtə verir. Burada və gələcəkdə **X,Y** koordinatlarını tam ədədlər və ya tam hesabi ifadələrlə vermək olar. Məsələn, **PSET(35,105),2**.

3) **LINE(X1,Y1)-(X2,Y2),R** operatoru **(X1,Y1)** və **(X2,Y2)** koordinatları olan nöqtələrlə məhdudlaşdırılan düz xətt parçası çəkir. **R** isə xəttin rəngini təyin edir. Məsələn, **LINE(20,20)-(200,100),1**. Bu operator QBASIC-də seçilmiş palitradan asılı olaraq mavi və ya yaşıl düz xətt parçasını təsvir edir.

4) **LINE(X1,Y1)-(X2,Y2),R,P** operatoru koordinat oxlarına paralel tərəfləri olan düzbucaqlı qurur, burada (X1,Y1) düzbucaqlının yuxarı sol təpə nöqtəsinin koordinatlarıdır, (X2,Y2) isə aşağı sağ təpə nöqtəsinin koordinatlarıdır, R düzbucaqlının konturunun rənginin nömrəsidir. P parametri iki cür qiymət ala bilir: P=B olduqda düzbucaqlı qurulur, P=BF olduqda isə

düzbucaqlının daxili konturunun rəngi ilə rənglənir. *Məsələn,*

```
10 SCREEN 1,0:COLOR 6,0
20 LINE(100,110)-(160,160),1,BF
30 LINE(50,60)-(200,180),1,B
40 END
```

QBASIC-də bu proqram qəhvəyi fonda qırmızı konturlu düzbucaqlının daxilində yaşıl rənglə rənglənmiş düzbucaqlı qurur.

Müxtəlif fiqur və onların elementlərini quran proqramlara da baxaq:

```
10 SCREEN 1,0:X=0:Y=160
20 FOR I=0 TO 160:PSET(X,Y),1
30 FOR J=1 TO 40:NEXT J
40 PSET(X,Y),0:X=X+2:Y=Y-1:NEXT I
50 END
```

```
10 SCREEN 1,0:FOR I=120 TO 170 STEP 5
20 LINE(5,I)-(I,I):NEXT I
30 FOR I=10 TO 60 STEP 10
40 LINE(I,I)-(I+50,I+50),,B
50 NEXT I
60 FOR I=10 TO 125 STEP 35
70 LINE(170,I)-(220,I+25),,BF:NEXT I
80 END
```

5) **CIRCLE(X,Y),R,R1** operatoru mərkəzi **(X,Y)** nöqtəsində yerləşən, **R1** nömrəli rəngli konturu olan **R** radiuslu çevrə çəkir. Məsələn, **CIRCLE(160,100),50,2** operatoru QBASIC-də mərkəzi (160,100) nöqtəsində, radiusu 50 olan qırmızı və ya al qırmızı konturlu çevrə çəkir.

CIRCLE operatorunda N parametrindən sonra daha iki parametr B və E parametrləri (uyğun olaraq qövsün başlanğıc və son nöqtələrini bildirir) verilə bilər. Bu B və E parametrləri (onlardan yalnız birlikdə istifadə etmək olar) verildikdə ekranda tam çevrə əvəzinə göstərilmiş uc nöqtəli qövs qurulur. Burada B və E parametrləri radianlarla - 0-dan 6.2831-ə qədər verilməlidir. Məsələn:

```
40 CIRCLE(160,100),50,,0,3.141593
```

operatoru, mərkəzi ekranın mərkəzi ilə üst-üstə düşən, 50 radiuslu, standart ağ rəngli yuxarı yarımçevrə quracaqdır. Qeyd edək ki, bu operatorda rəngi təyin edən N parametri verilməsə də, onun mövqeyi vergüllə qeyd edilmişdir.

Burada B və E parametrlərindən ixtiyarının və ya hər ikisinin qarşısında mənfi işarəsi dura bilər. Bu isə onu bildirir ki, qövsun uyğun uc nöqtəsinə (qarşısında mənfi işarəsi qoyulmuş) çevrənin mərkəzindən radius çəkiləcəkdir. Məsələn:

```
50 CIRCLE(160,100),50,1,-1.570796,-3.141593
```

operatoru, bundan əvvəl verdiyimiz misalda qurulmuş yarımçevrənin ikinci 1/4 hissəsi olan sektoru mavi rənglə qeyd edəcəkdir.

Ekranda ellipslərin qurulması üçün də CIRCLE operatorundan istifadə olunur, təkcə burada daha bir CR parametrini də əlavə etmək lazımdır. Bu CR parametri ellipsin hündürlüyünün onun eninə olan nisbətini bildirir və buna görə də bu parametri adi kəsr şəklində ifadə etmək daha əlverişli olur. Məsələn:

```
100 CIRCLE(170,70),30,1,0,6.2831,1/3
```

operatoru ekranda üfüqi istiqamətdə uzadılmış ellips quracaqdır.

Üfüqi və şaquli istiqamət üzrə koordinatlar bir-birindən fərqləndiyindən, orta imkanlı qrafik iş rejimində 5/6 münasibəti çevrəni təyin edir, CR= 5/12 münasibəti isə yüksək imkanlı iş rejimində çevrəni təyin edəcəkdir.

Misallar.

Aşağıdakı proqram su üzərindəki çevrələrə oxşar çevrələr qurur.

```
10 SCREEN 1,0
20 FOR J=1 TO 3
30 FOR R=10 TO 70 STEP 5
40 CIRCLE(128,96),R,J
50 NEXT R: NEXT J
60 END
```

Aşağıdakı proqramlar müxtəlif çevrə və qövslərin qurulmasını təmin edir.

```
10 SCREEN 1,0
20 R=96
```

```
30 FOR SA=0 TO 6.28 STEP 0.4188
40 CIRCLE (128,96),R,15,SA,0
50 R=R-6: NEXT SA
60 FOR J=1 TO 300:NEXT J
70 CLS
80 R=96
90 FOR EA=0 TO 6.28 STEP 0.4188
100 CIRCLE(128,96),R,15,0,EA
101 R=R-6:NEXT EA
102 END
```

```
10 SCREEN 1,0
20 FOR I=1 TO 10
30 X=RND(1)*224+16
40 Y=RND(1)*160+16
50 FOR R=5 TO 20 STEP 2
60 CIRCLE(X,Y), R
70 NEXT R: NEXT I
80 END
```

```
10 C=5
20 SCREEN 1,0
30 FOR Y=40 TO 160 STEP 40
40 FOR X=40 TO 216 STEP 40
50 FOR EA=0 TO 6.29 STEP 0.4188
60 CIRCLE (X,Y),19,C,0,-EA
70 NEXT EA: NEXT X
80 C=C+2
90 NEXT Y
100 END
```

Aşağıdakı proqram isə ellipslərin qurulmasını təmin edir:

```
10 SCREEN 1,0
20 FOR I=80 TO 24 STEP -4
30 CIRCLE(128,96),96, , , , 10/I
40 CIRCLE(128,96),96, , , , I/10
50 NEXT I
60 END
```

6) **PAINT(X,Y),N,N1** operatoru yerinə yetirilərkən ekran

(X,Y) koordinatlı nöqtədən başlayaraq müntəzəm olaraq bütün istiqamətlərdə **N** nömrəli rənglə rənglənməyə başlayır və bu proses ekranda **N1** nömrəli rəngli nöqtəyə çatana qədər davam etdirilir. Məsələn: **300 PAINT(70,90),1,3**. Aşkardır ki, ekranın hər hansı bir hissəsini rəngləmək üçün, bu hissə əvvəlcədən **N1** nömrəli rəngi olan xətt ilə əhatə olunmalıdır.

Misal. Aşağıdakı proqramlar ekranın qeyd edilmiş hissələrinin verilmiş rənglə rənglənməsini təmin edir.

```
10 SCREEN 1,0
20 FOR C=15 TO 1 STEP -1
30 CIRCLE(128,96),C*7,C
40 PAINT(128,96),C,C
50 NEXT C
60 END
```

```
10 SCREEN 1,0
20 LINE(10,10)-(20,20),8,BF
30 LINE(180,170)-
(200,190),8,BF
40 CIRCLE (128,96),70,15
50 PAINT (10,10),15,15
60 END
```

Səs operatorları. Fərdi kompüterlərdə səs operatorlarının köməyi ilə səs və musiqi vermək olur. Beyzikdəki ən sadə səs operatoru **BEEP** operatorudur. Bu operator fərdi kompüterlərdə olan dinamiklərə 800 *hers* tezlikli və 0,25 *san* müddətli səs siqnalı verir. Səsin tezlik və uzunluğunu idarə etməyə imkan verən digər operator aşağıdakı şəkildədir:

SOUND W,T

Harada **W** parametri 37-dən 32767 hersədək tezliyi verir, T parametri isə səs siqnalının uzunluğunu təyin edib, 0-dan 65535-dək qiymətlər ala bilər (hər bir vahid 55 milli saniyəyə bərabərdir). Məsələn:

10 SOUND 532.25, 16.55

operatoru notlar içində «do» notunun birinci oktavasına uyğun gələn səs yaradır.

Səs siqnalı səslənən zaman EHM, **SOUND** operatorunun tam yerinə yetirilməsini gözləmir və proqramın işi davam etdirilir. Proqramda ikinci **SOUND** operatoru olarsa və əgər birinci **SOUND** operatoru hələ işini bitirməyibsə, proqramın yerinə yetirilməsi səs kanalı boşalana qədər dayandırılır. Ümumiyyətlə, səs siqnalını, ixtiyari vaxt (**T=0**) sıfır uzunluqlu **SOUND** operatoru verməklə kəsmək olar. Tezliyi15000 hersdən çox olan səs siqnalını insan qulağı eşitmədiyindən **W>15000** parametri olan **SOUND** operatorları fasilələr (pauzalar) verəcəkdir. Misallar: Aşağıdakı proqramlar müxtəlif səs effektləri yaradır:

```
10 FOR I=0 TO 10: SOUND I,0 : NEXT I
20 SOUND 5,239
30 SOUND 6,30
40 SOUND 10,15000
50 FOR I=1 TO 250:SOUND 4,100:NEXT I
60 FOR I=1 TO 20000:NEXT I
70 SOUND 5,255
80 END
```

```
10 DATA 62,2,60,2,60,2,0
20 DATA 56,16,16,16,120,30,13
30 FOR I=0 TO 13:READ A:SOUND I,A:NEXT I
40 FOR I= 1 TO 1000:NEXT I
50 X=RND(1)*250
60 IF x<30 THEN 50
70 Y=RND(1)*10+1
80 SOUND 0,X:SOUND 1,Y
90 SOUND 2,X:SOUND 3,Y
100 SOUND 4,X+5:SOUND 5,Y
110 SOUND 13,13
120 END
```

SOUND səs operatoru ilə yanaşı aşağıdakı səs operatorundan da istifadə olunur:

PLAY «musiqi sətri»

burada «musiqi sətri» ümumi halda oktavanın nömrəsindən (0-dan 6-dək; birinci oktava 3 nömrəsinə malikdir) və notlar ardıcıllığından ibarətdir.

Musiqi sətrin elementləri **PLAY** operatorunun alt əmri adlanır. Onların hər birî musiqi savadının qaydalarına uyğun olaraq müxtəlif məqsəd daşıyırlar. Xüsusi halda notları göstərmək üçün uyğun olaraq **C, D, E, F, G, A, B** hərflərindən istifadə edilir. Əgər not diezlə verilirsə, onda uyğun hərfdən sonra **+** (plyus) və ya **#** işarəsi, bimol ilə verilirsə, hərfdən sonra - (minus) işarəsi qoyulur. Oktava, **O** hərfi ardınca nömrəsi ilə göstərilir, məsələn, birinci oktava 03 kimi işarələnir.

Ümumiyyətlə yeddi mümkün oktavada 84 səs vardır ki, onları **PLAY** operatorunda göstərmək üçün oktavanın nömrəsi və notanın uyğun hərf işarəsi əvəzinə **N** alt əmri və 0-dan 84-dək olan notların sıra nömrəsi ilə (0 nömrəsi fasilə bildirir) vermək olar.

Beləliklə, məsələn, «do» notasının birinci oktavasından başlayaraq bütün səs sırasını vermək üçün aşağıdakı operatorların ixtiyari birindən istifadə etmək olar:

```
10 PLAY "03CC # DD # EFF # GG # AA # B"
10 PLAY "N37 N38 N39 N40 N41 N42 N43 N44 N45
   N46 N47 N48"
```

Misallar.

Aşağıdakı proqramlar müxtəlif səs effektləri yaradır:

```
10 A$="M500 O4L4DF #ARAF #D"
20 FOR I=1 TO 8
30 DATA S1,S4,S8,S10,S11,S12,S13,S14
40 READ S$
50 PLAY S$
60 PLAY A$
70 FOR J=1 TO 2000:NEXT J
80 NEXT I
90 END
10 DATA C,A,E,F,G,B,D,G,E,C
20 PLAY "T25505L16"
30 FOR I=1 TO 10
40 READ A$
50 PLAY A$
60 NEXT I:END
```

4.15. QBASIC dilinin proqramlaşdırma mühiti və sistemi

Əvvəlcə proqramlaşdırma mühitləri və sistemləri haqqında ümumi anlayışlar verək. Translyator – istifadəçinin hər hansı bir proqramlaşdırma dilində hazırladığı proqramı EHM-in daxili dilinə çevirən proqramdır. Translyator əməliyyat sisteminin idarəsi altında işləyir. Praktikada müxtəlif iş prinsipləri olan iki cür translyatordan istifadə edilir. Birinci tip translyator verilən proqramın hər bir operatorunu növbə ilə maşın dilinə çevirir və onları dərhal yerinə yetirir. Belə traslyatorlar interpretatorlar adlanır. İkinci tip translyatorlar isə kompilyatorlar adlanır. Bu proqram isə interpretatordan fərqli olaraq əvvəlcə verilmiş proqramı tamamilə maşın dilinə çevirir, onun doğruluğunu yoxlayır, bu proqrama standart funksiyaların ($\sin x, \cos x, \ln x$ və s.) alt proqramlarını və digər alt proqramları qoşur. Nəticədə ümumi yükləmə proqramı alınır. Bundan sonra əməliyyat sistemi yükləmə proqramını işə salır və nəticədə qoyulmuş məsələnin həlli təyin edilir.

İstifadəçinin birbaşa translyatorla işləməsi əlverişli olmur. Buna görə də əməliyyat sistemlərində olduğu kimi burada da istifadəçinin translyatorla ünsiyyətini sadələşdirən örtük proqramlardan istifadə edilir. Bu cür proqramlar və ya proqramlar kompleksi porqramlaşdırma mühiti adlanır. Bu mühitə translyator, örtük-proqram, proqram mətnlərinin daxil edilməsi və redaktə edilməsi üçün ekran redaktoru və s. daxildir.

Proqramlaşdırma sistemi-müəyyən tip EHM-də konkret proqramlaşdırma dilində verilən proqramların yaradılması və işlədilməsini təmin edən vasitələr kompleksidir. Bura adətən, proqramlaşdırma dilinin müəyyən bir versiyası, translyator və ya proqramlaşdırma mühiti daxildir. Məsələn, QuickBasic proqramlaşdırma sisteminə proqramlaşdırma mühiti, nümayişetdirmə proqramı, öyrədici proqram, alt proqramlar kitabxanası, operativ yada salma sistemi və s. aiddir.

İndi isə QBASIC proqramlaşdırma mühiti və sisteminə baxaq. Bu sistemə BASIC dilinin versiyası, proqramlaşdırma mühiti aiddir. Proqramlaşdırma mühiti qeyd etdiyimiz kimi proqramın hazırlanması, onunla işlə bağlı bütün əməliyyatlar kompleksini ye-rinə yetirməyə imkan verir və aşağıdakılardan ibarətdir: interpretator (QuickBASIC proqramlaşdırma sisteminə həm də kom-

pilyator da daxildir), mühitin işini menyu vasitəsilə idarə etməyə imkan verən örtük proqramı, proqram mətnlərini daxil etmək və redaktə etməyə imkan verən mətnlərin intellektual redaktoru, proqramların sazlanmasını təmin edən sazlayıcı proqram və s.

Proqramlaşdırma mühiti, ümumi həcmi 325 *Kbayt* olan, qbasic.exe, qbasic.hlp və qbasic.ini fayllarından ibarətdir. Onlardan əsası birinci fayldır. Mühit MS DOS 3.3 və daha yüksək versiyalı əməliyyat sistemlərinin idarəsi ilə işləyə bilir. QBASIC-in proqramlaşdırma mühitinə keçmək üçün MS DOS-la işləyərkən C:\ DOS\ qbasic.exe əmrini vermək lazımdır. Nəticədə ekrana mühitin pəncərəsi veriləcəkdir. Hər hansı F faylında olan konkret proqramla iş üçün isə əmr sətrində C:\ DOS\ qbasic.exe < F faylının tam adı > vermək lazımdır. Mühitlə işi sona çatdırmaq üçün Alt düyməsini sıxıb, menyuya keçib, *File* (*Fayl*) bölməsinin *Exit* (*Çıxış*) əmrini seçmək lazımdır.

QBASIC məhiti pəncərəsinin əsas elementləri: işçi pəncərə, dərhal yerinə yetirilmə pəncərəsi və yada salma sətri.

1) İşçi pəncərə ekranın əsas hissəsini tutur və mühitin əsas funksiyalarının yerinə yetirilməsini təmin edir. Onun tərkibinə menyu və işçi sahə daxildir. Menyu pəncərənin birinci sətrində yerləşir və 8 bölmədən: File (Fayl), Edit (Redaktə etmək), View (Görünüş), Search (Axtarış), Run (Yerinə yetirmək), Debug (Sazlama), Options (Parametrlər), Help (Kömək) ibarətdir. Hər menyu bölməsi proqramlaşdırma mühitində proqramla iş üçün nəzərdə tutulmuş əmrlər ardıcıllığından ibarətdir. Menyu ilə iş zamanı siçandan və klaviaturadan istifadə edilir. Menyuya daxil olmaq üçün Alt, çıxmaq üçün isə Esc düymələrindən istifadə olunur. İşçi sahə menyudan aşağıda yerləşir və mühit pəncərəsinin əsas hissəsini tutur. Sahənin yuxarı sətrində, pəncərəyə yüklənmiş faylın adı ilə üst-üstə düşən pəncərə adı göstərilir. Pəncərənin başlanğıc adı "Untitled" ("Adsız") olur. İşçi sahəyə proqram mətni yüklənir və onun üzərində iş aparılır. Onun daxilində kursor olduqda işçi sahə işə hazır hesab edilir.

2) Dərhal yerinə yetirilmə "Immediate" ("Dərhal") pəncərəsi işçi pəncərədən aşağıda yerləşir. Bu pəncərəyə yazılan operatorlar (BASIC dilinin) *Enter* düyməsinin sıxılması ilə dərhal yerinə yetirilirlər. Bu pəncərəyə daxil olmaq (çıxmaq) üçün F6 düyməsini sıxmaq lazımdır. Bu zaman kursor bu pəncərəyə çıxarılır.

3) Yada salma sətri pəncərənin axırıncı sətrinin sol hissəsini tutur. Burada funksional düymələrin cari təyinatları verilir. Həmin sətrin sağ tərəfində isə kursorun aktiv pəncərədəki cari mövqeyinin koordinatları verilir. Qeyd edək ki, mühitdə göstərilən pəncərələrdən başqa, üçüncü pəncərə çıxış pəncərəsindən də istifadə edilir. Bu pəncərəyə proqramın tələb etdiyi başlanğıc verilənlər klaviaturadan daxil edilir və proqramın yerinə yetirlmə nəticələri çıxarılır. Çağırış zamanı bu pəncərə bütün ekranı tutur və digər pəncərələri örtür. Proqramın yerinə yetirilməsi zamanı çıxış pəncərəsi proqram tərəfindən tələb olunduqca çağırılır. Bu pəncərənin çağırılması (bağlanması) üçün *F4* düyməsindən istifadə edilir. Zamanın bir anında yalnız bir pəncərə aktiv ola bilər. Aktiv olan pəncərəni bütün ekran boyunca açmaq üçün *Ctrl+F10* kombinasiyasından istifadə edilir. Bu düymələr pəncərəni əvvəlki vəziyyətinə də qaytara bilir.

QBASIC proqramlaşdırma mühitində əsas əməliyyatlara baxaq:

1) Mühitin işçi sahəsinə fayldan proqram çağırmaq üçün File (Fayl) menyu bölməsindəki Open (Açmaq) əmrini vermək lazımdır. Nəticədə Open (Açmaq) dialoq lövhəsi ekrana verilir. Burada File Name (Faylın adı) sahəsində çağrılan fayla gedən yolu və onun adını verib, Enter düyməsini sıxmaq lazımdır. Nəticədə fayldakı proqram mətni işçi sahəyə çıxarılacaqdır.

2) İşçi sahədən proqramı fayla yazmaq üçün iki variant var:

a) Əgər proqramda müəyyən dəyişikliklər aparıb, onu əvvəlki fayla yenidən yazmaq tələb olunursa, File (Fayl) menyu bölməsinin Save (Saxlamaq) əmrini vermək lazımdır. Bu zaman fayldakı köhnə mətn silinir və yeni proqram mətni ora yazılır.

b) Əgər proqram mətnini onun olduğu fayldan fərqli başqa fayla yazmaq tələb olunursa, File (Fayl) bölməsinin Save As (Necə saxlamaq) əmrini vermək lazımdır. Nəticədə ekrana verilən dialoq lövhəsindəki File Name (Faylın adı) sahəsində yeni fayla gedən yolu və onun yeni adını verib, Enter düyməsini sıxmaq lazımdır.

3) Mühitdə proqramı yerinə yetirmək üçün proqram mətnini fayldan işçi sahəyə çağırdıqdan sonra Run (Yerinə yetirmək) menyu bölməsinin Start (Başlamaq) əmrini vermək lazımdır. Nəticədə proqram yerinə yetirilməyə başlanır. Bu zaman əgər proq-

ramda **INPUT** operatoru varsa, proqramın yerinə yetirilməsi müvəqqəti dayandırılır, ekrana çıxış pəncərəsi çağırılır və EHM verilənlərin daxil edilməsini gözləyir. Pəncərədəki "?" işarəsindən sonra klaviaturadan tələb olunan verilənləri bir-birindən vergüllə ayırmaqla daxil edib, sonda *Enter* düyməsini sıxmaq lazımdır.

İndi isə proqram mətninin daxil edilməsi və redaktə edilməsi qaydalarına baxaq. Proqram mətni mühit redaktorunun intellektual iş rejimində daxil edilir. Bu halda redaktor proqram mətni daxil edildikcə, sintaksis yoxlama aparılmasını təmin edir. Bu iş rejiminin daxil edilməsi (çıxarılması) üçün *Options (Parametrlər)* menyu bölməsinin *Sintax Checking* (*Sintaksis yoxlama)* əmrini seçmək lazımdır. İşçi sahənin mətnin daxil edilməsinə (redaktə edilməsinə) hazır olması, burada kursorun olması ilə müəyyən olunur. Ekran iki rejimdə işləyə bilir:

1) Daxil etmə rejimi – bu rejimdə yeni daxil edilən simvollar artıq daxil edilmiş simvolların arasına daxil edilir, bu rejimdə kursor « – » formasında olur;

2) Əvəz etmək rejimi – bu rejimdə yeni daxil edilən simvollar artıq daxil edilmiş simvolları əvəz edir, burada kursor « | » formasında olur. Rejimlər INS düyməsi ilə dəyişdirilə bilir. Yeni proqram mətnini daxil etmək üçün *File (Fayl)* bölməsinin *New (Yaratmaq)* əmrini vermək lazımdır. Nəticədə ekranda artıq fayla yazılmış proqram mətni varsa, ekran təmizlənir və yeni proqram daxil etmək üçün şərait yaranır. Əgər bu proqram mətni fayla yazılmayıbsa, onda ekrana bu əməliyyatı yerinə yetirmək təklifi ilə dialoq lövhəsi verilir. Mətni faylda saxlamaq üçün bu lövhədəki *Yes (Hə)* düyməsini sıxıb, yeni açılan *Save As (Necə saxlamaq)* dialoq lövəsinin daxil etmək sahəsində faylın tam adını verib, *Enter* düyməsini sıxmaq lazımdır. Proqram mətninin saxlanılmasından imitna etdikdə isə *No* (*Yox*) düyməsini sıxmaq lazımdır. Hər iki halda ekran yeni proqram mətninin daxil edilməsi üçün təmizlənir.

V FƏSİL

TURBO PASCAL ALQORİTMİK DİLİ.

5.1. Dilin əlifbası. Verilənlər. Proqramın strukturu

Dilin əlifbası hərf, rəqəm və xüsusi simvollardan ibarətdir. Hərflər – latın əlifbasının böyük (A-Z) və kiçik (a-z) hərfləri; rəqəmlər – on ərəb (0-9) rəqəmləri və 0,1,...,9,A,B,C,D,E,F onaltılıq say sisteminin rəqəmləri; xüsusi simvollar **+ - * / = > < . , ' : ; [ ] ( ) { } ^ @ $ #**.

Xüsusi simvollara eləcə də aşağıdakı simvol cütləri də (onları probellə ayırmaq olmaz) aiddir:

:= (mənimsətmə işarəsi), **>=** (böyük bərabər), **<>** (fərqli), **<=** (kiçik bərabər), **(* *)** **({}** işarəsi ilə eynigüclü olan qeyd məhdudlaşdırıcısı) **(. .)** **([ ]** işarəsinin ekvivalenti). Burada probel boş yer işarəsi də xüsusi yer tutur. Bu simvol identifikator, sabit ədəd, işçi sözlər üçün məhdudlaşdırıcı kimi nəzərdən keçirilir. Birbirinin ardınca verilən bir neçə probel işarəsi bir işarə kimi qəbul edilir (sətir sabitləri istisna olmaqla).

Turbo Pascal dilində aşağıdakı işçi sözlərdən istifadə olunur: **and, asm, array, begin, case, const, constructor, destructor, div, do, downto, else, end, file, for, function, goto, if, implementation, in, inline, interface, label, mod, nil, not, object, of, or, packed, procedure, program, record, repeat, set, shl, shr, string, then, to, type, unit, until, uses, var, while, with, xor**.

İşçi sözlərdən başqa məqsədlər üçün istifadə edilə bilməz. Dil nöqteyi-nəzərdən onlar vahid simvol hesab edilirlər. Dildə standart elanlar kimi aşağıdakı işçi sözlərdən də istifadə edilir: **absolute, assembler, external, far, forward, interrupt, near, private, virtual**.

Turbo Pascal dilində identifikator – sabit, dəyişən, nişan, tip, obyekt, prosedur, funksiya, modul, proqram və yazılışlardakı sahə adlarıdır. İdentifikator – hərflə başlayan ixtiyari hərflər və rəqəmlər ardıcıllığıdır. Turbo Pascal-da altdan xətt çəkmə («_»)

işarəsi də hərflərə aiddir. İdentifikator ixtiyari uzunluqlu ola bilər, lakin burada yalnız birinci 63 simvol nəzərə alınacaqdır. İdentifikatorda probeldən və dilin xüsusi simvollarından istifadə edilə bilməz. *Məsələn*: **x**, **y**, **ALPHA**, **_beta**, **_1**, **z12**, **max**, **MIN** və s.

Sabitlər kimi Turbo Pascal-da tam, həqiqi, onaltılıq ədədlər, məntiqi sabitlər, simvollar, sətirlər, çoxluq konstruktorları və qeyri-müəyyən göstərici əlaməti – NIL istifadə edilə bilər. Tam ədədlər adi qayda ilə işarə və ya işarəsiz yazılır və -2147483648-dən +2147483647-dək qiymətlər ala bilər. Həqiqi ədədlər işarə və ya işarəsiz onluq nöqtə ilə və ya eksponensial hissə ilə yazılırlar. Eksponensial hissə **e** (**E**) simvolu ilə başlayır, ondan sonra «+» və ya «-» işarəsi gələ bilər və onluq tərtib verilir. Məsələn, **3.5E5** (yəni $3,5*10^5$), **-17e-3** (yəni $-17*10^{-3}$) və s. Onaltılıq ədədlər, qarşısında $ işarəsi olan onaltılıq say sisteminin ədədləridir. Onların dəyişmə diapazonu $00000000-dan $FFFFFFFF-ə qədərdir. Məntiqi sabit – **FALSE** (yalan) və ya **TRUE** (doğru) sözlərindən biridir. Simvol sabitlər klaviaturanın apastrof işarələri arasına alınmış ixtiyari simvollarıdır. Məsələn, 'z', 'A', '9', və s. Sətir sabiti apastrof işarələri arasına alınan ixtiyari simvollar ardıcıllığıdır. Məsələn, 'Turbo Pascal alqoritmik dili'. Sətirdə heç bir simvol verilməzsə, belə sətir boş sətir adlanır. Çoxluq konstruktoru – kvadrat mötərizə daxilində verilən çoxluq elementlərinin siyahısıdır. Məsələn, **[1,2,3..8,12]**, **[true]**, **[ ]**, **[blue,red]** və s. Qeyd edək ki, standart Pascal dilindən fərqli olaraq Turbo Pascal dilində sabitlər kimi elementləri əvvəlcədən elan edilmiş sabitlər, tip adları, obyektlər və funksiyalar olan ixtiyari ifadələrdən də istifadə edilə bilər.

Proqram vahidi başlıqdan, təsvirlər bölməsindən, operatorlar bölməsindən və proqramın sonunu bildirən nöqtədən ibarətdir:

program <proqramın adı> – başlıq
uses – modullar bölməsi
label – nişanlar bölməsi
const – sabitlər bölməsi
type – tiplər bölməsi
var – dəyişənlər bölməsi

procedure (**function**) – alt proqramlar bölməsi
begin
<operatorlar bölməsi>
end.

Başlıqda proqram vahidinə verilən ad yerləşdirilir, ad üzərinə identifikatorlar üçün təyin edilmiş şərtlər qoyulur. Turbo Pascal dilində başlıq verilməyə də bilər. Ümumiyyətlə, proqramın bölmələrində heç bir təsvir və yerinə yetirilən operatorlar verilməyə də bilər. Standart Pascal dilindən fərqli olaraq Turbo Pascal dilində **label**, **const**, **type**, **var** bölmələri bir-birinin ardınca ixtiyari qaydada verilə bilər və təsvirlər bölməsində ixtiyari sayda ola bilər. Təsvirlər bölməsində operatorlar bölməsində istifadə edilən bütün identifikatorlar təsvir edilməlidir. Burada modulların interfeys hissələrində təyin edilən və posedurların (funksiyaların) qlobal identifikatorları istisna təşkil edir. Əgər proqram vahidində hər hansı bir modulun identifikatorundan istifadə edilirsə, onda **uses** bölməsində bu modulun adı elan edilməlidir. Burada da **system** modulu istisna təşkil edir. Çünki bu modul əvvəlcədən elan edilmiş hesab edilir. Təsvirlər bölməsində tiplərin, obyektlərin, sabitlərin, dəyişənlərin identifikatorları, həmçinin nişanlar, prosedur və funksiyalar elan edilir. Tip və obyektlərin təsviri – **type** bölməsində, sabitlərin təsviri – **const**, dəyişənlərin təsviri – **var**, nişanların təsviri isə **label** bölməsində yerinə yetirilir. *Məsələn*,

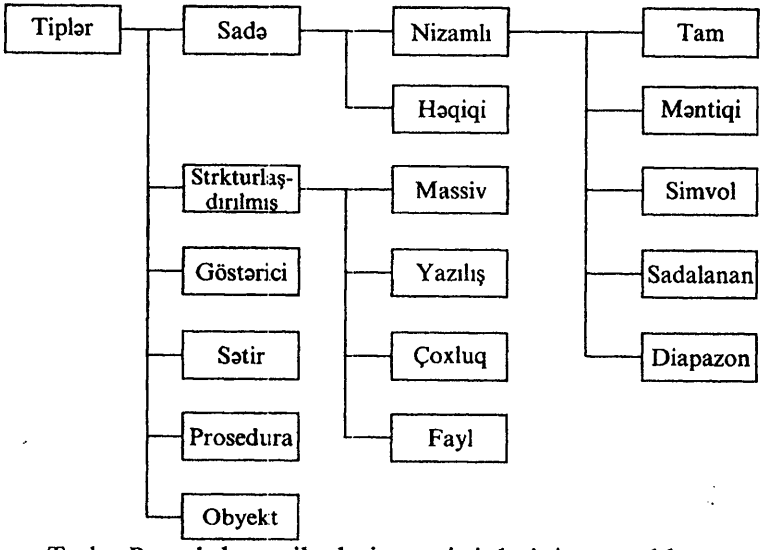
```
type b=array[1..10] of real;
       a=set of '0'..'9';
       c=string[100];
const n=100; eps=1e-3;
var x,y:real; s1:c; k:b;
label 1,2,lb1,lb2;
```

Turbo Pascal dilində proqram vahidində gedən prosesləri izah etmək üçün şərhlərdən geniş istifadə edilir. Burada şərh fiqurlu mötərizələr daxilində gətirilən ixtiyari simvolların istənilən ardıcıllığıdır. Şərhlər proqramın ixtiyari yerində verilə bilər və proqramın yerinə yetirilməsinə heç bir təsir göstərmir. Şərh məhdudlaşdırıcısı kimi fiqurlu mötərizələrlə yanaşı (* şərh *) işarəsindən də istifadə etmək olar. Məsələn, {Daxil etmək}, (* Cavab*)

və s. Qeyd edək ki, proqram mətnində hər sətrin sonunda qoyulan nöqtə vergül işarəsi sətrin sonunu bildirir.

5.2. Verilənlərin tipləri. Tiplərin uyğunluğu və çevrilməsi

İxtiyari verilənlər, yəni sabitlər, dəyişənlər, funksiya qiymətləri və ya ifadələr, Turbo Pascal-da öz tipləri ilə xarakterizə olunurlar. Tip bu və ya digər obyektin ala biləcəyi qiymətlər çoxluğunu və bu obyektlərə tətbiq oluna biləcək əməliyyatlar çoxluğunu təyin edir. Bundan əlavə tip, verilənlərin EHM yaddaşında daxili ifadə formatını müəyyən edir. Turbo Pascal-da verilənlərin tiplərinin strukturunu aşağıdakı kimi ifadə etmək olar:



Turbo Pascal-da verilənlərin yeni tiplərinin yaradılması mexanizmi nəzərdə tutulduğundan, proqramda istifadə edilən tiplərin sayı ixtiyari qədər böyük ola bilər.

Sadə tiplərə nizamlı və həqiqi tiplər aiddir. Nizamlı tiplərin hər biri sonlu sayda mümkün qiymətə malikdir. Bu qiymətləri müəyyən qaydada nizamlamaq olur (tipin adı da buradan irəli gəlir) və deməli, onların hər birinə qarşı hər hansı bir tam ədəd-qiymətin sıra nömrəsini qoymaq olur. Həqiqi tiplər də həqiqi ədədin daxili ifadə formatı ilə müəyyən edilən sonlu sayda qiymətə malikdir. Lakin həqiqi tiplərin ala biləcəyi qiymətlərin sayı o qə-

dər böyükdür ki, onların hər birinə qarşı tam ədəd (onun nömrəsini) qoymaq mümkün deyil.

Nizam tipinə tam, məntiqi, simvol, sadalanan və diapazon tipləri aiddir. Bütün bu tiplərə **ord(x)** funksiyası tətbiq edilə bilər, həmin funksiya **x** ifadəsinin qiymətinin sıra nömrəsini təyin edir. Tam tiplər üçün bu funksiya **x**-ın aldığı qiyməti verir, yəni **ord(x)=x**, burada **x**-ixtiyari tam tipə aiddir. Bu funksiya məntiqi, simvol və sadalanan tiplərə tətbiq olunduqda, məntiqi tip üçün 0-1, simvol tip üçün 0-255, sadalanan tip üçün 0-65535 diapazonunda müsbət tam ədəd alınır. Nizamlı tiplərinə həmçinin **pred(x)** – nizam tipinin əvvəlki qiymətini (**ord(x)-1** sıra nömrəsinə uyğun) təyin edən, yəni **ord(pred(x))=ord(x)-1** funksiyasını və succ(x) – nizam tipinin sonrakı qiymətini (**ord(x)+1** sıra nömrəsinə uyğun) təyin edən, yəni **ord(succ(x))=ord(x)+1** funksiyasını tətbiq etmək olar.

Tam tiplərə aşağıdakılar aiddir:

Adı	*Baytlarla uzunluğu*	*Qiymətlər diapazonu*
Byte	1	0...255
ShortInt	1	-128 ... +127
Word	2	0 ... 65535
Integer	2	-32768 ... +32767
LongInt	4	-2147483648 ... +2147483647

Tam tiplərə tətbiq olunan prosedur və funksiyalar aşağıdakılardır:

Funksiya	*Təyinatı*	*Arqumentin tipi*	*Nəticənin tipi*
abs(x)	**x**-in mütləq qiymətinin təyini	İxtiyari tam tip	Arqumentin tipi
chr(x)	Simvolu onun **x** kodu üzrə təyin edir	**Byte**	**Char**
dec(x[,i])	**x**-in qiymətini **i** qədər, **i** verilmədikdə bir vahid azaldır	İxtiyari tam tip	Arqumentin tipi

inc(x[,i])	**x**-in qiymətini **i** qədər, **i** verilmədikdə bir vahid artırır	İxtiyari tam tip	Arqumentin tipi
hi(x)	**x**-in yüksək baytını təyin edir	**Integer Word**	**Byte**
lo(x)	**x**-in aşağı baytını təyin edir	**Integer Word**	**Byte**
odd(x)	**x**-in tək ədəd olduqda **True** qiymətini alır	**LongInt**	**Boolean**
random(x)	**0** ... **x-1** diapazonunda bərabər paylanmış təsadüfi ədədi təyin edir	**Word**	**Word**
sqr(x)	**x**-in kvadratını təyin edir	İxtiyari tam tip	Arqumentin tipi
swap(x)	Sözdə yüksək və aşağı baytların yerini dəyişdirir	**Integer Word**	**Integer Word**
randomize	Təsadüfi ədədlər generatorunun aktivləşdirilməsi	–	–

Qeyd edək ki, tam tipli parametrləri olan prosedur və funksiyalardan istifadə edərkən nəzərə almaq lazımdır ki, əgər **Word** tipindən istifadə edilirsə, burada **Byte** tipindən də istifadə oluna bilər (əksinə yox), eləcə də **Integer** tipi **LongInt**-ə daxildir, **ShortInt** tipi isə **Integer**-ə daxildir. Tam ədədlərlə aparılan əməliyyatların nəticəsi əməliyyatda iştirak edən verilənlərin tipi ilə eyni olur. Onların tipi müxtəlif olduqda isə nəticənin tipi buradakı ən yüksək qiymətlər diapazonu olan tiplə üst-üstə düşür. Turbo Pascal-da əvvəlcədən elan edilmiş **Integer** tipli sabit **MaxInt** 32767 qiymətini alır. Məntiqi tipin qiyməti əvvəlcədən elan edilmiş **False** (yalan) və ya **True** (doğru) sabitlərindən hər hansı biri ola bilər. Onlar üçün aşağıdakı qaydalar doğrudur:

```
ord(False)=0;
ord(True)=1;
```

```
False<True;
succ(False)=True;
pred(True)=False.
```

Simvol tipin qiymətləri EHM-dəki bütün simvollar çoxluğudur. Hər bir simvola 0 ... 255 diapazonunda bir tam ədəd uyğun gəlir. Bu ədəd simvolun daxili ifadə kodudur və **ord** funksiyası ilə təyin edilə bilir. Kodlaşdırma üçün ASCII (American Standard Code for Information Interchange – informasiya mübadiləsi üçün Amerika standart kodu) kodundan istifadə edilir. **Char** tipinə münasibət əməliyyatları və aşağıdakı funksiyalar tətbiq edilə bilir:

chr(x) – **char** tipli funksiya olaraq byte tipli **x** ifadəsini simvola çevirir.

upcase(x)-də **char** tipli funksiya olub, char tipli **x** arqumentini kiçik latın hərfi olduqda onu uyğun böyük latın hərfinə çevirir, əks halda isə **x** simvolunun özünü qaytarır.

Sadalanan tip, onun ala biləcəyi qiymətlərin sadalanması ilə verilir. Hər bir qiymət müəyyən identifikatorla adlandırılıb, yumru mötərizələrlə məhdudlaşdırılan siyahıda verilir. *Məsələn*,

```
type c=(qirmizi,sari,qara,boz);
var
ay:(yan,fev,mart,apr,may,iyun,iyul,avq,
sen,
okt,noy,dek);
```

Sadalanan tipdə siyahıdakı birinci elementin sıra nömrəsi sıfır, ikinci elementin nömrəsi bir və s. olur. Sadalanan tipin maksimal gücü 65536 qiymətdir. *Məsələn*,

```
type gaz=(c,o,n,f);
       metal=(fe,co,na,cu);
var g1,g2,g3:gaz; m1,m2,m3:metal;
```

Diapazon tipi özünün baza tipinin alt çoxluğudur. Baza tipi diapazon tipdən başqa ixtiyari nizam tipi ola bilər. Diapazon tipi baza tipi daxilində öz qiymətlərinin sərhədləri ilə verilir: <min. qiymət> .. <maks. qiymət>. Burada <min. qiymət> - diapazon tipinin minimal qiyməti, <maks. qiymət> isə maksimal qiymətidir. *Məsələn*,

```
type k1='0'..'9'; k2=1966..2007;
```

və ya

```
var date:1..31; month:1..12;
k3:'A'..'Z';
```

və s.

Burada «..» simvolu bir simvol kimi qəbul edildiyindən nöqtələr arasında probel qoyula bilməz və diapazonun sol sərhəddi sağ sərhəddini aşa bilməz. Diapazon tiplərlə iş üçün aşağıdakı funksiyalar nəzərdə tutulub:

high(x) – funksiyası **x**-in aid olduğu diapazon tipinin maksimal qiymətini verir,

low(x) - funksiyası **x**-in aid olduğu diapazon tipinin minimal qiymətini verir. *Məsələn,*

```
var k:integer;
begin
        writeln(low(k),'...',high(k))
end.
```

Nəticədə alarıq: -32768 ... 32767.

Həqiqi tiplərə aşağıdakılar aiddir:

Adı	*Baytlarla uzunluğu*	*Əhəmiyyətli rəqəmlərin sayı*	*Qiymətlər diapazonu*
Real	6	11 – 12	$2.9 \cdot 10^{-39} \ldots 1.7 \cdot 10^{38}$
Single	4	7 – 8	$1.5 \cdot 10^{-45} \ldots 3.4 \cdot 10^{38}$
Double	8	15 – 16	$5 \cdot 10^{-324} \ldots 1.7 \cdot 10^{308}$
Extended	10	19 – 20	$3.4 \cdot 10^{-4932} \ldots 1.1 \cdot 10^{4932}$
Comp	8	19 – 20	$-2 \cdot 10^{63}+1 \ldots +2 \cdot 10^{63}-1$

EHM-də mütləq dəqiqliklə ifadə olunan nizam tiplərindən fərqli olaraq həqiqi tiplərin qiyməti ixtiyari ədədi yalnız müəyyən sonlu dəqiqliklə təyin edir. Burada hər şey həqiqi ədədin daxili ifadə formatından asılı olur. Yuxarıda verdiyimiz birinci beş həqiqi tip bir-birindən qiymətlər diapazonu və dəqiqliyinə görə fərqlənir. Kəsr və eksponensial hissəsi olmayan **Comp** tipi faktiki

olaraq 19-20 əhəmiyyətli onluq rəqəm saxlayan, işarəli böyük tam ədəddir. Lakin eyni zamanda ifadələrdə **Comp** tipi digər ixtiyari həqiqi tiplərlə uyğunlaşır və onun üzərində həqiqi tiplərə uyğun bütün əməliyyatlar aparıla bilər. **Comp** tipinin ən əlverişli tətbiq sahəsi mühasibat işləridir, belə ki, burada pul kütləsi qəpik və ya sentlərlə ifadə olunur və onlar üzərindəki əməliyyatlar kifayət qədər uzun tam ədədlərə gətirib çıxarır.

Həqiqi tiplərə tətbiq olunan funksiyalar aşağıdakılardır:

Funksiya	*Təyinatı*	*Arqumentin tipi*	*Nəticənin tipi*
abs(x)	**x**-in modulunu təyin edir	İxtiyari həqiqi tip	Arqumentin tipi
arctan(x)	**x**-in arktangensini (radianlarla) təyin edir	İxtiyari həqiqi tip	Arqumentin tipi
cos(x)	**x**-in kosinusunu (radianlarla) təyin edir	İxtiyari həqiqi tip	Arqumentin tipi
exp(x)	**x**-in eksponentasını təyin edir	İxtiyari həqiqi tip	Arqumentin tipi
frac(x)	**x**-in kəsr hissəsini təyin edir	İxtiyari həqiqi tip	Arqumentin tipi
int(x)	**x**-in tam hissəsini təyin edir	İxtiyari həqiqi tip	Arqumentin tipi
ln(x)	**x**-in natural loqarifmini təyin edir	İxtiyari həqiqi tip	Arqumentin tipi
pi	$\pi = 3.141592 \ldots$ ədədini verir	–	İxtiyari həqiqi tip
random	[0,1) intervalında təsadüfi ədəd təyin edir	–	İxtiyari həqiqi tip
sin(x)	**x**-in sinusunu (radianlarla) təyin edir	İxtiyari həqiqi tip	Arqumentin tipi
sqr(x)	**x**-in kvadratını təyin edir	İxtiyari həqiqi tip	Arqumentin tipi

sqrt(x)	**x**-dən kvadrat kök alır	İxtiyari həqiqi tip	Arqumentin tipi
trunc(x)	**x**-i modulca aşmayan ən yaxın tam ədədi verir	İxtiyari həqiqi tip	**Integer**
round(x)	**x**-i ən yaxın tam ədədə qədər yuvarlaqlaşdırır.	İxtiyari həqiqi tip	**Integer**

İki tip öz arasında aşağıdakı şərtlər daxilində uyğun hesab edilir:

1) hər ikisi eyni tipə aiddirsə,
2) hər ikisi həqiqi tiplidirsə,
3) hər ikisi tam tiplidirsə,
4) bir tip ikinci tipin diapazon tipidirsə,
5) hər ikisi eyni bir baza tipinin diapazon tipidirsə,
6) hər ikisi eyni bir baza tipinin elementlərindən qurulmuş çoxluqlardırsa,
7) hər ikisi eyni maksimal uzunluqlu (**packed** sözü ilə təyin edilmiş) sətirlərdirsə,
8) tipin biri sətir tip, digəri isə sətir tip, sıxılmış sətir və ya simvoldursa,
9) bir tip ixtiyari göstərici, digəri isə qeyri-tip göstəricidirsə,
10) bir tip obyektə göstərci, digəri isə həmin obyektə qohum olan obyektə göstəricidirsə,
11) hər ikisi eyni tip nəticə alan prosedur tiplərdirsə.

Mənimsətmə operatorunda **t1:=t2;** (**t1** – dəyişənin tipi, **t2** – ifadənin tipi) mənimsətmə aşağıdakı hallarda mümkündür:

1) **t1** və **t2** eyni tiplidirsə və bu tip fayl və fayllar massivinə aid deyilsə və ya fayl-sahələri olan yazılışlara və ya bu yazılışların massivlərinə aid deyilsə,

2) **t1** və **t2** uyğunlaşdırılmış nizam tiplərinə aiddirsə və **t2**-nin qiyməti, **t1**-in mümkün qiymətləri diapazonuna daxildirsə,

3) **t1** və **t2** həqiqi tiplidirsə və **t2**-nin qiyməti **t1**-in mümkün qiymətləri diapazonuna daxildirsə,

4) **t1** – həqiqi tip, **t2** isə tam tiplidirsə,

5) **t1** – sətir, **t2** isə simvoldursa,

6) **t1**-sətir, **t2** isə sıxılmış sətirdirsə,

7) **t1** və **t2** uyğunlaşdırılmış sıxılmış sətirlərdirsə,

8) **t1** və **t2** uyğunlaşdırılmış çoxluqlardırsa və **t2**-nin bütün hədləri **t1**-in mümkün qiymətləri çoxluğuna daxildirsə,

9) **t1** və **t2** uyğunlaşdırılmış göstəricidirlərsə,

10) **t1** və **t2** uyğunlaşdırılmış prosedur tipləridirsə,

11) **t1** – obyekt, **t2** isə onun varisidirsə.

Proqramda bir tip verilənlər digər tip verilənlərə çevrilə bilər. Bu cür çevirmələr aşkar və ya qeyri-aşkar ola bilər. Tiplərin aşkar çevirmələri zamanı arqumentləri bir tipə, qiymətləri isə başqa tipə aid olan xüsusi funksiyalardan (**ord, trunc, round, chr, ptr**) istifadə olunur. Turbo Pascal-da tiplərin çevrilməsinin daha ümumi mexanizmindən də istifadə edilə bilər. Burada çevirmə standart tip identifikatorunun və ya istifadəçi tərəfindən çevrilən tip ifadəyə çevirmə funksiyasının identifikatoru kimi təyin edilən identifikatorun tətbiqi ilə həyata keçirilir. Turbo Pascal-da tiplərin çevrilməsi üçün daha bir aşkar üsul təyin olunub: müəyyən tip dəyişənin yaddaşda tutduğu sahəyə, onun yaddaşda daxili ifadə uzunluğuna bərabər uzunluqlu digər tip dəyişəni yerləşdirmək olar.

Tiplərin qeyri-aşkar çevirməsi yalnız aşağıdakı iki halda mümkündür:

1) həqiqi və tam tipli dəyişənlərdən təşkil olunmuş ifadələrdə tam tipli dəyişənlər avtomatik olaraq həqiqi tipə çevrilir və ifadə həqiqi qiymət alır.

2) yaddaşın eyni bir sahəsində növbə ilə gah bir, gah da digər tip verilənlərin yerləşdiyi elan edilir.

5.3. Əməllər. İfadələr

Turbo Pascal dilində aşağıdakı əməliyyatlar təyin edilib:

unar əməliyyatlar: **not**, @;

multiplikativ əməliyyatlar: *, /, **div**, **mod**, **and**, **shl**, **shr**;

additiv əməliyyatlar: +, –, **or**, **xor**;

münasibət əməliyyatları: =, <>, <, >, <=, >=, **in**.

Bu əməliyyatların yerinə yetirilmə üstünlüyü verdiyimiz ardıcıllığa uyğundur, yəni ifadələrdə birinci növbədə unar, axırıncı növbədə isə münasibət əməliyyatları yerinə yetiriləcəkdir. Əməliyyat iştirakçılarını operandlar adlandıracağıq. Müxtəlif tip ope-

randlarla əməliyyat qaydaları aşağıdakı cədvəldə verilir:

İşarə	*Əməliyyat*	*İfadə*	*Operandların tipləri*	*Nəticənin tipi*
not	İnkar	**not A**	Məntiqi	Məntiqi
not	İnkar	**not A**	İxtiyari tam	Operandın tipi
@	Ünvan	**-**	İxtiyari tam	Göstərici
*	Vurma	**A*B**	İxtiyari tam	Ən kiçik tam tip
*	Vurma	**A*B**	İxtiyari tam	**Extended**
*	Çoxluqların kəsişməsi	**A*B**	Çoxluq	Çoxluq
/	Bölmə	**A/B**	İxtiyari həqiqi	**Extended**
div	Tam bölmə	**A div B**	İxtiyari tam	Ən kiçik tam tip
mod	Bölmə qalığı	**A mod B**	İxtiyari tam	Ən kiçik tam tip
and	Məntiqi vurma	**A and B**	Məntiqi	Məntiqi
and	Məntiqi vurma	**A and B**	İxtiyari tam	Ən kiçik tam tip
shl	Sola sürüşmə	**A shl B**	İxtiyari tam	Ən kiçik tam tip
shr	Sağa sürüşmə	**A shr B**	İxtiyari tam	Ən kiçik tam tip
+	Toplama	**A+B**	İxtiyari tam	Ən kiçik tam tip
+	Toplama	**A+B**	İxtiyari həqiqi	**Extended**
+	Çoxluqların birləşdirilməsi	**A+B**	Çoxluq	Çoxluq
+	Sətirlərin birləşdirilməsi	**A+B**	Sətir	Sətir
-	Çıxma	**A-B**	İxtiyari tam	Ən kiçik tam tip

-	Çıxma	A-B	İxtiyari həqiqi	Extended
or	Məntiqi toplama	A or B	Məntiqi	Məntiqi
or	Məntiqi toplama	A or B	İxtiyari tam	Ən kiçik tam tip
=	Bərabər	A=B	İxtiyari sadə və ya sətir	Məntiqi
<>	Fərqli	A<>B	İxtiyari sadə və ya sətir	Məntiqi
<	Kiçik	A<B	Məntiqi	Məntiqi
<=	Kiçik və ya bərabər	A<=B	Məntiqi	Məntiqi
>	Böyük	A<B	Məntiqi	Məntiqi
>=	Böyük və ya bərabər	A>=B	Məntiqi	Məntiqi

Unar @ əməliyyatı ixtiyari tip operanda tətbiq edilə bilir və operandın ünvanı olan **pointer** tipli nəticə verir. Əgər bu əməliyyat prosedur, funksiya və ya obyektə tətbiq olunursa, onun nəticəsi bu proseduraya (funksiyaya, obyektə) giriş nöqtəsinin ünvanı olacaqdır.

Turbo Pascal dilində aşağıdakı məntiqi əməliyyatlar təyin edilib:

not – məntiqi inkar;

and – məntiqi vurma (konyunksiya);

or – məntiqi toplama (dizyunksiya);

xor – məntiqi toplamanın inkarı.

Məntiqi əməliyyatlar tam və məntiqi tipli operandlara tətbiq edilə bilir. Əgər operandlar tam tiplidirsə, məntiqi əməliyyatın nəticəsi də tam ədəd olacaqdır. Məntiqi tipli verilənlər üzərindəki məntiqi əməliyyatın nəticəsi məntiqi tip olacaqdır. **Integer** tipli verilənlər üzərindəki məntiqi əməliyyatların nəticələri aşağıdakı cədvəllə verilir:

Operand1	Operand2	not	and	or	xor
1	–	0	–	–	–
0	–	1	–	–	–

0	0	–	0	0	0
0	1	–	0	1	1
1	0	–	0	1	1
1	1	–	1	1	0

Boolean tipli verilənlər üzərindəki məntiqi əməliyyatların nəticələri aşağıdakı cədvəllə verilir:

Operand1	Operand2	not	and	or	xor
True	–	False	–	–	–
False	–	True	–	–	–
False	0	–	False	False	False
False	True	–	False	True	True
True	0	–	False	True	True
True	True	–	True	True	False

Turbo Pascal dilində məntiqi tipə, tam ədədlər üzərində aparılan aşağıdakı iki sürüşdürmə əməliyyatı da aid edilir: **i shl j** – əməliyyatı nəticəsində **i** – nin tərkibi j sayda mərtəbə sola sürüşdürülür, bu zaman boşalan kiçik mərtəbələr sıfırlarla doldurulur; **i shr j** – əməliyyatı nəticəsində isə **i**-nin tərkibi j sayda mərtəbə sağa sürüşdürülür, bu zaman boşalan yüksək mərtəbələr sıfırlarla doldurulur. Bu əməliyyatlarda **i** və **j** ixtiyari tam tipli ifadələrdir.

In münasibət əməliyyatı iki operanda tətbiq edilir. Sol operand ixtiyari nizam tipli, ikinci operand isə həmin tip elementlərdən ibarət çoxluq və ya çoxluq tipli identifikator olmalıdır.

Proqramın yerinə yetirilən hissəsinin qurulduğu əsas elementlər, sabitlər, dəyişənlər və funksiyalara münasibətlərdir. Bu elementlərin hər biri öz qiyməti ilə xarakterizə edilir və verilənlərin hansı bir tipinə aid olur. Əməliyyat işarələri və mötərizələrin köməyi ilə onlardan ifadələr təşkil etmək mümkün olur. İfadələr isə faktiki olaraq, yeni qiymətlərin alınması qaydalarıdır. İfadənin xüsusi halı sadəcə bir element, yəni sabit, dəyişən və ya funksiyaya müraciət ola bilər. Bu cür ifadənin qiyməti, aşkardır ki, elementlə eyni bir tipə aid olacaqdır. Daha ümumi halda ifadə bir neçə elementdən (operanddan) və əməliyyat işarələrindən ibarətdir. İfadənin qiymətinin tipi, operandların tipi və onlara tətbiq olunmuş əməliyyatların tipi ilə müəyyən olunur. Hesabi ifadələr

ədədi kəmiyyətlər üzərindəki əməllərin yerinə yetirilmə ardıcıllığını müəyyən edir. Bu ifadələr hesabi əməliyyatlardan, funksiyalara müraciətlərdən, operandlardan (sabitlər, dəyişənlər) və yumru mötərizələrdən ibarət olur. *Məsələn*,

$$2*a+sqrt(2*\sin(x+y)))/(0.2*c-\ln(x-y)).$$

Bu ifadələrdə, yerinə yetirilmə üstünlüyu daha yüksək olan əməliyyatlar əvvəl yerinə yetirilir. Burada yerinə yetirilmə üstünlüyü azalma sırası ilə aşağıdakı kimidir:

1) Funksiyaların hesablanması;
2) (-) işarənin dəyişdirilməsinin unar əməliyyatı;
3) *****, **/**, **div**, **mod**;
4) **+**, **-**.

Bir-birinin ardınca gələn eyni hüquqlu əməliyyatlar, ifadədə soldan sağa doğru yerinə yetirilir. İfadədə mötərizə daxilindəki əməliyyatlar, yernə yetirilmə üstünlüyündən asılı olmayaraq birinci növbədə hesablanırlar. Riyazi mənası olmayan ifadələr, məsələn, sıfıra bölmə, mənfi ədədin loqarifmi və s. kimi ifadələr yazmaq olmaz. Məsələn, aşağıdakı ifadə üzərindəki rəqəmlər əməliyyatların yerinə yetirilmə ardıcıllığını bildirir:

$$\overset{1}{(}1+y)*\overset{7}{(}2*x+\overset{4}{sqrt}\overset{5}{(}y\overset{3}{)}-\overset{6}{(}x+y\overset{2}{)})/\overset{12}{(}y+1/\overset{11}{(}\overset{10}{sqr}\overset{8}{(}x\overset{9}{)}-4))$$

Turbo Pascal dilində ədədin ixtiyari tərtibindən qüvvətə yüksəltmə əməli və ya standart funksiyası nəzərdə tutulmayıb. x^y -in hesablanması üçün əgər y tam ədəddirsə, qüvvət vurma əməli ilə təyin edilir, məsələn, $x^3 \rightarrow sqr(x)*x$ daha böyük qüvvətlər isə dövr daxilində vurma ilə tapılır. Burada **y** – həqiqi ədəd olduqda isə $x^y = \exp(y \cdot \ln(x))$ riyazi düsturundan istifadə edilir, Pascal dilində bu $\exp(y*\ln(x))$ şəklində ifadə edilir.

Məntiqi ifadələr məntiqi əməliyyatlar və yumru mötərizələrlə əlaqələndirilən məntiqi operandlardan ibarətdir. Məntiqi ifadənin yerinə yetirilmə nəticəsi **false** və ya **true** məntiqi sabitidir. Məntiqi operandlar, məntiqi sabitlər, dəyişənlər, funksiyaya müraciətlər, münasibət əməliyyatları ola bilər. *Məsələn*,

1) **x<2*y**; 2) **true**; 3) **d**; 4) **odd(k)**; 5) **not not d** ; 6) **not(x>y/2)**;

7) **d and (x<>y) and b**; 8) **(c or d) and (x=y)**

or not b.

Burada **d=true; b=false; c=true; x=3.0; y=0.5; k=5** olarsa, nəticədə alarıq:

1) **false;** 2) **true**; 3) **true**; 4) **true**; 5) **true**;

6) **false**; 7) **false**; 8) **true**.

5.4. Mənimsətmə operatoru, qurma operator və boş operator

Turbo Pascal-ın əsas operatorlarından biri mənimsətmə operatorudur. Operatorun sol tərəfində dəyişən adı verilir, sağ tərəf isə dəyişən adı ilə eyni tipli olan ifadədən ibarətdir. Operatorun sol və sağ tərəfləri mənimsətmə işarəsi adlanan «**:=**» simvollar cütü ilə əlaqələndirilir. *Məsələn*:

x:=x+1; A:=5; z:=-637.225; D:=(x>y) and (k<>); k:='PASCAL'.

Qeyd edək ki, mənimsətmə operatorunda həmişə «**:=**» simvollar cütündən istifadə edilir, sabitlərin təsvirində isə «**=**» simvolu tətbiq edilir. Mənimsətmə operatorunda dəyişən adı ilə ifadə arasındakı tip uyğunluğunun mümkün variantları yuxarıda verilmişdir.

Qurma operator – **begin - end** operator mötərizəsi daxilinə alınmış proqramın ixtiyari operatorlar ardıcıllığıdır. Qurma operatorlar proqramları struktur proqramlaşdırmanın müasir texnologiyaları ilə hazırlımağa imkan verən vacib alətlərdən biridir. Turbo Pascal dili qurma operatora daxil olan operatorlara heç bir məhdudiyyətlər qoymur. Bu operatorların daxilində digər qurma operatorlar da ola bilər. Burada **begin - end** sözləri ilə məhdudlaşdırılmış operatorlar ardıcıllığı bir qurma operator deməkdir. Burada **end** sözü bağlayıcı operator mötərizəsi olduğundan, o həm də ondan əvvəlki operatorun sonunu bildirir, buna görə də ondan əvvəl «**;**» işarəsini qoymaq məcburi deyildir. **end** sözündən əvvəl «**;**» işarəsinin qoyulması, axırıncı operator və end sözləri arasında boş operatorun verildiyini bildirir. Boş operator heç bir əməliyyat yerinə yetirmir və onun daxil edilməsi üçün proqrama əlavə «**;**» işarəsini əlavə etmək kifayətdir. Boş operator əsasən idarəetməni qurma operatorun sonuna verilməsi üçün istifadə olunur.

5.5. Daxiletmə və xaricetmə operatorları

Daxiletmə operatoru (daxiletmə standart prorseduruna müraciət) aşağıdakı formadadır:

read(<daxiletmə siyahısı>) ;

burada <daxiletmə siyahısı> vergüllə bir-birindən ayrılan dəyişən adları ardıcıllığıdır. Məsələn, **read(a,b,c,d)**;. Bu operatorun yerinə yetirilməsi zamanı proqramın işi dayandırılır və istifadəçi klaviaturada **a,b,c,d** dəyişənlərinin qiymətlərini, bir-birindən probellə ayırmaq şərtilə daxil etməlidir. Bu zaman daxil edilən qiymətlər ekranda görünür. Sonda *Enter* düyməsi sıxılır. *Məsələn,*

```
var t:real; i:integer; k:char;
begin read(t,i,k) end;
```

Klaviaturada məsələn, yığmaq olar: **123.41 10 G (Enter)**.

Əgər proqramda bir neçə **read** operatoru varsa, onda onlar üçün verilənlər bir-birinin ardınca daxil edilir. *Məsələn,*

```
var a,b:integer; c,d:real;
begin read(a,b);
      read(c,d);
end;
```

Onda verilənlər klaviaturadan aşağıdakı kimi daxil edilər:

187 34 (Enter) 2.17E-02 1.5E+01 (Enter)

Bu operatorun digər variantı aşağıdakı formaya malikdir:

readln (<daxiletmə siyahısı>) ;

Bu operatorun **read** operatorundan fərqi yalnız ondadır ki, bir **readln** operatorunun siyahısındakı axırıncı verilən daxil edildikdən sonra ondan sonra gələn digər daxiletmə operatorunun siyasındakı verilənlər yeni sətirdən başlayaraq daxil ediləcəkdir. Məsələn, yuxarıdakı misalda

```
readln(a,b);
readln(c,d);
```

əvəzləməsi aparsaq, klaviaturadan daxiletmə aşağıdakı şəkildə olacaqdır:

187 34 (Enter)
2.17E-02 1.5E+01 (Enter).

Qeyd edək ki, bu operatorun <daxiletmə siyahısı> boş da ola bilər. Bu operatorla klaviaturadan daxil edilən verilənlər arasında sadəcə boş sətir daxil etmək olur.

Xaricetmə operatoru (standart xaricetmə proseduruna müraciət) aşağıdakı formaya malikdir:

write (<xaricetmə siyahısı>)

Burada < xaricetmə siyahısı>nın elementləri müxtəlif tip ifadələr (xüsusi halda sabit və dəyişənlər) ola bilər. *Məsələn,*

```
write(132); write(a+b+2); write(x,y,z);
```

və s. Bu operatorla ekrana çıxarılan bir neçə ədəd bir-birindən probellə ayrılmırlar, buna görə də siyahıda bir neçə qiymət olduqda, bunu nəzərə al-maq lazımdır. *Məsələn,*

```
write(a,' ',b,' ',c);
```

Nəticədə ekranda alarıq:

```
1 2 3
```

Burada sonuncu qiymət çıxarıldıqdan sonra kursor elə bu sətirdə qalır.

Operatorun digər variantı aşağıdakı formadadır:

writeln (< xaricetmə siyahısı >);

Bu operatorun **write** operatorundan fərqi yalnız ondadır ki, bu operatorun siyahısındakı sonunucu qiymət ekrana çıxarıldıqdan sonra kursor növbəti sətrə keçirilir və yeni qiymətlər bu sətirdən başlayaraq çıxarılır. Bu operatorun siyahısı boş olduqda, ekrana boş sətir çıxarılır.

Operatorun siyahısında çıxış formatını təyin edən göstəricilər verilə bilər. Format xaric edilən qiymətin ekrandakı ifadəsini təyin edir. Format aid olduğu elementdən «:» işarəsi ilə ayrılır. Əgər format verilməyibsə, onda kompüter qiyməti, susmaqla nəzərdə tutulmuş qaydada çıxarır.

Müxtəlif tip verilənlərin formatlı və formatsız xaricetmə qaydalarına baxaq. Siyahıdakı verilənləri göstərmək üçün aşağıdakı işarələmələrdən istifadə edək: **i,p,q** – tam tipli ifadələr, **r** – həqiqi tipli ifadə, **m** – məntiqi tipli ifadə, **c** – simvol tipli kəmiyyət, **s** – sətir tipli ifadə, **#** - ədəd, *** -** "+"və ya «-» işarələri, **_** isə probel işarəsidir.

1) Formatsız çıxarışla **i** kəmiyyətinin kursorun durduğu mövqedən başlayaraq onluq ifadəsi çıxarılır:

```
i:=134; write(i); nəticə: 134;
```

`i:=287; write (i,i,i);` nəticə: **287287287**;

2) Formatlı çıxarışla **i** kəmiyyətinin **i:p** formatı ilə onluq ifadəsi **p** eni olan sahənin kənar sağ mövqeyinə çıxarılır:

`i:=134;write(i:6);` nəticə: `_ _ _134;`

`i:=123; write ((i+i):7);` nəticə: `_ _ _ _ 246;`

3) Formatsız çıxarışla **r** kəmiyyətinin eksponensial formada, 18 simvol eni olan sahəyə onluq ifadəsi çıxarılır. Burada əgər $r \geq 0{,}0$ olarsa, `_ #.##########E*##` formatından, əks halda `_ -#.##########E*##` formatından istifadə edilir:

`r:=666.787; write(r);`

nəticə: `_6.6678700000E+02;`

`r:=-1.999E+01; write(r);`

nəticə: `_-1.999000000E+01;`

4) Formatlı çıxarışla **r** kəmiyyətinin **r:p** formatı ilə eksponensial formada onluq ifadəsi **p** eni olan sahənin kənar sağ mövqeyinə çıxarılır. Burada müsbət ədədlər üçün çıxarış sahəsinin minimal uzunluğu 7 simvol, mənfi ədədlər üçün isə 8 simvoldur:

`r:=555.04; write(r:15);` nəticə:
`5.550400000E+02;`

`r:=46.78; write(-r:12);` nəticə: `-4.67800E+01;`

5) Formatlı çıxarışla **r** kəmiyyətinin **r:p:q** formatı ilə qeyd olunmuş onluq nöqtə ilə onluq ifadəsi **p** eni olan sahənin kənar sağ mövqeyinə çıxarılır. Burada onluq nöqtədən sonra **q** ($0 \leq q \leq 24$) sayda rəqəm, ədədin kəsr hissəsini ifadə edir. Əgər **q=0** olarsa, nə onluq nöqtə, nə də kəsr hissə çıxarılmır və əgər q > 24 olarsa isə ədəd eksponensial formada çıxarılır:

`r:=511.04; write(r:8:4);` nəticə: **511.0400**;

`r:=-46.78; write (r:7:2);` nəticə: `_-46.78;`

6) Formatlı çıxarışla **c** kəmiyyəti **c:p** formatı ilə **p** eni olan sahənin kənar sağ mövqeyinə çıxarılır: `c:='x'; write(c:3);` nəticə: `__x;`

7) Formatsız çıxırılışla **s** kəmiyyəti kursorun durduğu mövqedən başlayaraq çıxarılır: `s:='PASCAL'; write(s);` nəticə: **PASCAL**;

8) Formatlı çıxarışla **s** kəmiyyəti **s:p** formatı ilə **p** eni olan sahənin kənar sağ mövqeyinə çıxarılır:

```
s:='PASCAL'; write(s:10);
```

nəticə: `____PASCAL;`

9) Formatsız çıxarışla **m** kəmiyyəti kursorun durduğu mövqedən başlayaraq çıxarılır: `m:= true; write (m);` nəticə: `true;`

10) Formatlı çıxarışla **m** kəmiyyəti `m:p` formatı ilə p eni olan sahənin kənar sağ mövqeyinə çıxarılır:

```
m:=false; write(m:6, not m:7);
```

nəticə: `_false___true`.

5.6. Nişanlar və keçid operatorları. Şərt operatoru. Variant operatoru

Turbo Pascal-da nişan – proqramın ixtiyari opretorunu adlandırmağa imkan verən və beləliklə, ona müraciəti təmin edən ixtiyari identifikatordur. Standart Pascal dili ilə uyğunluq yaratmaq üçün Turbo Pascal dilində də nişan kimi işarəsiz tam ədədlərdən isitfadə edilə bilər. Nişan işarələdiyi operatorun bilavasitə qarşısında yerləşdirdirilir və ondan «:» işarəsi ilə ayrılır. Operatoru bir neçə nişanla işarə etmək olar, bu halda bu nişanlar birbirindən «:» işarəsi ilə ayrılır. Proqramda nişandan istifadə etməməzdən əvvəl onu təsvir etmək lazımdır. Nişanlar təsvirlər bölməsinin **label** (nişan) bölməsində elan edilir. Məsələn, `label 1,2,11,12;` və s.

Proqramda idarəetməni şərtsiz olaraq proqramın bu və ya digər hissəsinə vermək üçün şərtsiz keçid operatorundan istifadə olunur. Bu operatorun ümumi şəkli aşağıdakı kimidir:

goto < nişan >;

Burada **goto** operatorun işçi sözü, <nişan> isə proqramda istifadə olunmuş hər hansı nişandır. Bu operator yerinə yetirilərkən idarəetmə nişanı verilmiş operatora verilir. Bu operatordan istifadə edərkən aşağıdakıları nəzərə almaq lazımdır:

1) **goto** operatorunda istifadə edilən nişan təsvirlər bölməsində elan edilməlidir və bu nişan proqramda istifadə edilməlidir;

2) Prosedurda (funksiyada) təsvir olunmuş nişanlar burada lokallaşdırılır, buna görə də prosedurdan (funksiyadan) kənardan bu nişanlara müraciət etmək olmaz.

Ümumiyyətlə, müasir proqramlaşdırma texnologiyası

«**goto** operatoru olmadan proqramlaşdırma» prinsipinə əsaslanıb. Hesab olunur ki, bu operator proqramı mürəkkəbləşdirir, lakin buna baxmayaraq bəzi hallarda həmin keçid operatorundan istifadə əlverişli olur.

Şərt operatoru müəyyən şərti yoxlayaraq, yoxlamanın nəticəsindən asılı olaraq bu və ya digər əməliyyatı yerinə yetirməyə imkan verir. Beləliklə, şərt operatoru hesablama prosesinin budaqlanmasını təmin edən vasitədir. Şərt operatorunun strukturu aşağıdakı şəkildədir:

if < şərt > **then** < operator 1> **else** <operator 2 >;

Burada **if**, **then**, **else** operatorun işçi sözləridir, <şərt> - məntiqi tipli ixtiyari ifadədir, <operator 1> və <operator 2> isə Turbo Pascal dilinin ixtiyari operatorlarıdır.

Bu operator yerinə yetirilərkən, əvvəlcə <şərt> məntiqi ifadəsi hesablanır. Əgər ifadənin nəticəsi **true** (doğru) qiymətini alarsa, <operator1> yerinə yetirilir, <operator2> isə nəzərə alınmır, qiymət **false** (yalan) olarsa, əksinə <operator1> nəzərə alınmır və <operator2> yerinə yetirilir. *Məsələn,*

```
if x<0 then y:=x+1 else y:=2*x;
if (n>15) and (n<25) then a:=n+4 else
b:= n-5;
```

Misal, x və y ədədlərindən ən böyüyünü tapmalı.

```
program p1;
var x,y,max:integer;
begin read(x,y);
if x>y then max:=x else max:=y;
write(max)
end.
```

Şərt operatorunun **else** < operator 2 > hissəsi verilməyə də bilər, yəni operator aşağıdakı şəklə düşər:

if < şərt > **then** < operator 1>;

Bu zaman <şərt> **true** qiymətini aldıqda <operator 1> yerinə yetirilir, əks halda isə operator yerinə yetirilmədən proqramın növbəti sətrinə keçid yerinə yetirilir.

Misal. **x**, **y**, **z** ədədlərindən ən kiçiyini tapmalı.

```
program p2;
var x,y,z,min:integer;
```

```
begin read(x,y,z);min:=x;
if min>y then min:=y;
if min>z then min:=z;
write(min)
end.
```

Buradakı <operator 1> və <operator 2> - dən ixtiyari biri, ixtiyari tipli, o cümlədən digər şərt operatoru da ola bilər və operatorda **else** <operator 2> hissəsi verilməyə də bilər. Bu zaman yarana bilən qeyri-müəyyənlik Turbo Pascal dilində aşağıdakı qaydada həll olunur: proqram mətnində rast gəlinən ixtiyari **else** hissəsi ona proqramda ən yaxın olan **THEN** hissəsinə uyğun gətirilir.

Misal. Kvadrat tənliyin həllini tapmalı.

```
program kv;
var a,b,c,d,x1,x2:real;
begin read(a,b,c);d:=sqr(b)-4*a*c;
if d>=0 then begin x1:=(-b+sqrt(d))/(2*a);
                   x2:=(-b-sqrt(d))/(2*a);
                write('x1=',x1,'x2=',x2)
             end
        else write('helli yoxdur');
end.
```

Variant operatoru proqramın bir neçə mümkün variantlarından hər hansı birini seçməyə imkan verir. Seçkinin aparıldığı parametr <seçki açarı> **real** və **string** tipləri istisna olmaqla ixtiyari tip ifadədir. Variant operatorunun ümumi strukturu aşağıdakı kimidir:

case <seçki açarı> **of** < seçki siyahısı> **else** <operatorlar> **end;**

burada **case**, **of**, **else**, **end** işçi sözlərdir, < seçki açarı > - seçki açarı, <seçki siyahısı> - <seçki sabiti> : <operator> formalı bir və ya daha çox konstruksiyalar; <seçki sabiti> isə <seçki açarı> ifadəsi ilə eyni tipli sabitdir, <operatorlar> - Turbo Pascal dilinin ixtiyari operatorlarıdır.

Variant operatoru aşağıdakı qaydada işləyir: əvvəlcə <seçki açarı> ifadəsi hesablanır, sonra <seçki siyahısı> operatorları ardıcıllığında qarşısında hesablanmış ifadənin aldığı qiymətə bərabər

olan sabit gələn operator seçilir. Tapılmış operator yerinə yetirilir və seçki operatorunun işi sona çatır. Əgər seçki siyahısında seçki açarının hesablanmış qiymətinə uyğun gələn sabit tapılmazsa, onda idarəetmə **else** sözündən sonra gələn operatorlara verilir. Qeyd edək ki, operatorda **else** <operatorlar > hissəsini verməmək də olar. Bu halda seçki siyahısında lazım olan sabit tapılmadıqda heç bir hadisə baş verməyəcəkdir və seçki operatoru sadəcə olaraq öz işini sona çatdıracaqdır.

Misal. Toplama, çıxma, vurma və ya bölmə əməliyyatlarından hər hansı birinin seçilməsi ilə ixtiyari **x, y** ədədləri üçün bu əməlin yerinə yetirilməsi.

```
program p3;
var c:char; x,y,z:real; s:boolean;
begin s:=false;
repeat read(x,' ', y); read(c);
case c of
'+': z:=x+y;
'-': z:=x-y;
'*': z:=x*y;
'/': z:=x/y;
else s:=true;
end;
if not s then writeln('z =',z)
until s
end.
```

Burada seçki siyahısındakı ixtiyari operatorun qarşısında bir deyil, bir-birindən vergüllə ayrılan bir neçə seçki sabiti dura bilər. *Məsələn,*

```
var c:char;
begin read(c);
case c of
'n','N': wrieln ('No');
'y','Y': writeln ('Yes')
end
end.
```

5.7. Dövr operatorları

Turbo Pascal dilində üç müxtəlif dövr operatoru mövcuddur:

1) Hesabi dövr operatoru **FOR** aşağıdakı struktura malikdir:

for <dövr parametri>**:=** <başlanğıc qiymət> **to** <son qiymət> **do** <operator>**;**

Burada **for, to, do** – operatorun işçi sözləridir, <dövr parametri> – **integer** (daha doğrusu ixtiyari nizam tipli) tipli dəyişəndir, <başlanğıc qiymət> - dövr parametrinin başlanğıc qiymətidir və onunla eyni tiplidir, <son qiymət> – dövr parametrinin aldığı son qiymətdir və onunla eyni tiplidir, < operator > – Turbo Pascal dilinin ixtiyari operatorudur.

Bu operator yerinə yetirilərkən əvvəlcə <başlanğıc qiymət> ifadəsi hesablanır və <dövr parametri>**:=**<başlanğıc qiymət> mənimsədilməsi yerinə yetirilir. Sonra <dövr parametri> **<=** <son qiymət> şərti yoxlanılır, əgər şərt ödənilmirsə **for** operatorunun işi sona çatır, şərt ödənildikdə isə <operator> yerinə yetirilir və<dövr parametri> dəyişəni bir vahid artırılır: <dövr parametri>**:=** <dövr parametri>**+1** Bundan sonra təsvir etdiyimiz proses şərt ödənilməyənədək təkrarlanır.

Misal. Birinci N natural ədədin cəmini tapmalı.

```
program p4;
var i,n,s:integer;
begin readln(n);s:=0;
for i:=1 to n do s:=s+i;
    writeln(s)
end.
```

Qeyd edək ki, **for** operatorunun işini idarə edən şərt <operator> - un yerinə yetirilməsindən əvvəl yoxlanılır, əgər şərt **for** operatorunun işinin əvvəlində yerinə yetirilmirsə, onda operator bir dəfə də olsun yerinə yetirilməyəcəkdir. **for** operatorunda dövr parametrinin artım addımı sabitdir və (+1)-ə bərabərdir. Lakin **for** operatorunun aşağıdakı forması da mövcuddur:

for <dövr parametri>**:=** <başlanğıc qiymət> **downto** <son qiymət> **do** <operator>

Burada **to** sözünün **downto** sözü ilə əvəz edilməsi dövr parametrinin artım addımının (-1)-ə bərabər olduğunu və idarəedici şərtin <dövr parametri> >= <son qiymət> olduğunu göstərir. Məsələn, yuxarıda baxdığımız misal üçün proqramı elə dəyişdirmək olar ki, proqram həm müsbət, həm də mənfi ədədlərin cəmini tapa bilsin:

```
program p5;
var i,n,s:integer;
begin readln(n); s:=0;
if n>=0 then for i:=1 to n do s:=s+i
        else for i:=-1 downto n do s:=s+i;
writeln(s)
end.
```

2) Şərt qabaqcadan yoxlanılan **while** dövr operatorunun ümumi şəkli aşağıdakı kimidir:

while <şərt> **do** <operator>

Burada **while**, **do** – operatorun işçi sözləridir, <şərt> - məntiqi tipli ifadə, <operator> isə Turbo Pascal dilinin ixtiyari operatorudur.

Bu operator yerinə yetirilərkən əvvəlcə <şərt> ifadəsi hesablanır, əgər o, **true** qiymətini alırsa, yəni şərt ödənilərsə <operator> yerinə yetirilir və <şərt> ifadəsinin hesablanıb, yoxlanılması davam etdirilir. Bu <şərt> **false** qiyməti alana qədər təkrarlanır, <şərt> bu qiyməti alan kimi **while** operatoru işini sona çatdırır. Əgər elə birinci yoxlamada < şərt > **false** qiymətini alırsa <operator> bir dəfə də yerinə yetirilmədən **while** operatoru işini başa çatdırır.

Misal. $f(x)=\cos 2x$ funksiyasının [0,1] parçasında $h=0{,}05$ addımı ilə qiymətlər cədvəlini almalı.

```
program p6;
const x0=0; h=0.05; xs=1;
var x,y:real;
begin x:=x0;
  while x<xs+h do
  begin y:=cos(2*x); writeln(x:5:2,y:6:4);
        x:=x+h
```

```
   end
end.
```

3) Şərt sonradan yoxlanılan **repeat-until** dövr operatorunun ümumi şəkli aşağıdakı kimidir:

repeat <dövrün gövdəsi> **until** <şərt>; burada **repeat**, **until**-dövr operatorunun işçi sözləridir, <dövr gövdəsi> - Turbo Pascal dilinin operatorlarının ixtiyari ardıcıllığıdır, <şərt> - məntiqi ifadədir.

Operator yerinə yetirilərkən, <dövr gövdəsi> operatorları ən azı bir dəfə yerinə yetirilir və bundan sonra <şərt> məntiqi ifadəsi hesablanır. Əgər bu ifadə **false** qiymətini alırsa, yəni şərt ödənilmirsə, <dövrün gövdəsi> operatorları təkrarən yerinə yetirilir və bu prosses <şərt> **true** qiymətini alana qədər davam etdirilir. Bu halda dövr operatorunun işi sona çatır.

Misal. Müsbət a tam ədədi verilib ($a>1$). $n!>a$ şərtini ödəyən ən kiçik n ədədini tapmalı.

```
program p7;
var p,n,a:integer;
begin read(a); p:=1; n:=1;
   repeat n:=n+1; p:=p*n
   until p>a;
   writeln(n)
end.
```

Burada gördüyümüz kimi **repeat** – **until** cütü **begin** – **end** operator mötərizələrinə oxşardır, ona görə də **until**-dən əvvəl «;» qoymaq məcburi deyildir.

Qeyd edək ki, dövrlər bir-birinin daxilində də verilə bilər. Bu zaman daxildəki dövr xaricdəki dövrün hər bir qiyməti üçün tam yerinə yetirilir.

Misal. $y=2k+n$-nin $n=1,2,3$ və $k=2,4,6$ hallarında qiymətlərini tapmalı.

```
program p8;
var n,k,y:integer;
begin for n:=1 to 3 do
          begin k:=2;
            while k<8 do
```

```
                begin y:=2*k+n;
                      write(n,k,y);
                      k:=k+2
                end
          end
end.
```

Qeyd edək ki, **for**, **while** və **repeat** dövr operatorlarının idarə edilməsini sadələşdirmək üçün Turbo Pascal dilinə aşağıdakı iki prosedur daxil edilib:

1) **break** – dövrdən dərhal çıxışı təmin edir, bu prosedur idarəetməni dövr operatorundan dərhal sonra gələn operatora verir;

2) **continue** – dövrün növbəti təkrarlanmasının vaxtından əvvəl sona çatdırılmasını təmin edir, yəni idarəetmənin dövr operatorunun sonuna ötürülməsi prosesinə ekvivalentdir.

Misal. $1+\frac{1}{2}+\frac{1}{3}+...+\frac{1}{i}+...$ harmonik sırasının **eps** dəqiqliyi ilə qiymətini hesablayaq.

```
program p8;
var s,eps:real; i:integer;
begin read(eps); s:=0; i:=1;
    while (1/i>=eps) and (i<maxint) do
       begin s:=s+1/i; i:=i:+1 end;
    write (s)
end.

program p9;
var s, eps: read; i:integer;
begin read(eps); s:=0; i:=1;
  repeat
    s:=s+1/i; i:=i+1
  until (1/i<eps) or (i>=maxint);
  write(s)
end.
```

5.8. Massivlər

Massiv – eyni bir identifikatorlarla işarələnən eyni tipli elementlər ardıcıllığıdır. Massivlərin təsvirinin ümumi şəkli aşağıdakı kimidir:

< tipin adı > **=array[**< indeks tiplərinin siyahısı >**] of** < tip >;

burada <tipin adı> - identifikatordur, **array**, **of** – operatorun işçi sözləridir, <indeks tiplərinin siyahısı> – kvadrat mötərizələr daxilində bir-birindən vergüllə ayrılmaqla gətirilən bir və ya bir neçə indeks tiplərinin siyahısıdır, <tip> – Turbo Pascal dilinin ixtiyari tipidir. İndeks tipi kimi Turbo Pascal-da **longint** tipindən və **longint** baza tipi olan diapazon tiplərindən başqa ixtiyari nizam tiplərindən istifadə etmək olar. Dəyişəni massiv kimi, onun bilavasitə təsviri zamanı, massiv tipinin əvvəlcədən təsviri olmadan təyin etmək olar. *Məsələn*,

```
var a,b:array[1..10] of real;
```

İndeks tipi kimi adətən indekslərin dəyişmə sərhədləri verilən diapazon tipindən istifadə edilir. Burada **OF** sözündən sonra gələn <tip> Turbo Pascal-ın ixtiyari tipi olduğundan, bu tip xüsusi halda digər massiv də ola bilər. *Məsələn*,

```
type mat=array[0..5]of array[-2..2] of
array[char] of byte;
```

bu yazılışı aşağıdakı kimi də vermək olar:

```
type mat=array [0..5,-2..2,char] of byte;
```

İndeks tiplərinin siyahısındakı elementlərin sayı məhdudlaşdırılmır, lakin ixtiyari massivin daxili ifadəsinin uzunluğu 65520 baytı aşmamalıdır. Kompüter yaddaşında massiv elementləri bir-birinin ardınca gəlir və burada kiçik ünvanlardan daha yüksəklərinə keçid zamanı daha tez massivin ən sağda duran indeksi dəyişir. *Məsələn*,

```
var a:array[1..2,1..2] of byte;
begin  a[1,1]:=1;  a[2,1]:=2;  a[1,2]:=3;
a[2,2]:=4;
end.
```

Bu halda kompüter yaddaşında ardıcıl olaraq bir-birinin ardınca 1, 2, 3, 4 qiymətlərinə uyğun baytlar düzüləcəkdir.

Turbo Pascal dilində bir mənimsətmə operatoru ilə bir massivin bütün elementlərini onunla eyni tipli olan başqa massivə mənimsətmək olar.

Məsələn,

```
var a,b:array[1..5] of single;
begin
.....
  a:=b;
.....
end.
```

Nəticədə, **A** massivinin bütün beş elementi, **B** massivinin elementlərinin aldığı qiymətləri alacaqdır. Lakin massivlər üzərində münasibət əməliyyatları nəzərdə tutulmayıbdır. Massivlərin yalnız elementlərini müqayisə etmək olar. Aşağıdakı qayda ilə massivin maksimal elementini və onun massivdəki mövqeyini təyin etmək olar:

```
program p9;
var t:array[1..12] of integer;
    i,max,p:integer;
begin
  for i:=1 to 12 do read(t[i]);
  max:=t[1]; p:=1;
  for i:=2 to 12 do
    if t[i]>max then begin max:=t[i]; p:=i
end;
  write(max,p)
end.
```

Massivdəki elementləri artım sırası ilə düzmək üçün aşağıdakı qayda ilə çeşidləmə aparmaq olar:

```
.....
for i:=1 to n-1 do
for k:=1 to n-i do
if t[k]>t[k+1] then
begin a:=x[k];   x[k]:=x[k+1];   x[k+1]:=a
end;
```

5.9. Yazılışlar

Yazılış – yazılış sahələri adlanan, qeyd olunmuş sayda elementdən ibarət verilənlər strukturudur. Massivdən fərqli olaraq, yazılışın elementləri (sahələri) müxtəlif tiplidir. Yazılışın bu və ya digər elementinə müraciət etmək mümkünlüyünü təmin etmək üçün sahələr adlandırılırlar. Yazılışın elanının ümumi strukturu aşağıdakı kimidir:

<tip adı> **=record** <sahələr siyahısı> **end**;

burada <tip adı> – identifikator, **record**, **end** – operatorun işçi sözləridir, <sahələr siyahısı> – ayrıcı kimi aralarında «**;**» işarəsi qoyulan yazılış bölmələrinin ardıcıllığından ibarət sahələr siyahısıdır.

Yazılışın hər bir bölməsi, bir-birindən vergüllə ayrılan bir və ya bir neçə sahə identifikatorlarından ibarətdir. İdentifikatordan sonra «**:**» işarəsi və sahə tipinin təsviri verilir. *Məsələn,*

```
type birthday=record
                    day, month: byte;
                    year: word;
                  end;
var a,b:brithday;
```

Massivlərdə olduğu kimi yazılış tipinin dəyişənlərinin qiymətini, eyni tipli başqa dəyişənlərə mənimsətmək olur, məsələn, **a:=b;**

Yazılışın hər bir elementinə müraciət üçün elementin dəqiqləşdirilmiş adından istifadə edilir:

<dəyişən adı>.<sahə adı>;

Məsələn, **a.day:=21; b.year:=1966;**

Yazılışda sahələr bir-birinin daxilində də verilə bilər. *Məsələn,*

```
type brithday=record
                    day,month:byte;
                    year:Word
                  end;
var c:record
```

```
            name:string;
            bd:brithday
          end;
begin
......
If c.bd.year=1966 then ...
end.
```

Yazılış sahələrinə müraciəti sadələşdirmək üçün **with** birləşdirmə operatorundan istifadə edilir:

with <dəyişən> **do** <operator>;

burada **with**, **do** – operatorun işçi sözləridir, <dəyişən> – ardınca daxilində ola biləcək sahələrin siyahısı olan yazılış tipli dəyişən adıdır, <operator> – Turbo Pascal dilinin ixtiyari operatorudur. *Məsələn*,

```
whith c.bd do month:=4;
```

və ya ona ekvivalent olan aşağıdakı operatorlar

```
with c do with bd do month:=4;
```

və ya

```
with c,bd do month:=4;
```

və ya

```
c.bd.month:=4;
```

Misal. Tələbələrin proqramlaşdırma fənni üzrə imtahan cədvəli verilib. Əla qiymətlər almış tələbələrin sayını tapmalı.

```
program imt;
type t=record
          saa:string[30];
          qiy:2..5
        end;
var s:t; i,k,n:byte;
begin k:=0; readln(n);
  for i:=1 to n do
    begin
       read(s.saa); read(s.qiy);
       if s.qiy=5 then k:=k+1
    end;
   write (k)
end.
```

5.10. Çoxluqlar

Çoxluqlar – bir-birilə məntiqi əlaqələndirilən eyni tipli obyektlər külliyatıdır. Obyektlər arasındakı əlaqə istifadəçi tərəfindən müəyyən olunur və Turbo Pascal tərəfindən heç bir qayda ilə nəzarət edilmir. Çoxluqlar sonlu olur və buradakı elementlərin sayı 0-dan 256-ya qədər dəyişə bilər. Elementləri olmayan çoxluq boş çoxluq adlanır. Massiv elementlərindən fərqli olaraq çoxluq elementləri nömrələnməyib və müəyyən qayda ilə düzülməyib. Əməliyyatlar yalnız bütövlükdə çoxluq üzərində aparıla bilər. Çoxluğun konkret qiymətləri çoxluq konstruktorunun köməyi ilə daxil edilir. Konstruktor, kvadrat mötərizələrə alınan, bir-birindən vergüllə ayrılmış elementlər ardıcıllığıdır. Elementlər sabit və ya baza tipinin ifadələri ola bilər. Məsələn, **[3,4,7,9,12]**, **[1..100]**, **['a','b','c']**, **['A'..'Z']** və s. Burada **[ ]** simvolu boş çoxluğu, yəni heç bir elementi olmayan çoxluğu bildirir. İki çoxluq o vaxt ekvivalent hesab edilir ki, onların bütün elementləri eyni olsun və burada elementlərin çoxluq daxilində verilmə qaydasının əhəmiyyəti yoxdur. Əgər bir çoxluğun bütün elementləri, digər çoxluğa da daxildirsə, onda birinci çoxluğun ikinci çoxluğa daxil olunduğu qəbul olunur. Boş çoxluq ixtiyari çoxluğa daxildir.

Çoxluq tipi aşağıdakı formada təsvir olunur:

< tip adı > **=set of** < baza tipi >;

burada < tip adı > - identifikatordur, **SET**, **OF**-operatorun işçi sözləridir, < baza tipi > çoxluq elementlərinin baza tipidir. Baza tipi **WORD**, **INTEGER**, **LONGINT** tiplərindən başqa ixtiyari nizamlı tip ola bilər.

Məsələn,

```
type D1=set of '0..'9'; D2=set of 0..9;
var S1,S2,S3:D1;S4,S5,S6:D2;
begin
S1:=['1','2','3']; S2:=['3','2','1'];
S3:=['2','3'];S4:=[0..3,6];S5:=[4,5];
S6:=[3..9];
end.
```

Burada **S1** və **S2** çoxluqları ekvivalentdir, **S3** isə **S2**-yə daxildir, lakin onlar ekvivalent deyillər.

Çoxluqlar üzərində aşağıdakı əməliyyatlar nəzərdə tutulub:

1) Çoxluqların kəsişməsi: **A*B**. İki **A** və **B** çoxluqlarının kəsişməsi, eyni zamanda A və B çoxluqlarına aid olan elementlərdən təşkil olunmuş çoxluqdur. **S4*S6**-nın nəticəsi [3], **S4*S5**-nin nəticəsi isə boş çoxluqdur.

2) Çoxluqların birləşməsi: **A+B**. İki A və B çoxluqlarının birləşməsi, **A** və ya **B** çoxluqlarından heç olmasa birinə aid olan elementlərdən təşkil olunmuş çoxluqdur. Məsələn, **S4+S5**-in nəticəsi [0,1,2,3,4,5,6], **S5+S6**-nın nəticəsi [3,4,5,6,7,8,9].

3) Çoxluqların fərqi: **A-B**. **A** və **B** çoxluqlarının fərqi, **A** çoxluğunun B çoxluğuna aid olmayan elementlərindən təşkil olunmuş çoxluqdur. Məsələn, **S6-S5**-in nəticəsi [3,6,7,8,9], **S4-S5**-in nəticəsi [0,1,2,3,6].

4) Ekvivalentliyin yoxlanılması: **A=B**.Əgər çoxluqlar ekvivalentdirsə, nəticə **TRUE** olacaqdır, əks halda cavab **FALSE** olacaqdır.

5) Ekvivalent olmamasının yoxlanılması: **A<>B**. Əgər çoxluqlar ekvivalent deyillərsə, nəticə **TRUE** qiyməti olacaqdır, əks halda nəticə **FALSE** olacaqdır.

6) Daxil olmanın yoxlanılması:

a) **A<=B**. Əgər birinci çoxluq ikinciyə daxildirsə, nəticə **TRUE** əks halda **FALSE** olacaqdır.

b) **A>=B**. Əgər ikinci çoxluq birinciyə daxildirsə, nəticə **TRUE**, əks halda **FALSE** olacaqdır. Məsələn, tutaq ki,

```
var M:set of Byte;
  M:=[3,4,7,9];
```

onda `M=[4,7,3,3,9] → true; M<>[7,4,3,9] → false;`

`[3,4]<=M → true; []<=M → true;`

`M>=[1..10] → false; M<=[3..9] → true.`

7) Aid olmasının yoxlanması: **X IN A**. Bu əməliyyat çoxluğun baza tipi ilə üst-üstə düşən skalyar kəmiyyətlə, çoxluq arasındakı əlaqəni müəyyən edir. Əgər **X** qiyməti **A** çoxluğuna daxildirsə, əməliyyatın nəticəsi true,əks halda isə false olacaqdır. *Məsələn*, **3 in S6** –nın nəticəsi **TRUE**, **2*2 in S1**-in isə **FALSE** olacaqdır.

Bu əməliyyatlardan əlavə olaraq çoxluqlara aşağıdakı iki

proseduru da tətbiq etmək olar:

1) **INCLUDE** – çoxluğa yeni element daxil edir. Prosedura müraciətin ümumi şəkli aşağıdakı kimidir:

INCLUDE (S, I);

Burada S – TSetBase baza tipli elementlərdən ibarət çoxluq, I-isə bu tipə aid olan və çoxluqda daxil edilməsi tələb olunan ele-mentdir.

2) **EXCLUDE** – çoxluqdan elementi çıxarır. Ümumi şəkli aşağıdakı kimidir:

EXCLUDE (S, I)

burada S və I parametrləri **INCLUDE** prosedurunda olduğu kimidir:

Misal 1. 2-dən N-ə qədər **(1<N≤255)** natural ədədlər arasında 6-ya qalıqsız bölünən ədədlər çoxluğunu və 2-yə və ya 3-ə qalıqsız bölünən ədədlər çoxluğunu ayırmalı.

```
program p13;
const n=20;
var S2,S3,S6,S23:set of 2..n;
      k:byte;
begin S2:=[];S3:=[];
for k:=1 to n do
if k mod 2=0 then S2:=S2+[k];
if k mod 3=0 then S3:=S3+[k]
end;
S6:=N2*N3; S23:=S2+S3;
For k:=1 to n do
If k in S6 then write(k);
For k:=1 to n do
If k in S23 then write(k)
end.
```

Misal 2. 2-dən N-ə qədər **(1<N≤255)** natural ədədlər ardıcıllığından bütün sadə ədədləri tapmalı (Eratosfen əleyi).

```
program Eratosfen;
const n=201;
var A,B:set of 2..n;k,p:integer;
begin A:=[2..n];B:=[];p:=2;
repeat
```

```
while not(p in A)do p:=p+1;
B:=B+[p];k:=p;
while k<=n do
begin A:=A-[k];k:=k+p end
until A=[];
for p:=2 to n do if p in B then writeln(p)
end.
```

5.11. Sətirlər

String (sətir) tipi apostrof işarələri arasına alınan simvollar ardıcıllığından ibarətdir. Bu tip birölçülü simvol tipli **ARRAY [0..N] OF CHAR** massivinə oxşardır, lakin ondan fərqli olaraq sətir dəyişəndəki simvolların sayı **0**-dan **N**-ə qədər dəyişə bilir, burada **N**-sətirdəki simvolların maksimal sayıdır. N-in qiyməti **STRING [N]** tip elanı ilə təyin edilir və 255-dən çox olmamaq şərti ilə nizam tipli ixtiyari sabit ola bilər. Turbo Pascal dilində **N**-in qiymətini göstərməmək olar, bu sətrin uzunluğu maksimal mümkün, yəni **N=255** qəbul edilir. Sətir tipi ümumi şəkildə aşağıdakı kimi təsvir edilir:

var <identifikator>:string[<sətrin maksimal uzunluğu>]

Məsələn,

```
var Name:String[20];S:String;
```

Turbo Pascal dilində sətrə simvollar ardıcıllığı kimi baxılır. Sətir daxilindəki simvollar birdən başlayaraq nömrələnirlər, sətrin hər bir elementi sətrin adı və kvadrat mötərizədəki indeksi ilə göstərilə bilər. İndeks tam tipli müsbət sabit, dəyişən və ya ifadə ola bilər. İndeksin qiyməti sərhəddindən kənara çıxmamalıdır. *Məsələn*, **Name[5], Name[i], Name[k+1]** və s.

Sətirlərə birləşdirmə "+" əməliyyatını tətbiq etmək olur. *Məsələn,*

```
st:='a'+'b';
st:=st+'c';
```

Nəticədə alırıq **st→abc**. Əgər birləşdirilmiş sətrin uzunluğu maksimal uzunluğu aşırsa, onda "artıq" simvollar atılır.

Məsələn,

```
var st:string[1];
begin st:='123'; writeln(st)end.
```

proqramı nəticədə 1 simvolunu çap edəcək.

İki sətir arasındakı =, < >, >, <, >=, <= münasibət əməliyyatları, sətirdəki simvolların daxili kodlaşdırma nömrələrinin müqayisəsi ilə aparılır. Əgər bir sətir o birindən qısadırsa, qısa sətirdə çatmayan simvollar CHR (0) qiymətləri ilə əvəz edilir. Məsələn, aşağıdakı münasibət əməliyyatları TRUE qiymətini verəcək:

```
'A'>'1'; 'Turbo'<'TurboPascal'; 'cosm1'<'cos
m2'; 'pascal'>'PASCAL'; 'MSDOS'='MSDOS'.
```

Sətirlər və simvollar üzərindəki digər əməliyyatlar aşağıdakı standart prosedur və funksiyaların köməyi ilə həyata keçirilir:

1) `CONCAT(S1,S2,...,SN)` – `String` tipli funksiya olub, `S1,S2,-...,SN` sətirlərini bir sətirdə birləşdirir. *Məsələn,* `Concat('AA', 'BB', 'C');` nəticədə `'AABBC'` verir.

2) `COPY(ST,P,N)` – String tipli funksiya olub, `ST` sətrinin `N` sayda simvolunun `P` nömrəli simvoldan başlayaraq təkrarını alır. *Məsələn,* `ST:='ABCDEFG';Copy(ST,2,3);` nəticədə 'BCD' verər.

3) `DELETE(ST,P,N)` – proseduru, `ST` sətrinin `N` sayda simvolunu, `P` nömrəli simvoldan başlayaraq ləğv edir. Məsələn, `ST:='abcdefg'; Delete(St,3,2);` nəticədə `'abcdefg'` alınır.

4) `INSERT(SUBST,ST,P)` – proseduru, `SUBST` alt sətrini, `ST` sətrinə, onun `P` nömrəli simvolundan başlayaraq daxil edir. Məsələn, `ST:=TURBO; Insert('PASCAL',ST,6);` nəticə: `'TURBO PASCAL'`.

5) `LENGTH(ST)` – `INTEGER` tipli funksiya olub, `ST` sətrinin uzunluğunu təyin edir. Məsələn, `ST:='ALGORITM'; Length (ST);` nəticə: 8.

6) `POS(SUBST, ST)` – `INTEGER` tipli funksiya olub, `ST` sətrinə `SUBST` alt sətrinin birinci daxil olduğu mövqeyin nömrəsini təyin edir, əgər `SUBST` alt sətri `ST` sətrində tapılmazsa, funksiyanın nəticəsi sıfır olar. Məsələn,

`ST:='abcdef';POS('cd',ST);` nəticə: 3;

ST:='abcdef'; POS('k',ST); nəticə: **0.**

7) **STR(X[:N[:M]],ST)** –proseduru, ixtiyari həqiqi və ya tam tipli **X** ədədini, **WRITELN** prosedurunun etdiyi qaydada **ST** simvollar sətrinə çevirir. Burada məcburi olmayan **:N** və **:M** parametrləri çevirmənin formatını təyin edirlər, belə ki, **:N,X** ədədinin simvol ifadəsi üçün ayrılan ümumi sahəni, **:M** isə **X** ədədinin kəsr hissəsindəki (əgər **X** həqiqi tiplidirsə) simvolların sayını təyin edir.

8) **VAL(ST,X,CODE)** – proseduru, **ST** simvollar sətrini, tam və ya həqiqi tipli **X** dəyişəninin onun tipi ilə təyin edilən daxili ifadəsinə çevirir. Burada əgər çevirmə tam yerinə yetirilibsə, **CODE** parametri sıfır qiymətini alır və **X** dəyişəninə çevirmə nəticəsi mənimsədilir, əks halda **CODE** parametri, **ST** sətrində səhv simvolun tapıldığı mövqeyin nömrəsini alır və **X**-ın qiyməti dəyişməz qalır.

9) **UPCASE(CH)** – char tipli funksiya olub, latın əlifbasının kiçik hərfini bildirən **CH** simvol ifadəsini, onun əlifbadakı uyğun böyük hərfinə çevirir. Burada **CH** ixtiyari digər bir simvol olduqda, funksiya onu olduğu kimi saxlayır.

Misal.

```
var x:real;y:integer;st,st1:string;
begin
st:=concat('12','345');{12345}
st1:=copy(st,3,Length(st)-2);{345}
insert('-',st1,2);{3-45}
delete(st,pos('2',st),3);{15}
 str(pi:6:2,st);{3.14}
val('3.1415',x,y);{y=2,x="3.1415"}
end.
```

5.12. Alt proqramlar

Alt proqramlar proqramı bir-birindən müəyyən mənada müstəqil olan hissələrə bölməyə imkan verir. Bu birinci növbədə yaddaşa qənaət etməyə imkan verir, alt proqram bir dəfə yazılır, lakin ona ixtiyari sayda müraciət etmək olur. Alt proqramların daxilində öz növbəsində başqa alt proqramlar verilə bilər. Alt

proqramlara müraciət üçün prosedurun çağırış operatorunda prosedurun çağırış adını və ya ifadədə funksiyanın adını vermək kifayətdir. Turbo Pascal dilində iki cür alt proqram – prosedur və funksiyalardan istifadə edilir. Burada funksiya prosedurdan yalnız onunla fərqlənir ki, funksiyanı təşkil edən operatorların yerinə yetirilməsinin nəticəsi yeganə qiymət və ya göstərici olur. Alt proqram başlıq və gövdədən ibarətdir. Başlıqda alt proqramın adı və formal parametrlər elan olunur. Funksiya üçün başlıqda alınan nəticənin tipi də göstərilir. Başlığın ardınca alt proqramın gövdəsi gəlir, o da əsas proqram kimi təsvirlər və operatorlar bölməsindən ibarətdir. Alt proqramın təsvirlər bölməsində daha aşağı səviyyəli alt proqramların təsviri, onlarda isə digər alt proqramların təsviri və s. ola bilər. İxtiyari alt proqramdan istifadə edilməmişdən qabaq o, təsvir olunmalıdır. Buna görə də aşağı səviyyədə olan alt proqramdan yuxarıdakı alt proqrama müraciət etmək olur, lakin bunun əksini etmək mümkün deyildir. Yəni alt proqrama bu alt proqramın təsvirindən əvvəl təsvir olunmuş yuxarı səviyyə obyektlərə müraciət etmək olur. Bu cür obyektlər alt proqrama nəzərən qlobal adlandırılır. Turbo Pascalda təsvirlər bölməsində sabitlərin, dəyişənlərin, tiplərin, nişan və alt proqramların təsvirlərinin verilmə ardıcıllığı ixtiyari ola bilər.

Prosedurun başlığı aşağıdakı şəkildədir:

Procedure <ad> (formal parametrlərin siyahısı) ;

Funksiyanın başlığı isə aşağıdakı şəkildədir:

Function <ad> (formal parametrlərin siyahısı) : <tip>;

Burada <ad> – alt proqramın adını bildirən identifikatordur, <formal parametrlərin siyahısı> – alt proqramın formal parametrlərinin siyahısını təşkil edir, <tip> – funksiyanın nəticəsinin tipini bildirir. Alt proqramın başlığının ardınca aşağıdakı standart direktivalardan ixtiyari biri gələ bilər: **assembler**, **external**, **far**, **forward**, **inline**, **interrupt**, **near**. Bu direktivalar kompilyatorun işini dəqiqləşdirir və yalnız bu alt proqrama aid olur, ondan sonra gələ bilən alt proqramlara aid olmur. **Assembler** direktivası prosedura giriş və çıxış zamanı yaranan maşın göstərişlərinin standart ardıcıllığını ləğv edir. Bu zaman alt

proqramın gövdəsi daxili **assembler** əmrlərinin köməyilə yerinə yetirilməlidir. **External** direktivası ilə xarici alt proqram elan edilir. **Far** – çağırışın uzaq modelinə əsaslanan alt proqram kodu yaratmaq haqqında kompilyatora göstəriş verir. **Near** – direktivası isə çağırışın yaxın modelinə əsaslanan alt proqram kodu yaratmaq haqqında kompilyatora göstəriş verir. Proqramlarda iki yaddaş modelindən: yaxın və uzaq modellərindən istifadə edilə bilər. Yaddaş modeli proqramın müxtəlif yerlərindən proseduraların çağırış imkanlarını müəyyən edir. Əgər yaxın modeldən istifadə edilirsə, çağırış yalnız 64 *kbayt* ətrafında mümkündür, uzaq model halında isə çağırış ixtiyari hissədən mümkün olur. **forward** – direktivası istifadə edilən alt proqramın təsvirinin proqram mətnində sonradan veriləcəyini kompilyatora xəbər verir. **inline** – göstərir ki, alt proqramın gövdəsi daxili maşın göstərişlərinin köməyi ilə idarə olunur. **interrupt** – kəsilmələrin emalı prosedurlarının yaradılması üçün istifadə olunur.

Başlıqdakı formal parametrlər siyahısı verilməyə də bilər. Əgər bu siyahı verilirsə, onda burada formal parametrlərin adları və onların tipləri göstərilməlidir. *Məsələn*,

```
Procedure S(a:real;b:integer;c:char);
Function F(a,b:real):real;
```

Siyahıdakı parametrlər bir-birindən " ; " ilə ayrılır və eyni tipli parametrlər alt siyahılarda birləşdirilə bilər. Alt proqramın gövdə operatorları formal parametrlər siyahısını təsvirlər bölməsinin genişlənməsi kimi nəzərdən keçirir, belə ki, bu siyahıdakı bütün dəyişənlər alt proqram daxilində ixtiyari ifadələrdə istifadə edilə bilər. Alt proqrama müraciət onun adı üzrə yerinə yetirilir, addan sonra mötərizədə formal parametrlərin yerinə qoyulacaq faktiki parametrlər verilir. Turbo Pascal dilində formal parametrlərin sayı və tipi, alt proqrama müraciətdəki faktiki parametrlərin sayı və tipinə uyğun gəlməlidir. İstifadə olunan faktiki parametrlərin mənası, alt proqrama müraciətdə onların hansı ardıcıllıqla verilməsindən asılıdır. Alt proqramın ixtiyari formal parametri, ya parametr-qiymət, ya parametr-dəyişən və ya nəhayət parametr-sabit ola bilər. Parametr dəyişənlərin qarşısında **VAR** işçi sözünü, parametr-sabitlərin qarşısında isə **CONST** sözünü

vermək lazımdır. *Məsələn*,

```
procedure k(var a:real; b:real;
                          const c:string);
```

burada **a** – parametr-dəyişən, **b** – parametr-qiymət, **c** – isə parametr-sabitdir.

Formal parametr, parametr-dəyişən kimi elan edilibsə, onda alt proqrama müraciətdə ona eyni tipli dəyişən formasında faktiki parametr uyğun gəlməlidir. Formal parametr parametr-qiymət və ya parametr-sabit kimi elan edilibsə, onda alt proqrama müraciətdə ona ixtiyari ifadə uyğun gətirilə bilər. Bu qaydalara riayət olunması Turbo Pascal-ın kompilyatoru tərəfidən nəzarətdə saxlanılır. Əgər parametr, parametr-qiymət kimi təyin edilibsə, onda alt proqrama müraciətdən əvvəl, bu qiymət hesablanır, alınan nəticə müvəqqəti yaddaşa köçürülür və alt proqrama ötürülür. Parametr, parametr-dəyişən kimi təyin edildikdə, alt proqrama müraciət zamanı alt proqrama dəyişənin özü (təkrarı deyil) ötürülür və parametr-dəyişənin dəyişməsi, çağıran proqramdakı faktiki parametrin dəyişməsinə gətirir.

Parametr-sabit halında da alt proqrama dəyişənin və ya hesablanmış qiymətin yerləşdiyi yaddaş oblastının ünvanı verilir. Lakin parametr-sabitə alt proqram daxilində yeni qiymətin mənimsədilməsinin qarşısı kompilyator tərəfindən alınır.

Misal 1. Birinci N natural ədədin cəmini və hasilini tapmalı.

```
program p20;
varn,s,p:integer;
procedure prim(k:integer;var x,y:integer);
var i:integer;
begin x:=0; y:=1;
for i:=1to k do begin x:=x+i; y:=y*i end
end;
begin read(n); prim(n,s,p);
writeln('s=',s); write('p=',p)
end.
```

Misal 2. ƏBOB-un tapılması üçün Evklid alqoritmi.

```
program p21;
var a,b,c:integer;
```

```
procedure Evklid(m,n:integer;var
k:integer);
begin while m<>n do
              if  m>n then m:=m-n else
                                  n:=n-m; k:=m
end;
begin write('a='); readln(a); write('b=');
readln(b);
Evklid(a+b,abs(a-b),c);
Evklid(c,a*b,c); writeln('c=',c)
end.
```

Misal 3. Misal 2-ni Function alt proqram ilə həll edək.

```
program  p22;
var a,b,c:integer;
function Evklid(m,n:integer):integer;
begin while m<>n do
if m>n then m:=m-n else n:=m;
Evklid:=m
end;
begin  write('a=');readln(a);  write('b=');
readln(b);
c:=Evklid(Evklid(a+b,abs(a-b)),a*b);
writeln('c=',c)
end.
```

Formal parametrlər siyahısındakı ixtiyari parametrin tipi yalnız standart və ya əvvəlcədən elan edilmiş tip ola bilər. Buna görə də məsələn, aşağıdakı proseduru elan etmək olmaz:

```
procedure s(a:array[1..10]of real);
```

Alt proqrama massiv ötürülürsə, bu massivləri əvvəlcədən təsvir etmək tələb olunur. Məsələn,

```
type mas=array[1..10]of real;
procedure s(a:mas);
```

Sətir də faktiki olaraq massiv olduğundan, onun da alt proqrama ötürülməsi analoji qaydada aparılır:

```
type st=string[15];st1=string[30];
function s(a: st):st1;
```

Funksiya və prosedurlardan, digər prosedur və funksiyalara müraciət zamanı faktiki parametr kimi istifadə edilməsini təmin etmək üçün prosedur tiplərdən istifadə edilir. Prosedur tipi elan etmək üçün adı göstərilməyən prosedur (funksiya) başlıqlarından istifadə edilir:

```
type p1=procedure(a,b,c:real;d:real);
     p2=procedure(var a,b);p3=procedure;
          f1=function:string;
          f2=function(var s:string):real;
```

Proqramda prosedur tiplərin dəyişənləri də elan edilə bilər. *Məsələn*,

```
var k:p1;L1,L2:f2;
    ap:array[1..N]of p1;
```

Prosedur tip dəyişənlərə, qiymət kimi uyğun alt proqramların adları mənimsədilə bilər.

Məsələn,

```
type Proc=Procedure(n:word; var a:byte);
var p1:proc;x,y:byte;
Procedure Proc1 (x:word; var y:byte);far;
begin if x>255 then y:=x mod 255 else
y:=Byte(x)
end;
begin p1:=proc1;
for x:=150 to 180 do
begin p1(x+100,y); write(y:8)
end
end.
```

Turbo Pascal dilində qeyri-tip parametrlərdən də istifadə edilə bilir. Parametr o vaxt qeyri-tip sayılır ki, alt proqramın başlığında formal parametr-dəyişənin tipi göstərilməsin, ona uyğun faktiki parametr ixtiyari tipli dəyişən ola bilər. Qeyd edək ki, qeyri-tip yalnız parametr-dəyişənlər ola bilər. Qeyri-tip parametrlərdən, verilənlərin tipi vacib olmadığı halda istifadə edilir.

Rekursiya – alt proqramın onu təşkil edən operatorların yerinə yetirilməsi prosesində özü-özünə müraciətdir. *Məsələn*,

```
program Factorial;
```

```
var n:integer;
Function Fac(n:integer):extended;
var F:extended;
begin
if n=0 then Fac:=1 else begin F:=Fac(n-1);
Fac:=F*n end
end;
begin readln(n);Writeln('n!=',Fac(n))end.
```

Rekursiv müraciət dolayısı yolla verilə bilər. Bu halda alt proqram özünə, ona müraciət olan digər alt proqramı çağırmaqla müraciət edilir. *Məsələn,*

```
Procedure A(i:byte);
begin
.....
B(i);
.....
end;
Procedure B(j:byte);
.....
begin
.....
A(j);
.....
end;
```

Lakin ixtiyari identifikatordan istifadə etməmişdən əvvəl onu təsvir etmək tələb olunduğundan bu cür proqram konstruksiyasından istifadə etmək olmaz. Bu cür müraciətin mumkun olması üçün qabaqlayıcı təsvirdən istifadə edilir:

```
Procedure B(j:byte);
Procedure A(i:byte);
begin
.....
B(i);
```

```
.....
end;
Procedure B;
begin
.....
A(j);
.....
end;
```

5.13. Fayllar

Fayl dedikdə biz, kompüterin xarici yaddaşının adlandırılmış oblastını yaxud informasiya qaynağı və ya qəbuledicisi olan məntiqi qurğu başa düşürük. İxtiyari faylın üç xarakter əlaməti var. Birinci addır ki, proqrama eyni zamanda bir neçə faylla işləməyə imkan verir. İkincisi, hər bir fayl yalnız eyni bir tip elementlərdən ibarət olur və bu tip fayllardan başqa Turbo Pascal-ın ixtiyari tipi ola bilər. Üçüncüsü isə odur ki, yeni yaradılan faylın uzunluğu qabaqcadan elan edilmir və yalnız xarici yaddaş qurğularının həcmi ilə məhdudlaşdırılır.

Fayl tipini və ya fayl tipli dəyişəni aşağıdakı üç üsuldan biri ilə vermək olar:

```
<ad>= file of <tip>;
<ad >=text;
<ad >=file;
```

Burada < ad > faylın tipinin adı, **file, of**-operatorun işçi sözləri, **TEXT**-mətn fayllarının standart tipinin adı, < tip > - fayl tipindən başqa Turbo Pascal dilinin ixtiyari tipidir. *Məsələn,*

```
type p=record
        name:string;
        code:word;
        cost:comp
end;
t1= file of string[80];
```

```
var f1:file of char:f2:text;f3:file;
      f4:t1;f5:file of p;
```

Elan üsulundan asılı olaraq faylların aşağıdakı üç formasını göstərmək olar:

1) tip fayllar (**file of** ilə verilir);

2) mətn faylları (**text** tipi ilə təyin edilir);

3) qeyri-tip fayllar (**file** tipi ilə təyin edilir).

Misalda **f1**, **f2**, **f4**, **f5**-tip fayllar, **f2**-mətn faylı, **f3** isə qeyri-tip fayldır. Fayl forması ümumiyyətlə informasiyanın faylda saxlanılması qaydasını təyin edir. İxtiyari proqram qabaqcadan elan edilmiş standart fayl dəyişənli iki fayl: **input** (verilənlərin klaviaturadan oxunması üçün) və **output** (verilənlərin ekrana çıxarılması üçün) fayllarından istifadə edə bilir. Yerdə qalan ixtiyari fayllardan, eləcə də məntiqi qurğulardan proqramda, onların açılmasını təmin edən xüsusi prosedurların yerinə yetirilməsindən sonra isitfadə etmək olar. Bu prosedura əvvəlcədən elan edilmiş fayl dəyişənin mövcud və ya yaradılan faylın adı ilə əlaqələndirilir, eləcə də informasiya mübadiləsinin istiqamətini: fayldan oxunmaq və ya fayla yazılış olacağını təyin edir.

Fayl dəyişəni, fayl adı ilə **assign** standart proseduru vasitəsilə əlaqə yaradır:

assign (<fayl dəyişəni>, < fayl adı və ya məntiqi qurğu>) ;

burada <fayl dəyişəni> - proqramda fayl tipli dəyişən kimi elan edilmiş identifikatordur, < fayl adı və ya məntiqi qurğu> - fayl və ya məntiqi qurğu adından ibarət məntiqi ifadədir. Əgər fayl adı boş sətir şəklində verilirsə, məsələn, **assign(f,' ')**, onda informasiya mübadiləsinin istiqamətindən asılı olaraq fayl dəyişəni **input** və ya **output** standart faylı ilə əlaqə yaradır.

Fayl adı aşağıdakı şərtləri ödəyən ixtiyari sətir tipli ifadədir:

1) ad – böyük və kiçik latın hərfləri, rəqəmləri və **!,@,#,$,%,^, &,(),`,~,-, _** işarələrinin 8-ə qədər ixtiyari ardıcıllığıdır, ad bu simvollardan ixtiyari biri ilə başlaya bilər, addan sonra faylın üçdən çox olmayan sayda ad genişlənməsi

verilə bilər, ad genişlənməsi addan nöqtə ilə ayrılır. Addan əvvəl isə fayla gedən yol verilə bilər. Bu yol disk adından və (və ya) cari kataloq adından və daha yüksək səviyyəli kataloq adından ibarət olur. Disk adı A...Z simvollarından ixtiyari biri ola bilər və ondan sonra " : " işarəsi qoyulur. Burada A:, B: adları elastik disklərə, C:, D:, E: və s. isə bərk disklərə aiddir. Disk adı verilmədikdə cari disk qəbul edilir. Disk adından sonra faylın yerləşdiyi kataloq adı verilə bilər, əgər kataloq adından əvvəl "\" işarəsi varsa, deməli fayla yol diskin baş kataloqundan başlayır, bu işarə yoxdursa, onda yol cari kataloqdan başlayır. Kataloq adından sonra aşağı səviyyəli bir və ya bir neçə kataloq adı verilə bilər, bütün bu adlar bir-birindən "\" işarəsi ilə ayrılır. Adın yol ilə maksimal uzunluğu 79 simvoldur. *Məsələn,*

```
var f1: text;f2:file of string;
const name='c:\k1\k2\t3.txt';
begin assign(f1,'d1.dat');
          assign(f2,name)
end.
```

Kompüterin klaviatura, displey ekranı, printer və giriş-çıxış kanalları kimi standart aparat qurğuları Turbo Pascal-da məntiqi qurğular adlanan xüsusi adlarla təyin edilirlər. **con** – klaviatura və ya displey ekranını təyin edən məntiqi addır. Turbo Pascal-da bu fiziki qurğular arasında verilənlərin ötürülməsi istiqaməti üzrə fərq qoyulur: verilənləri yalnız klaviaturadan oxumaq və verilənləri yalnız ekrana çıxarmaq olur. **prn** – printerin məntiqi adıdır. Əgər kompüterə bir neçə printer qoşulubsa, onda onlarla əlaqə **lpt1, lpt2** və **lpt3** məntiqi adları üzrə yerinə yetirilir. **turbo.tpl** kitabxanasına aid olan **printer** standart kitabxana modulu **lst** fayl dəyişəninin adını elan edir və onu **lpt1** məntiqi qurğusu ilə əlaqələndirir. Bu isə printerə sadə müraciət təşkil etməyə imkan verir. *Məsələn,*

```
Uses Printer;
 begin writeln (lst,'TURBO PASCAL')
end.
```

aux – kompüterin digər maşınlarla əlaqəsinin yaradılması üçün istifadə olunan məntiqi addır. Bu ada uyğun kanal verilənlərin həm qəbulunu, həm də verilməsini təmin edə bilir. Lakin

proqramda o, zamanın bir anında bu funksiyalardan yalnız birini icra edə bilir. Adətən, kompüterdə bu cür iki kanal olur və onlara **COM 1** və **COM 2** məntiqi qurğu adları verilir. **NUL** – «boş» qurğunun məntiqi adıdır. Bu qurğu qeyri-məhdud həcmli informasiyanın qəbuledici qurğusu kimi istifadə edilir. **NUL** qurğusuna informasiya mənbəyi kimi müraciət edildikdə faylın sonu əlaməti **EOF** qiyməti alınır.

Məntiqi qurğu ilə faylın dəyişəni **assign** proseduru ilə əlaqələndirilir. *Məsələn*,

```
var f1,f2:text;
 begin assign(f1,'AUX');
       assign(f2,'LPT2')
       end.
```

Turbo Pascal dilində faylı oxumaq üçün, fayla informasiya yazmaq üçün, eləcə də eyni zamanda həm informasiya oxumaq və yazmaq üçün açmaq olur. Fayldakı informasiyanı oxumaq üçün **reset** standart prosedurundan istifadə olunur:

reset (<fayl dəyişəni>);

burada <fayl dəyişəni> - əvvəlcədən **assign** prosedurunun köməyi ilə artıq mövcud fayl və ya informasiya qəbuledici məntiqi qurğusu ilə əlaqələndirilmiş fayl dəyişənidir. Bu prosedur yerinə yetirilərkən, disk faylını və ya məntiqi qurğunu informasiyanın oxunmasına hazırlaşır. Nəticədə xüsusi göstərici dəyişən faylın başlanğıcını, yəni sıfır sıra nömrəli elementi göstərəcəkdir.

Turbo Pascal-da **reset** proseduru ilə informasiyanın yalnız oxunması üçün açılmış tip fayllara **write** (yəni informasiyanın yazılması üçün) proseduru ilə müraciət etməyə icazə verilir. Bu isə əvvəlcədən yaradılmış tip faylları asanlıqla yeniləşdirməyə və ehtiyac olduqda onları genişləndirməyə imkan verir. **reset** proseduru ilə açılmış mətn fayllarına **write** və ya **writeln** prosedurlarını tətbiq etmək olmaz.

rewrite (<fayl dəyişəni>);

standart proseduru <fayl dəyişəni> ilə əvvəlcədən əlaqələndirilmiş fayl və ya məntiqi qurğuya informasiyanın yazılışını təmin edir. Bu prosedura ilə əvvəlcədən mövcud olan disk faylına

informasiya yazmaq olmaz, belə ki, bu prosedurun yerinə yetirilməsi zamanı köhnə fayl ləğv edilir, yeni fayl informasiya qəbuluna hazırlaşdırılır və onun göstərici-dəyişəni sıfır qiymətini alır.

append (<fayl dəyişəni>);

standart proseduru əvvəlcədən mövcud olan mətni faylın genişləndirilməsi üçün ona informasiya yazılışını təmin edir və bu zaman göstərici faylın sonuna yerləşdirilir. Bu prosedurla tip və ya qeyri-tip fayllara informasiya yazmaq olmaz. Əgər mətni fayl əvvəlcədən **reset** və ya **rewrite** prosedurları ilə açılmışdırsa, onda **append** prosedurunun tətbiqi bu faylın bağlanmasına gətirəcək və bundan sonra fayl onun genişləndirilməsi üçün yenidən açılacaqdır.

İxtiyari tip fayllara tətbiq edilə bilən prosedur və funksiyalara baxaq:

1) **close** proseduru faylı bağlayır, bu zaman əvvəlcədən **assign** proseduru ilə fayl adı ilə fayl dəyişəni arasında yaradılmış əlaqə saxlanılır. Prosedurun formatı **close** (<fayl dəyişəni>); şəklindədir. Yeni faylın yaradılması və köhnə faylın genişləndirilməsi zamanı bu prosedura fayldakı bütün yeni yazılışları saxlayır və faylı kataloqda registrasiya edir. Bu prosedur yerinə yetirildikdən sonra faylla, fayl dəyişəni arasındakı əlaqə saxlandığından, həmin faylı əlavə **assign** prosedurundan istifadə etmədən yenidən açmaq olar.

2) **rename** proseduru faylın adını dəyişdirir. Prosedurun formatı **rename** (<fayl dəyişəni>, <yeni ad>); şəklindədir. Burada <yeni ad> -faylın yeni adını bildirən sətir ifadəsidir. Əgər fayl qabaqcadan **reset, rewrite** və ya **append** prosedurları ilə açılıbsa, onda **rename** prosedurunun yerinə yetirilməsindən əvvəl faylı bağlamaq lazımdır.

3) **erase** proseduru faylı ləğv edir. Prosedurun formatı aşağıdakı kimidir:

erase (<fayl dəyişəni>);

Əgər fayl qabaqcadan **reset, rewrite, append** prosedurları ilə açılıbsa, onda **erase** prosedurunun yerinə yetirilməsindən əvvəl faylı bağlamaq lazımdır.

4) **flush** proseduru faylın daxili buferini təmizləyir və beləliklə, diskdəki fayldan yerinə yetirilmiş axırıncı dəyişikliklərin saxlanılmasını təmin edir. Prosedurun formatı **flush**(<fayl dəyişəni>) ; formasındadır.

5) **eof**(<fayl dəyişəni>) :**Boolean** funksiyası faylın sona çatıb-çatmamasını müəyyən edən məntiqi funksiyadır. Fayl göstəricisi faylın sonunda olduqda **EOF** funksiyası **TRUE** qiymətini alır. Yazılış zamanı bu növbəti elementin faylın sonuna əlavə ediləcəyini, oxunuş zamanı isə faylın sonuna çatdığını bildirir.

6) **chdir** proseduru cari kataloqu dəyişdirir. Formatı: CHDIR (<yol>); burada <yol>-yeni kataloqa gedən yolu göstərən sətir ifadəsidir.

7) **getdir** proseduru cari kataloqun adını təyin edir. Formatı: **GETDIR** (<qurğu>, <kataloq>); burada <qurğu>–qurğunun nömrəsini: **0**- susmaqla qurğu, **1**-A diski, **2**-B diski və s. bildirən **WORD** tipli ifadə, <kataloq>-göstərilən diskdəki cari kataloqa gedən yolu bildirən **STRING** tipli dəyişəndir.

8) **mkdir** proseduru göstərilən diskdə yeni kataloq yaradır. Formatı: **mkdir** (<kataloq>); burada <kataloq>-kataloqa gedən yolu təyin edən, **string** tipli ifadədir.

9) **rmdir** proseduru kataloqu ləğv edir. Formatı: **rmdir** (<kataloq>); burada <kataloq> - ləğv edilən kataloqa gedən yolu təyin edən, **string** tipli ifadədir. Qeyd edək ki, ləğv edilən kataloq boş olmalıdır.

10) **ioresult:word** funksiyası axırıncı giriş-çıxış əməliyyatının şərti əlamətini verir. Əməliyyat müvəffəqiyyətlə sona çatarsa, funksiya sıfır qiymətini alar.

11) **diskfree** (<disk>): **longint** funksiyası göstərilən diskdə azad sahənin həcmini baytlarla göstərir.

12) **disksize** (<disk>):**longint** funksiyası göstərilən diskin tam həcmini baytlarla göstərir.

13) **findfirst** proseduru göstərilən kataloqda qeydiyyatdan keçmiş fayllardan birincisinin atributlarını müəyyən edir. Formatı: **findfirst** (<maska>, <atributlar>, <ad>); burada

<maska> faylın maskasını ifadə edən sətir ifadə, <atributlar> maskanı dəqiqiləşdirən **byte** tipli ifadə, <ad> isə faylın adı qaytarılan **searchrec** tipli dəyişəndir. Fayl maskasının formalaşdırılmasında aşağıdakı simvollardan istifadə olunur: *-bu işarə onun durduğu yerdə fayl adı və ya ad genişləndirilməsinə aid ixtiyari sayda simvolun yerləşdirilə biləcəyini göstərir, ?- işarəsi bildirir ki, bu yerdə icazə verilmiş bir simvol yerləşdirilə bilər. Məsələn, *.* maskası kataloqdakı bütün faylları seçir, C*.* isə C ilə başlayan bütün faylları seçir və s. Maskadan qabaq fayla gedən yol gətirilir. Burada <atributlar>-**findfirst** proseduruna müraciət zamanı hansı fayllardan istifadə etməyin mümkün olduğunu təyin edir. DOS.TPU modulunda fayl atributları aşağıdakı kimi elan edilir:

```
Const readonly=$01; - yalnız oxumaq üçün,
      hidden=$02; - gizlədilmiş fayl,
     sysfile=$04; - sistem faylı,
     volumeid=$08; - cild identifikatoru,
     directory=$10; - alt kataloqun adı,
     archive=$20; - arxiv faylı,
     anyfile=$3F; - ixtiyari fayl.
```

findfirst prosedurunun nəticəsi **searchrec** tip dəyişən qaytarır. Bu tip **DOS.TPU** modulunda aşağıdakı kimi təyin edilir:

```
type Searchrec=record
        fill:array[1..21]of Byte;
        attr:Byte;
        time:Longint;
        size:Longint;
        name:String[12]
end;
```

14) **findnext** proseduru kataloqdakı növbəti faylın adını təyin edir. Formatı: **findnext** (<fayl adı>); burada <fayl adı> - **SEARCHREC** tipli yazılışdır. Yazılışda fayl haqqında informasiya verilir.

15) **getftime** proseduru faylın yaradıldığı və ya axırıncı yerləşdirildiyi tarixi təyin edir. Formatı: **getftime** (<fayl dəyi-

şəni>, <tarix>); burada <tarix> - **longint** tipli dəyişəndir.

16) **setftime** proseduru faylın yaradılmasını və ya təzələnməsinin yeni tarixini təyin edir. Formatı: **setftime** (<fayl dəyişəni>, <tarix>); burada <tarix> -zaman və tarixi bildirir.

17) **getfattr** proseduru fayl atributlarını almağa imkan verir. Formatı: **getfattr** (<fayl dəyişəni>, <atributlar>); burada <atributlar>- Word tipli dəyişəndir.

18) **setfattr** proseduru fayl atributlarını təyin etməyə imkan verir. Formatı: **setfattr** (<fayl dəyişəni>, <atributlar>);

19) **fsearch:pathstr** funksiyası kataloqlar siyahısında fayl axtarır. Formatı: **fsearch** (<ad>, <kataloqlar siyahısı>); burada <ad>-axtarılan faylın adı, <kataloqlar siyahısı>- faylın axtarıldığı kataloqların siyahısı.

20) **fsplit** proseduru fayl adını hissələrinə ayırır, yəni fayla gedən yolu, onun adını, ad genişlənməsini ayrı-ayrı parametrlər şəklində qaytarır. Formatı: **fsplit** (<fayl>, <yol>, <ad>, <ad genişlənməsi>);

21) **fexpand:pathstr** funksiyası fayl adına ona gedən yolu və qurğunun adını əlavə edir. Formatı: **fexpand** (<fayl>);

Mətn faylları, **text** standart tipinə aid olan fayl dəyişənləri ilə əlaqə yaradır. Mətni fayllar, mətni informasiyanın saxlanılması üçün nəzərdə tutulub. Məsələn, proqram mətnləri, məhz bu tip fayllarda saxlanılır. Mətn fayl elementləri (yazılışları) dəyişən uzunluqlu ola bilər. Turbo Pascal-da mətni fayl dəyişən uzunluqlu sətirlər külliyatı kimi qəbul olunur. Hər bir sətrə birincidən başlayaraq ardıcıl keçmək olar.

Mətni fayl yaradılarkən hər bir sətrin sonunda xüsusi **eoln** (End of Line – sətir sonu) əlaməti, faylın sonunda isə **eof** (End of File – faylın sonu) əlaməti qoyulur. Fayldakı sətirlərə (yazılışlara) **Read**, **Readln**, **Write**, **Writeln** prosedurları ilə keçmək olar. Bu prosedurlar fayllara dəyişən saylı faktiki parametrlərlə müraciət edə bilir və parametr kimi simvol, sətir və ədədlərdən istifadə edilə bilər. Bu prosedurlarda birinci parametr fayl dəyişəni ola bilər. Bu halda disk faylına və ya məntiqi qurğuya müraciət yerinə yetirilir. Əgər fayl dəyişəni göstərilməyibsə, onda **input** və

output standart fayllarına müraciət yerinə yetirilir.

READ proseduru simvol, sətir və ədədlərin daxil edilməsini təmin edir və aşağıdakı formata malikdir:

READ (<fayl dəyişəni>, <daxiletmə siyahısı>);

və ya

READ (<daxil etmə siyahısı>);

burada <daxil etmə siyahısı>-**char, string** tipli bir və ya daha artıq dəyişənlər, eləcə də ixtiyari tam və ya həqiqi tipli dəyişənlər ardıcıllığıdır.

READLN proseduru da simvol, sətir və ədədlərin daxil edilməsini təmin edir və **READ** porseduru ilə eynigüclüdür. Fərq yalnız ondadır ki, burada sətirdəki axırıncı dəyişən oxunduqdan sonra sətrin **EOLN** əlamatinə qədər qalan hissəsi buraxılır və buna görə də **READLN** və ya **READ** prosedurlarına növbəti müraciət yeni sətrin birinci simvolundan başlanır. Bundan əlavə bu proseduru <daxiletmə siyahısı> parametrini vermədən də tətbiq etmək olar. Bu isə cari sətirdəki **EOLN** əlamətinə qədər olan bütün simvolların buraxılmasına gətirib çıxarır.

WRITE proseduru mətni informasiyanın mətni fayla və ya məntiqi qurğuya ötürülməsini təşkil edir və aşağıdakı formata malikdir:

WRITE (<fayl dəyişəni>, <xaricetmə siyahısı>);

və ya

WRITE (<xaricetmə siyahısı>);

burada <xaricetmə siyahısı>-**CHAR, STRING, BOOLEAN** tipli bir və ya daha artıq dəyişənlər, eləcə də ixtiyari tam və ya həqiqi tipli dəyişənlər ardıcıllığıdır. Burada <fayl dəyişəni> varsa, o əvvəlcədən **TEXT** tipli dəyişən kimi təsvir olunmalı və **ASSIGN** proseduru ilə fayl adı və ya məntiqi qurğu ilə əlaqələndirilməlidir.

WRITELN proseduru, **WRITE** proseduru ilə üst-üstə düşür. Yeganə fərq ondadır ki, xaric olunan simvollar sətri **CR** və **LF** kodları ilə qurtarır. Bu prosedurda <xaricetmə siyahısı>nı verməmək də olar. Bu halda kursor növbəti sətrin əvvəlinə keçirilir.

EOLN məntiqi funksiyası mətni faylda sətrin sonuna çat-

dıqda **TRUE** qiymətini verir və aşağıdakı formata malikdir:

EOLN (<fayl dəyişəni>);

Əgər burada <fayl dəyişəni> parametri verilmirsə, onda funksiya standart **INPUT** faylını yoxlayacaqdır.

SEEKEOLN məntiqi funksiyası sətir sonluğu əlaməti **EOLN** və ya birinci əhəmiyyətli işarəyə rast gəlinən bütün probel və tabulyasiya işarələrini buraxır və **EOLN** əlamətinə rast gəldikdə **TRUE** qiymətini alır. Fomatı aşağıdakı şəkildədir:

SEEKEOLN (<fayl dəyişəni>);

SEEKEOF məntiqi funsiyası fayl sonluğu əlaməti **EOF** və ya birinci əhəmiyyətli işarəyə qədər rast gəlinən bütün probel, tabulyasiya və sətir sonluğu **EOLN** işarələrini buraxır və **EOF** əlamətinə rast gəldikdə **TRUE** qiymətini alır. Formatı aşağıdakı kimidir:

SEEKEOF (<fayl dəyişəni>);

Misal 1. Note.txt mətni faylındakı mətnin neçə sətirdən ibarət olduğunu təyin etməli.

```
var Note:text; k:integer;
begin assign(Note,'Note.txt');
reset(Note); k:=0;
while not eof(Note) do
begin readln(Note); k:=k+1 end;
writeln(k); close(Note)
end.
```

Misal 2. Note.txt mətni faylındakı mətndə ən uzun sətri təyin etməli.

```
var Note: text; max, k: integer; c:char;
begin assign(Note,'Note.txt');
reset (Note); max:=0;
while not eof(Note) do
begin k:=0;
while not eoln(Note) do
begin read(Note,c); k:=k+1 end;
if k>max then max:=k; readln(Note)
end;
writeln(max); close(Note)
```

end.

Tip fayllarının ixtiyari elementinin uzunluğu ciddi sabitdir, bu isə onların hər birinə birbaşa müraciəti təşkil etməyə imkan verir (yəni elementə onun sıra nömrəsinə görə müraciət etmək olur). Giriş-çıxış prosedurlarına birinci müraciətdən əvvəl fayl göstəricisi faylın əvvəlində durur və sıfır nömrəli birinci elementi göstərir. Hər bir oxunma və ya yazılışdan sonra göstərici faylın növbəti elementinə doğru hərəkət edir. Giriş-çıxış siyahılarındakı dəyişənlər, fayl elementləri ilə eyni tipə aid olmalıdır.

READ proseduru tip faylının növbəti elementlərinin oxunmasını təmin edir və formatı aşağıdakı kimidir:

READ (<fayl dəyişəni>, <daxil etmə siyahısı>); burada <daxil etmə siyahısı>sı fayl elementləri ilə eyni tipli olan bir və ya bir neçə dəyişənlərdən ibarət siyahıdır. Burada <fayl dəyişəni> əvvəlcədən **FILE OF** ...cümləsi ilə elan edilməli və **ASSIGN** proseduru ilə faylın adı ilə əlaqələndirilməlidir. Fayl **RESET** proseduru ilə açılmalıdır.

WRITE proseduru tip faylına verilənlərin yazılışı üçün istifadə edilir və aşağıdakı formata malikdir:

WRITE (<fayl dəyişəni>, <xaric etmə siyahısı>); burada <xaric etmə siyahısı> -fayl elementləri ilə eyni tipli olan bir və ya bir neçə ifadədən ibarət siyahıdır.

SEEK proseduru fayl göstəricisini tələb olunan elementə doğru sürüşdürür və aşağıdakı formata malikdir:

SEEK (<fayl dəyişəni>, <N-ci element>); burada <N-ci element>-fayl elementini bildirən **LONGINT** tipli ifadədir. Faylın birinci elementi sıfır sıra nömrəsinə malikdir. Bu proseduru mətni fayllara tətbiq etmək olmaz.

FILESIZE funksiyası fayldakı elementlərin sayını göstərən **LONGINT** tipli qiymət alır və aşağıdakı formata malikdir:

FILESIZE (<fayl dəyişəni>);

Bu funksiyanı mətni fayllara tətbiq etmək olmaz.

FILEPOS funksiyası növbəti giriş-çıxış əməliyyatı ilə emal ediləcək fayl elementinin sıra nömrəsini ifadə edən **LONGINT** tipli qiymət qaytarır. Formatı aşağıdakı formadadır:

FILEPOS (<fayl dəyişəni>);

Bu funksiyanı mətni fayllara tətbiq etmək olmaz.

Misal 1. k1. dat faylına 20 həqiqi ədəd daxil etməli:

```
program H1;
var f:file of real; i:byte; x:real;
begin assign(f,'k1.dat'); rewrite(f);
for i:=1 to 20 do begin readln(x);
write(f,x)
end; close(f)
end.
```

Misal 2. k2.dat faylındakı ədədlərin cəmini tapmalı.

```
program H2;
var f1:file of real; s,x:real;
begin assign(f1,'k2.dat'); reset(f1);
s:=0;
 while not eof(f1) do
begin read(f1,x); s:=s+x end;
close(f1)
end.
```

Qeyri-tip faylları **FILE** tipli fayl dəyişənləri kimi elan olunurlar və digər fayllardan elementlərinin tipinin göstərilməsi ilə fərqlənirlər. Tipin göstərilməməsi bu faylları ixtiyari digər fayllara uyğunlaşdırılmağa imkan verir və disklə yaddaş arasında verilənlər mübadiləsinin sürətini artırmağa imkan verir. Qeyri-tip faylları **RESET** və ya **REWRITE** prosedurları ilə açarkən, bu faylların baytlarla uzunluğunu göstərmək olar. *Məsələn*,

```
var f:file;
begin assign(f,'k3.dat'); reset(f,512);
end.
```

Qeyri-tip faylın uzunluğu, **RESET** və **REWRITE** prosedurlarında ikinci parametr kimi verilir və bu parametr **WORD** tipli ifadə də ola bilər. Əgər uzunluq göstərilmirsə, o, 128 bayta bərabər qəbul edilir. Burada maksimal uzunluq 65535 bayt (Word tam tipinin tutumu) ola bilər. Qeyri-tip fayllarla iş zamanı, tip fayllara tətbiq olunan bütün prosedur və funksiyalardan istifadə olunur. Lakin **READ** və **WRITE** prosedurları daha sürətli

BLOCKREAD və **BLOCKWRITE** prosedurları ilə əvəz olunur:

BLOCKREAD (<fayl dəyişəni>, <bufer>, < [, <NN>]);

BLOCKWRITE (<fayl dəyişəni>, <bufer>, < [, <NN>]);

burada <bufer> - disklərdə verilənlərin mübadiləsində istifadə olunan dəyişən adı, <NN> verilməsi məcburi olmayan parametrdir və prosedurdan çıxış zamanı faktiki emal edilmiş yazılışların sayını göstərir.

5.14. Göstəricilər və dinamik yaddaş

Dinamik yaddaş – verilənlər seqmenti (64 kbayt), stek (adətən 16 kbayt) və bilavasitə proqram gövdəsini çıxmaqla proqramın işi üçün kompüterin ayırdığı operativ yaddaşdır. Bu yaddaşın ölçüləri müxtəlif ola bilir. Susmaqla bu ölçü kompüterin bütün mümkün yaddaşı ilə dəstəklənir və adətən ən azı 200-300 *kbayt* olur. Dinamik yaddaş faktiki olaraq böyük ölçülü verilənlər massivlərinin emalı üçün yeganə vasitədir. Bir çox praktiki məsələləri dinamik yaddaş olmadan həll etmək çətinlik törədir və ya heç mümkün olmur. Eləcə də kompüterin qrafik və səs imkanları ilə iş zamanı verilənlərin müvəqqəti yadda saxlanılması üçün də dinamik yaddaşdan geniş istifadə olunur. Kompüterin operativ yaddaşı, hər birinin ünvanı olan bir bayt həcmli oyuqlar ardıcıllığından ibarətdir. Turbo Pascal dilində dinamik yaddaşın idarə edilməsi üçün göstəricilərdən istifadə edilir. Göstərici-qiymət yaddaş oyuğunun ünvanı olan dəyişəndir. Kompüterdə ünvanlar seqment və yerdəyişmə adlanan iki onaltı mərtəbəli sözlərin kulliyatı kimi verilir. Seqment-uzunluğu 65536 bayt (64 kbayt) olan və 16-nın vuruğu (yəni, 0,16, 32, 48 və s.) olan fiziki ünvanla başlayan yaddaş hissəsidir. Yerdəyişmə isə lazımlı ünvana müraciət üçün seqmentin əvvəlindən neçə bayt buraxmaq lazım olduğunu göstərir. Kompüterin ünvan fəzası 1 Mbaytdır. Bir Mbayt hüdüdlarında ünvanlaşmaq üçün 20 dənə ikilik mərtəbə lazımdır ki, bu da 16-lıq sözdən (seqment və yerdəyişmə) aşağıdakı qayda ilə alınır: seqmentin tərkibi 4 mərtəbə sola yerdəyişmə edir, boşalmış sağ mərtəbələr sıfırlarla doldurulur və nəticə yerdəyişmənin tərkibi ilə toplanır. Burada 16 baytlıq yaddaş fraqmenti paraqraf adlanır,

buna görə də seqment yaddaşı paraqraf dəqiqliyi ilə ünvanlayır, yerdəyişmə isə bayt dəqiqliyi ilə ünvanlaşma aparır. Hər bir seqmentə kəsilməz və ayrıca ünvanlanan yaddaş oblastı uyğun gəlir. Seqmentlər yaddaşda bir-birinin ardınca intervalsız və ya intervala gələ bilər və ya nəhayət, üst-üstə düşə bilər. Beləliklə, öz daxili strukturuna görə ixtiyari göstərici, seqment və yerdəyişmə kimi qəbul edilən iki sözün (**WORD** tipli verilənlər) külliyatıdır. Göstəricilərin köməyi ilə dinamik yaddaşda Turbo Pascal-ın ixtiyari tip verilənlərini yerləşdirmək olar. Adətən, Turbo Pascal-da göstərici müəyyən tip verilənlərlə əlaqələndirilir. Bu cür göstəricilər tip-göstəricilər adlanır. Bu göstəricilərin elanı üçün uyğun tip qarşısında ^ işarəsi qoyulur. *Məsələn*,

```
var p1:^integer;p2:^real;
type p3=^pr;
pr=record
n:string;
m:string;
next:pr
end;
```

Qeyd edək ki, göstəricilər, proqramda hələ elan edilməmiş verilənlər tipinə müraciət edə bilir.

Turbo Pascal-da göstəricini hansısa konkret verilənlər tipi ilə əlaqələndirmədən də elan etmək olar. Bunun üçün **POINTER** standart tipindən istifadə edilir. *Məsələn*,

```
var p:pointer;
```

Bu növ göstəricilər, qeyri-tip göstəricilər adlanır. Qeyri-tip göstəricilər hər hansı konkret tiplə əlaqəli olmadığından, onların köməyi ilə proqramın işi boyunca strukturu və tipi dəyişən verilənləri dinamik yerləşdirmək əlverişli olur. Göstəricilərin qiyməti dəyişənlərin yaddaşdakı ünvanları olduğundan fərz etmək olardı ki, bir göstəricinin qiymətini digərinə vermək olar. Lakin bu belə deyil, Turbo Pascal-da eyni verilənlər tipi ilə əlaqəli olan göstəricilər arasında qiymətlər vermək olar. Məsələn, əgər

```
var p1,p2:^integer;p3:^real;
    p4:pointer;
```

onda **p1:=p2**; mənimsətməsi mümkündür, lakin **p1:=p3**; isə mümkün deyil, çünki **p1** və **p3** ayrı-ayrı verilənlər tipinə aiddir. Lakin bu qadağa qeyri-tip göstəricilərə şamil olunmur, buna görə də yazmaq olar:

```
p4:=p3;
p1:=p4;
```

Turbo Pascal-da dinamik yaddaş qalaq adlanan baytlar massivi kimi nəzərdən keçirilir. Fiziki olaraq, qalaq proqram gövdəsinin yerləşdiyi oblastdan sonra gələn böyük ünvanlarda yerləşir. Qalağın əvvəli **HEAPORG** standart dəyişənində, sonu isə **HEAPEND** dəyişənində yerləşir. Dinamik yaddaşın hələ tutulmamış hissəsinin cari sərhəddini **HEAPPTR** göstəiricisi bildirir. Dinamik yerləşdirilən ixtiyari dəyişən üçün yaddaş hissəsi **NEW** proseduru ilə ayrılır. Bu prosedura müraciət parametri tip göstəricisidir. Müraciət nəticəsində göstərici, verilənlərin yerləşdirilməsini başlamaq mümkün olan dinamik ünvana uyğun qiymət alır. Göstərici müəyyən qiymət aldıqdan sonra, yəni yaddaşın konkret fiziki baytını göstərdikdə, bu ünvan üzrə uyğun tip ixtiyari qiymət yerləşdirmək olar. Bunun üçün göstəricidən sonra heç bir probel qoymadan ^ işarəsi verilir. *Məsələn*,

```
i^:=2;  r^:=2*pi;
```

və s.

Beləliklə, göstəricinin göstərdiyi qiymət, yəni qalaqda yerləşdirilmiş verilənlər, göstəricidən sonra gələn ^ işarəsi ilə işarələnir. Əgər göstəricidən sonra ^ işarəsi yoxdursa, onda verilənlərin yerləşdirildiyi ünvan nəzərdə tutulur. Dinamik yerləşdirilmiş verilənlərdən proqramın ixtiyari yerində istifadə etmək olar, məsələn, `r^:=sqr(r^)+i^-17;` .Eyni zamanda `r^=sqr(r);` və ya `r:=sqr(r^)+i^;` kimi yazılışlar düzgün deyil.

Dinamik yaddaşı qalaqdan götürməklə yanaşı, onu qalağa qaytarmaq da mümkündür. Bunun üçün **DISPOSE** prosedurundan istifadə edilir. *Məsələn*,

```
var i,j:^integer;r:^real;
begin new(i);new(r);
...
dispose(i);dispose(r);
...
end.
```

Burada **DISPOSE** prosedurları, **NEW** prosedurlarının **i** və **r** göstəriciləri üçün ayırdıqları 8 baytı, qalağa qaytarır. Qeyd edək ki, **DISPOSE(X)** proseduru **X** göstəricisinin qiymətini dəyişmir, yalnız bu göstərici ilə əvvəldən əlaqəli olan yaddaşı qalağa qaytarır. Boşalan göstəricini **NIL** sözü ilə qeyd etmək olar. Hər hansı göstəricinin qeyd olub-olmamasını aşağıdakı kimi yoxlamaq olar:

```
const p:^real= NIL;
begin if p=NIL then new(p);
...
dispose(p); p:=NIL;
end.
```

Göstəricilər üzərində digər müqayisə əməliyyatları aparmaq qadağan olub.

Qalaqda olan bütün əməliyyatlar qalaq adminstratoru adlanan xüsusi alt proqramın vasitəsilə aparılır. **NEW** proseduruna müraciət zamanı həmin alt proqram tələb olunan dəyişənin yerləşdirilməsi üçün ən kiçik azad fraqment axtarıb tapır. Tapılan fraqmentin başlanğıcının ünvanı göstəricidə qaytarılır, fraqmentin özü və ya onun lazımi uzunluqlu hissəsi qalağın tutulmuş hissəsi kimi qeyd olunur. Digər imkan isə qalağın tam fraqmentinin azad olunmasından ibarətdir. Bu məqsədlə dinamik yaddaşın ayrılmasından əvvəl **HEAPPTR** göstəricisinin cari qiyməti **MARK** prosedurunun köməyi ilə dəyişən-göstəricidə yadda saxlanılır. İndi ixtiyari anda **MARK** prosedurunun yadda saxladığı ünvandan başlayaraq, dinamik yaddaşın sonuna qədər olan qalaq fraqmentini azad etmək olur. Bunun üçün **RELEASE** prosedurundan istifadə olunur. *Məsələn*,

```
var p,p1,p2,p3,p4,p5:^integer;
begin new(p1);new(p2); mark(p);
            new(p3); new(p4); new(p5);
...
release(p);
end.
```

Qeyd edək ki, **NEW** prosedurunun parametri yalnız tip göstərici ola bilər. Qeyri-tip göstəricilərlə iş üçün aşağıdakı prosedurlardan istifadə olunur:

GETMEM(p,SIZE) – yaddaşın ayrılması;

FREEMEM(p,SIZE) – yaddaşın azad edilməsi;

Burada p – qeyri-tip göstərici, size – isə tələb olunan və ya azad edilən qalaq hissəsinin baytlarla ölçüsüdür.

Dinamik yadaşla iş üçün istifadə olunan prosedur və funksiyaları qeyd edək:

1) **ADDR** funksiyası, arqumentin ünvanından ibarət **POINTER** tipli nəticə verir. Ümumi forması: **ADDR(x)** kimidir, burada **X**-proqramın ixtiyari obyektidir (ixtiyari dəyişən, prosedur, funksiya adıdır). Qaytarılan ünvan ixtiyari göstərici ilə uyğunlaşır. Qeyd edək ki, analoji nəticəni @ əməliyyatı da verir.

2) **CSEG** funksiyası mikroprosessorun **CS** registrində saxlanılan qiyməti qaytarır. Ümumi forması: **CSEG**. Nəticə **WORD** tipli sözdə qaytarılır.

3) **DISPOSE** proseduru, əvvəlcədən tip göstərici üçün ayrılmış dinamik yaddaş fraqmentini qalağa qaytarır. Ümumi forması: **DISPOSE(TP)**, burada **TP**-tip göstəricidir.

4) **DSEG** funksiyası mikroprsessorun **DS** registrində saxlanılan qiyməti qaytarır. Ümumi forması: **DSEG**. Nəticə **WORD** tipli sözdə qaytarılır.

5) **FREEMEM** proseduru, əvvəlcədən qeyri-tip göstərici üçün ayrılmış dinamik yaddaş fraqmentini qalağa qaytarır. Ümumi forması: **FREEMEM (P,SIZE)**, burada **P** – qeyri-tip göstəricidir, **SIZE** isə həmin fraqmentin baytlarla uzunluğudur.

6) **GETMEM** proseduru qeyri-tip göstərici üçün tələb olunan ölçülü dinamik yaddaş fraqmenti ayrır. Ümumi forması: **GETMEM(P,SIZE)**, burada **P** – qeyri-tip göstərici, **SIZE**-isə fraqmentin baytlarla uzunluğudur.

7) **MARK** proseduru **HEAPPTR** qalaq göstəricisinin cari qiymətini yadda saxlayır. Ümumi forması: **MARK(P)**, burada **P** – ixtiyari tip göstəricidir, bu göstəricidə **HEAPPTR**-ın cari qiyməti qaytarılacaq.

8) **MAXAVAIL** funksiyası qalağın ən böyük kəsilməz hissəsinin baytlarla ölçüsünü təyin edir. Ümumi forması: **MAXAVAIL**.

Nəticə **LONGINT** tiplidir.

9) **MEMAVAIL** funksiyası qalağın ümumi boş sahəsinin baytlarla ölçüsünü təyin edir. Ümumi forması: **MEMAVAIL**. Nəticə **LONGINT** tiplidir.

10) **NEW** proseduru dəyişənin yerləşdirilməsi üçün qalaq fraqmenti ayırır. Ümumi forması: **NEW(TP)**, burada **TP**-tip göstəricidir.

11) **OFS** funksiyası göstərilən obyektin ünvanının yerdəyişməsini bildirən **WORD** tipli qiymət qaytarır. Ümumi forması: **OFS(x)**, burada X-ixtiyari tipli ifadə və ya prosedur adıdır.

12) **PTR** funksiyası verilmiş **SEG** seqmenti və **OFS** yerdəyişməsinə görə **POINTER** tipli qiymət qaytarır. Ümumi forması: **PTR(SEG,OFS)**, burada **SEG**-seqmenti bildirən **WORD** tipli ifadə, **OFS**-isə yerdəyişməni bildirən **WORD** tipli ifadədir. Funksiyanın qaytardığı qiymət, ixtiyari tip göstərici ilə uyğunlaşır.

13) **RELEASE** proseduru qalaq hissəsini azad edir. Ümumi forması: RELEASE (PTR), burada **PTR**-əvvəlcədən **MARK** proseduru ilə yadda saxlanılan qalaq göstəricisinin qiymətini saxlayan ixtiyari tipli göstəricidir.

14) **SEG** funksiyası göstərilən obyektin ünvan seqmentini bildirən **WORD** tipli qiymət qaytarır. Ümumi forması: **SEG(x)**, burada x-ixtiyari tip ifadə və ya prosedur adıdır.

15) **SIZEOF** funksiyası göstərilən obyektin daxili ifadəsinin baytlarla uzunluğunu təyin edir. Ümumi forması: **SIZEOF(x)**, **x** – burada dəyişən, funksiya və ya tip adıdır.

İstifadəçi proqramı ilə qalaq arasında əlaqə yaradan işçi alt proqram – qalaq adminstratoru adlanır. Qalaq adminstratoru new, getmem, dispose, freemem və s. prosedurların işini təmin edir və **HEAPPTR**, **FREELIST** göstəricilərinin qiymətini dəyişdirir. Burada **HEAPPTR** göstəricisi qalağın azad hissəsinin aşağı sərhəddinin ünvanını verir, **FREELIST** göstəricisi isə birinci azad blokun ünvanını verir.

5.15 Tip sabitlər

Turbo Pascal dilində tip sabitlərdən istifadə edilə bilir. Onlar sabitlərin elanı bölməsində aşağıdakı şəkildə verilirlər:

<identifikator>: <tip>= <qiymət>;

burada <identifikator> - sabitin identifikatoru, <tip>-sabitin tipi,<qiymət> - isə sabitin qiymətidir. Proqramın yerinə yetirildiyi müddət ərzində tip sabitlərə digər qiymətlər mənimsətmək olur, buna görə də onlar faktiki olaraq başlanğıc qiymətləri olan dəyişənlərdir. Tip sabitlər, fayldan başqa ixtiyari tipə aid ola bilərlər.

Sadə tiplərin və **STRING** tipinin sabitlərinin elanı çətinlik törətmir, çünki onların qiymətləri kimi qeyri-tip sabitlərdən və ya onların identifikatorlarından istifadə olunur. *Məsələn,*

```
type colors =(white,red,black);
const
c1:colors=red;name:string='Aliyev A.';
year:word=1966;x:real=4.1;min:integer =0;
max:integer=10;days:1..31=21;
```

Tip sabit – massivlər üçün başlanğıc qiymət kimi yumru mötərizələr daxilində, bir-birindən vergüllə ayrılan sabitlər siyahısından istifadə edilir. Məsələn,

```
type colors=(white,red,black);
const c1:array[colors] of string[5]
=('white','red','black');
c2:array[1..5]of byte=(1,2,3,4,5);
```

CHAR simvol tipli sabit-massivlər kimi uyğun uzunluqlu simvol sətrini vermək olar. Məsələn, aşağıdakı elanlar eynigüclüdür:

```
const p1:array[0..5]of char=('0','1','2',
'3','4','5');
p2:array[0..5]of char='012345';
```

Çoxölçülü sabit-massivlərin elanı zamanı, hər bir ölçüyə uyğun sabitlər əlavə yumru mötərizələrə alınır və bir-birindən vergüllə ayrılır. *Məsələn,*

```
var i,j,k:byte;
```

```
   const mat:array[1..3,1..3]of
byte=((0,1,2),(3,4,5), (6,7,8));
     cube:array[0..1,0..1,0..2]of
integer=(((0,1,2), (3,4,5)),((6,7,8),
9,10,11)));
     begin for i:=1 to 3 do
               for j:=1 to 3 do
write(mat[i,j]:3);
     writeln;
               for i:=0 to 1 do
               for j:=0 to 1 do
               for k:=0 to 2 do
write(cube[i,j,k]:3);
     writeln
     end.
```

Sabit-yazılışlar aşağıdakı kimi təyin edilir:

<identifikator>: <tip>= (<sahə qiymətlərinin siyahısı>);

burada <identifikator>-sabit identifikatoru, <tip>-yazılışın tipi, <sahə qiymətlərinin siyahısı>-sahə qiymətlərinin siyahısıdır. Qeyd edək ki, sahə qiymətlərinin siyahısı, <sahə adı:sabit> formalı ardıcıllıqlar siyahısıdır. *Məsələn,*

```
     type point=record x,y:real end;
          vec=array[0..1] of point;
     month=(Jan,Feb,Mar,Apr,May,Jun,Jly,Aug,
Sep, Oct,Nov,Dec);
     date=record
          d:1..31;
          m:month;
           y:1966..1999
              end;
     const p1:point=(x:0;y:-1);
          p2:vec=((x:1.1;y:1.2),(x:4.5;y:7.5));
          p3:date=(d:21;m:Apr;y:1966);
```

Sabit-çoxluqların qiymətləri düzgün çoxluq konstruktoru kimi verilir, məsələn,

```
     type days=set of 1..31;
```

```
        d1=set of '0'..'9';
        d2=set of 1..24;
    const workdays:days=[1..5, 8..12, 15..19,
22..26,29,30];
          k1:d1=['0','2','4','6','8'];
          k2:d2=[];
```

Tip göstəricinin yeganə qiyməti NIL ola bilər. *Məsələn*,

```
Const p:^real=NIL;
```

5.16. Modullar

Modullar aşağıdakı struktura malikdir:

unit <ad>;
interface
<interfeys hissəsi>
implementation
<yerinə yetirilən hissə>
begin
<operatorlar bölməsi>
end.

Burada **unit**-modulun başlığını bildirən işçi söz, <ad>-isə modulun adını bildirən identifikatordur. **interface**-modulun interfeys hissəsini başlayan işçi söz, **implementation**-isə modulun yerinə yetirilən hissəsini başlayan işçi sözdür. **begin**-modulun operatorlar bölməsini başlayan işçi sözdür, burada **begin**<operatorlar bölməsi> verilməyə də bilər. Nəhayət **end** modulun sonunu bildirən işçi sözdür. Beləliklə, modul başlıqdan və ixtiyari boş ola biləcək üç tərkib hissədən ibarətdir.

Modulun başlığı **Unit** işçi sözündən və onun ardınca gələn modul adından ibarətdir. Bu ad modul mətninin yerləşdirildiyi disk faylının adı ilə üst-üstə düşməlidir. Məsələn, əgər **Unit** Global; başlığı verilibsə, onda uyğun modulun mətni Global.pas

faylında yerləşməlidir. Modul adı onun digər modullarla və əsas proqramla əlaqə yaradılması üçün nəzərdə tutulub. Bu əlaqə USES <modullar siyahısı>; bölməsi ilə yaradılır. Burada **Uses**-işçi söz, <modullar siyahısı> isə əlaqə yaradılan modullar siyahısıdır. Siyahıda bir-birindən vergüllə ayrılan modul adları gətirilir. Məsələn, **Uses CRT, Graph, Global;**. Əgər proqramda **Uses** elan bölməsindən isitfadə edilirsə, bu bölmə əsas proqramın təsvirlər bölməsində birinci gəlməlidir. Bir modulda digər moduldan istifadə edilə bilər. Belə ki, **Uses** bölməsi modullarda **Interface** və ya **Implementation** sözlərindən sonra və ya nəhayət eyni zamanda hər iki sözdən sonra dərhal verilə bilər (yəni eyni zamanda iki **Uses** bölməsi verilə bilər).

Modulun interfeys hissəsi **Interface** işçi sözü ilə açılır. Bu hissədə modulun əsas proqramda və (və ya) digər modullarda istifadəsi mümkün olan bütün qlobal obyektlərinin (tiplərin, sabitlərin, dəyişənlərin və alt proqramların) elanı verilir. Qlobal alt proqramların interfeys hissəsində elanı zamanı, onların yalnız başlıqları verilir. *Məsələn,*

```
Unit Cmplx;
Interface
type complex=record
                re,im:real
                  end;
procedure Addc(x,y:complex;var z:complex);
procedure Mulc(x,y:complex;var z:complex);
```

İndi əgər əsas proqrama **Uses Cmplx;** bölməsini daxil etsək, proqramda Cmplx modulunun Complex tipindən və **Addc, Mulc** prosedurlarından istifadə etmək olar.

Yerinə yetirilən hissə **IMPLEMENTATION** işçi sözü ilə başlayır və interfeys hissədə elan edilmiş alt proqramların təsvirindən ibarətdir. Burada eyni zamanda modul üçün lokal olan obyektlər – köməkçi tiplər, sabitlər, dəyişənlər və operatorlar bölməsində istifadə edildiyi halda nişanlar elan edilir. Modulun interfeys hissəsində elan olunmuş alt proqramın təsvirindən əvvəl, yerinə yetirilən hissədə onun başlığı verilir. Bu başlıqda formal dəyişənlərin

siyahısını və funksiya üçün nəticənin tipini verməmək də olar, çünki onlar artıq interfeys hissəsində təsvir edilmişdi. Lakin əgər alt proqramın başlığı tam şəkildə, yəni formal parametrlərin siyahısı və nəticənin elanı ilə gətirilirsə, onda o, interfeys hissədə elan edilmiş başlıqla üst-üstə düşməlidir. Məsələn,

```
Unit Cmplx;
Interface
Type complex=record
              re,im:real;
                 end;
procedure Addc(x,y:complex;var z:complex);
Implementation
procedure Addc;
begin z.re:=x.re+y.re;z.im:=x.im+y.im
end;
end.
```

Operatorlar bölməsi modulu sona çatdıran bölmədir. Bu bölmə onu başlayan **BEGIN** sözü ilə birlikdə modulda verilməyə də bilər və ya boş ola bilər. Bölmə boş olduqda **BEGIN** sözündən sonra modulun sonu əlaməti (**END** sözü və nöqtə işarəsi) gəlir. Bu bölmədə proqramın müəyyən fraqmenti olan operatorlar yerləşdirilir. Bu operatorlar əsas proqrama idarəetmə verilənə qədər yerinə yetirilirlər. Məsələn, burada lazımi fayllar açılır, digər kompüterlərlə əlaqə yaradılır və s.

Misal. Kompleks ədədlər üçün hesab əməllərini realizə edən modul quraq.

```
Unit cmplx;
Interface
type complex =record
                re,im:real;
                          end;
procedure        Addc(x,y:complex;        var
z:complex);
procedure Subc(x,y:complex;var z:complex);
procedure Mulc(x,y:complex;var z:complex);
procedure Divc(x,y:complex;var z:complex);
const c;complex=(re: 0.1; im:-1);
```

```
Implementation
procedure Addc;
begin z.re:=x.re+y.re;z.im:=x.im+y.im end;
procedure Subc;
begin z.re:=x.re-y.re;z.im:=x.im-y.im end;
procedure Mulc;
begin z.re:=x.re*y.re-x.im*y.im;
z.im:=x.re*y.im+x.im*y.re end;
procedure Divc;
var zz:real;
begin zz:=sqr(y.re)+sqr(y.im);
z.re:=(x.re*y.re+x.im*y.im)/zz;
z.im:=(x.re*y.im-x.im*y.re)/zz end;
end.
```

Bu modulun mətnini cmplx.pas faylında yerləşdirib, proqramda bu moduldan istifadə etmək olar. *Məsələn,*

```
Uses cmplx;
var a,b,c:complex;
begin a.re:=1;a.im:=1;b.re:=1;b.im:=2;
            addc(a,b,c);
writeln(c.re:5:1,c.im:5:1,'i');
            subc(a,b,c);
writeln(c.re:5:1,c.im:5:1,'i');
            mulc(a,b,c);
writeln(c.re:5:1,c.im:5:1,'i');
            divc(a,b,c);
writeln(c.re:5:1,c.im:5:1,'i');
end.
```

Turbo Pascal dilində tərkibində çoxlu sayda müxtəlif tiplər, sabitlər, prosedur və funksiyalar olan səkkiz standart modul var. Bu modullar: system, dos, crt, printer, graph, overlay, turbo3 və graph3 modullarıdır. Bunlardan graph, turbo3 və graph3 modulları ayrıca TPU tipli fayllarda yerləşdirilib, qalan modullar isə turbo.tpl faylına daxildir. Modullardan yalnız system modulu ixtiyari proqrama avtomatik olaraq qoşulur, yerdə qalan modullardan isə onların adları uses bölməsində verildikdən sonra istifadə etmək olur.

System moduluna standart Pascal dilinin bütün prosedur və funksiyaları, eləcə də digər standart modullara daxil olmayan prosedur və funksiyalar (məsələn, **inc,dec,getdir** və s.) aiddir.

Printer modulu mətnlərin printerə çıxarılmasını təmin edir. Bu modulda text tipli, lst fayl dəyişəni müəyyən edilir və prn məntiqi qurğusu ilə əlaqələndirilir. Məsələn, bu modul qoşulduqdan sonra aşağıdakı proqram yerinə yetirilə bilər:

```
uses printer;
begin writeln(lst,'Turbo Pascal')end.
```

Crt modulu ekranın mətn rejimini idarə edən prosedur və funksiyalardan ibarətdir. Bu modula daxil olan alt proqramların köməyi ilə kursoru ekranın ixtiyari mövqeyinə çıxarmaq, çıxarılan simvolların rəngini dəyişdirmək, pəncərələr yaratmaq olur.

Graph moduluna ekranın qrafik rejiminin idarə edilməsi üçün çoxsaylı tip, sabit, prosedur və funksiyalar daxildir. **Graph** moduluna daxil olan alt proqramlar vasitəsilə müxtəlif qrafik təsvirlər qurmaq və ekrana standart və ya istifadəçinin yaratdığı şriftlərlə yazıları çıxarmaq olur.

Dos modulunda **MS DOS** əməliyyat sisteminin imkanlarından istifadə üçün şərait yaradan prosedur və funksiyalar cəmlənib.

Overlay modulu mürəkkəb proqramların qurulmasında istifadə edilir.

Turbo3 və **graph3** modulları isə əvvəlki Turbo Pascal **3.0** versiyası ilə uyğunlaşdırılma üçün tətbiq olunur.

5.17. Obyektlər

Obyekt yönümlü proqramlaşdırma obyektlərə yeni xassələr verən üç əsas prinsipə əsaslanır. Bu prinsiplər inkapsulasiya, varislik və polimorfizmdir.

İnkapsulasiya verilənlər və onların emal alqoritmlərinin bir vahid tam şəkildə birləşdirilməsidir. Obyekt yönümlü proqramlaşdırılma (OYP) daxilində verilənlər obyektin sahələri, alqoritm isə obyekt üsulları adlandırılır. İnkapsulyasiya obyekti xarici mü-

hitdən maksimal dərəcədə təcrid etməyə imkan verir. Bu isə yaradılan proqramların etibarlılığını əhəmiyyətli dərəcədə artırır, çünki obyektdə lokallaşdırılmış alqoritmlər proqramla verilənlərin nisbətən kiçik həcmləri ilə mübadilə edirlər, həm də bu verilənlərin tip və sayı adətən ciddi şəkildə nəzarət altında saxlanılır.

Varislik, obyektlərin öz varislərini yaratmaq xassəsidir. Varis-obyekt avtomatik olaraq valideyn-obyektdən bütün sahə və üsullara varis olur. Bundan əlavə obyektləri yeni sahələrlə tamamlayıb, valideyn üsullarını yeniləri ilə əvəz edə (üstələyə) və ya tamamlaya bilir. Varislik prinsipi obyektlərin xassələrinin modifikasiyası problemini həll edir. Obyektlərlə iş zamanı istifadəçi adətən konkret məsələnin həlli üçün öz xassələrinə görə ən uyğun obyekt seçir, sonra isə ondan bu obyektin bacarmadığı işləri yerinə yetirən bir və ya bir neçə varis yaradır.

Polimorfizm – qohum obyektlərin (yəni bir ümumi valideyni olan) mahiyyəti üzrə oxşar problemləri müxtəlif üsullarla həll etmək qabiliyyətidir. Obyekt yönümlü proqramlaşdırmada obyektin xassələri, bu obyektə daxil olan üsularla müəyyən olunur. Obyektin varislərində istifadəçi bu və ya digər üsulun alqoritmlərini dəyişdirməklə, həmin varislərə, onların valideynlərində olmayan xassələr verə bilər. Üsulu dəyişdirmək üçün varisdə eyni adlı üsulu elan edib, onda lazım olan işləri aparmaq lazımdır. Nəticədə valideyn-obyekt və varis-obyektdə müxtəlif alqoritmik əsası olan və deməli, obyektlərə müxtəlif xassələr verən eyni adlı üsullar olacaqdır. Bu obyektlərin poliformizmi adlanır.

Turbo Pascal dilində obyektlərin yaradılması üçün objekt, constructor, destructor işçi sözlərindən və private, public, virtual standart əmrlərindən istiafdə edilir. Bunlardan objekt işçi sözü obyektlərin təsviri üçün istifadə edilir. Obyektin təsviri, tiplərin təsviri bölməsində verilməlidir. *Məsələn,*

```
type obyekt=object
    <obyektin sahələri>
    <obyektin üsulları>
end;
```

Əgər obyektin hər hansı bir valideyndən törənibsə, valideynin adı object işçi sözündən sonra dərhal yumru mötərizə daxilində verilir:

```
type sonobyekt=object(obyekt)
```

<obyektin sahələri>
<obyektin üsulları>
end;

Hər bir obyektin ixtiyari sayda varisi ola bilər, lakin yalnız bir valideyni olur. Obyektlərin qurulmasını, ekranda müxtəlif qrafik təsvirlər (nöqtə, çevrə, xətt, kvadrat) qurub, onları ekran boyunca hərəkət etdirilməsini təmin edən proqramın qurulması misalı üzərində araşdıraq. Bunun üçün əvvəlcə **TGobyekt** adlı valideyn-obyekt yaradaq. Həmin obyektdə yerdə qalan obyektlər üçün ümumi olacaq sahə və üsullar inkapsulyasiyadan keçəcəkdir:

```
type TGobyekt=object
Private
x,y:integer;color:word;
Public
Constructor Init(ax,ay:integer;ac:word);
procedure Draw(ac:word);Virtual;
procedure Show;
procedure Hide;
procedure MoveTo(dx,dy:integer);
end;
```

Sonradan nöqtə, xətt, çevrə və düzbucaqlının xassələrini realizə edən, **TGobyekt** obyektinin varislərini yaratmaq lazımdır. Bu qrafik obyektlərdən hər biri ekrandakı mövqeyi (**x** və **y** sahələri) və rəngi (**color** sahəsi) ilə xarakterizə olunur. Burada **TGobyekt** abstrakt obyekti konkret qrafik fiqurla əlaqəli deyil. Bu obyektin özündə bütün ümumi sahələri və üsulları birləşdirir, o, digər obyektlər üçün valideyn rolunu oynayır. **Private** əmri obyektin təsvirində gizlədilmiş sahə və üsulların açılmasını təmin edir. **Public** əmri isə private əmrini ləğv edir, buna görə də public əmrindən sonra gələn obyekt elementləri ixtiyari proqram vahidində istifadə edilə bilər. **Private** və **public** əmrləri bir obyekt daxilində ixtiyari qaydada növbələşə bilər.

Sahələrin təsviri adi dəyişənlərin təsvirindən fərqlənmir. Sahə kimi verilənlərin ixtiyari strukturlarından, o cümlədən digər obyektlərdən də istifadə etmək olar. Obyekt-yönümlü proqramlaşdırmada üsulların təsviri üçün Turbo Pascal dilinin standart

prosedur və funksiyalarından, eləcədə xüsusi prosedurlardan-konstruktor və destruktorlardan istifadə edilir. Konstruktorlar konkret obyektin yaradılması üçün istifadə edilir, belə ki, obyekt-verilənlər tipidir, yəni onun əsasında ixtiyari sayda obyekt tipli verilənlər yaratmaq olur. Konstruktorun başlığında procedure sözü əvəzinə **constructor** işçi sözündən istifadə olunur. Onun köməyi ilə virtual adlanan üsullar cədvəli yaradılır. Obyektdə virtual üsullar yoxdursa, burada konstruktor iştirak edə bilməz, əksinə üsullardan heç olmasa biri virtual təsvir edilibsə, obyektin tərkibinə ən azı bir konstruktor daxil olmalıdır və konstruktora müraciət, ixtiyari virtual üsula müraciətdən əvvəl gəlməlidir. Konstruktor obyekt sahələrini konkret qiymətlərlə təmin edir. Qeyd edək ki, eyni bir obyektin ayrı-ayrı nümunələri bir-birindən obyekt sahələrinin tərkibi ilə fərqlənir, lakin onların hamısı eyni bir obyekt üsullarından istifadə edirlər. Obyektin bütün xassələrini təsvir etmək üçün, obyekt üsullarının tərkibini açmaq, yəni uyğun prosedur və funksiyaları təsvir etmək lazımdır. Üsulların təsviri Turbo Pascal dilindəki adi qayda ilə təsvirlər bölməsində obyektin təsvirindən sonra verilir. *Məsələn*,

```
type TGobyekt=object
.....
end;
Constructor TGobyekt.Init;
begin x:=ax;y:=ay;color:=ac end;
procedure TGobyekt.Draw;
begin
end;
Procedure TGobyekt.Show;
begin Draw (color)end;
procedure TGobyekt.Hide;
begin Draw(GetBkColor)end;
procedure TGobyekt.MoveTo;
begin Hide;x:=x+dx;y:=y+dy;Show end;
```

Qeyd edək ki, üsulların təsviri zamanı üsulun adından əvvəl obyektin adı verilir, yəni üsulun qurma adından istifadə edilir.

Misal. Surət və məxrəcin ən böyük ortaq bölənini tapmaq, ixtisar etmək və natural tərtib üsulları olan adi kəsr obyektini təsvir edək.

```
type natur=1..32767;
frac=record p:integer; q:natur end;
drob=object a:frac;
procedure vvod;
procedure nod(var c:natur);
procedure sokr;
procedure stepen(n:natur; var c:frac);
procedure print;
end;
procedure drob.nod;
var m,n:natur;
begin m:=abs(a.p); n:=a.q;
while m<>n do
if m<>n then if m mod n<>0 then m:=m mod n
   else m:=n else if n mod m<>0 then n:=n
mod m
   else n:=m; c:=m end;
procedure drob.sokr;
var n:natur;
begin if a.p<>0 then begin drob.nod(n);
      a.p:=a.p div n; a.q:=a.q div n end
                  else a.q:=1 end;
procedure drob.stepen;
var i:natur;
begin c.p:=1; c.q:=1;
for i:=1 to n do begin c.p:=c.p*a.p;
                       c.q:=c.q*a.q
                  end; end;
procedure drob.vvod;
begin write ('Surat:');readln(a.p);
          write ('Maxrac:')readln(a.q);
end;
procedure drob.print;
begin writeln(a.p,'/',a.q)end;
var z:drob;f:frac;
begin z.vvod;z.print;z.sokr;z.print;
z.stepen(4,f);writeln(f.p,'/',f.q)
end.
```

5.18. Turbo Pascal dilinin qrafik imkanları

Turbo Pascal dilinin 4.0 versiyasından başlayaraq, onun tərkibinə GRAPH qrafik alt proqramlar kitabxanası daxil edilib. Burada qrafik ekranın idarə edilməsi üçün 50-dən çox prosedur və funksiyalar var. Kompüter işə salınarkən onun standart vəziyyəti mətn rejiminə uyğundur, buna görə də kompüterin qrafik imkanlarından istifadə edən ixtiyari proqram displey adapterin qrafik iş rejimini aktivləşdirməlidir. Proqram işini sona çatdırdıqda kompüter yenidən mətn rejiminə qayıdır. Qrafik prosedurların konkret adapterlə işi lazımi qrafik drayverin qoşulması ilə tənzimlənir. Drayver kompüterin bu və ya digər texniki vasitələrinin idarə edilməsini təmin edən xüsusi proqramdır. Qrafik drayver displey adapterini qrafik rejimdə idarə edir. Ən geniş yayılmış adapterlərin qrafik iş rejimlərinin xarakteristikasına baxaq:

1) **CGA** (**Color Graphics Adapter** – rəngli qrafik adapter) – 5 qrafik rejimə malikdir. Bunlardan 4 rejim ekranın zəif imkanlarına (320 piksel üfüqi və 200 piksel şaquli istiqamətdə, yəni 320×200) uyğundur və palitra ilə məhdudlaşdırılan rənglər qrupu ilə fərqlənir. Hər bir palitra işıq saçmayan qara rənglə birlikdə 4 rəngdən ibarədir. Bunlar: palitra 0 (açıq yaşıl, al qırmızı, sarı), palitra 1(açıq firuzəyi, açıq qırmızı, ağ), palitra 2 (yaşıl, qırmızı, qəhvəyi) və palitra 3 (firuzəyi, bənövşəyi, açıq boz). Beşinci rejim 640×200 yüksək imkanlara uyğundur, lakin palitrası cəmi iki rəngdən ibarətdir.

2) **EGA** (**Enhanced Graphics Adapter** – gücləndirilmiş qrafik adapter) adapteri CGA adapterinin bütün qrafik rejimlərini də dəstəkləyir. Bundan əlavə burada aşağıdakı rejimlər də mümkündür: zəif imkanlı rejim (640×200, 16 rəng, 4 səhifə) və yüksək imkanlı rejim (640×350, 16 rəng, 1 səhifə). Bəzi modifikasiyalarda monoxrom rejimdən də (640×350, 1 səhifə, 2 rəng) istifadə edilir.

3) **MCGA** (Multi-**Color Graphics Adapter** – çox rəngli qrafik adapter) adapteri CGA adapteri ilə uyğunlaşıb və bundan əlavə daha bir rejimə (640×480, 2rəng, 1 səhifə) malikdir.

4) **VGA** (**Video Graphics Array** – qrafik video massiv) adapteri **CGA** və **EGA** adapterlərinin rejimlərini dəstəkləyir və onları yüksək imkanlı rejimlə (640×480, 16 rəng, 1səhifə) tamamla-

yır.

İndi isə qrafik prosedur və funksiyalara baxaq:

1) **InitGraph** proseduru adapterin qrafik rejimini iş vəziyyətinə gətirir. Onun ümumi şəkli aşağıdakı kimidir:

```
procedure InitGraph (var Driver,
Mode:integer; path:string);
```

burada Driver-integer tipli dəyişən olub, qrafik drayverin tipini təyin edir, **Mode-integer** tipli dəyişən olub, qrafik adapterin iş rejimini təyin edir, **Path-string** tipli ifadə olub, drayver faylının adını və ona gedən yolu göstərir. Bu prosedurdan istifadədən əvvəl disk informasiya daşıyıcılarının birində lazımi qrafik drayver olduğu fayl verilməlidir. Prosedur bu drayveri operativ yaddaşa yükləyib, adapteri qrafik iş rejiminə keçirir. Drayverin tipi, qrafik adapterin tipinə uyğun gəlməlidir. Drayver tipini göstərmək üçün modulda aşağıdakı sabitlər təyin edilib:

```
const detect=0;CGA=1; MCGA=2; EGA=3;
EGA64=4; EGAMono=5; IBM8514=6; HercMono=7;
ATT400=8; VGA=9; PC3270=10;
```

Adapterlərin çoxu müxtəlif rejimlərdə işləyə bilir. Adapterə lazım olan iş rejimini vermək üçün Mode dəyişənindən istifadə edilir. Məsələn, **CGA.BGI** drayveri **C** diskindəki **TP\BGI** kataloqunda yerləşirsə və palitra 2 ilə 320×200 iş rejimini daxil edirsə, onda prosedura müraciət aşağıdakı kimi ola bilər:

```
Uses Graph;
var Driver,Mode:integer;
begin Driver:=CGA;Mode:=CGAC2;
InitGraph(Driver,Mode,'C:\TP\BGI');
```

Əgər kompüterin adapter tipi məlum deyilsə və ya əgər proqram ixtiyari adapterlə iş üçün nəzərdə tutulubsa, prosedura drayver tipini avtomatik təyin edilməsi tələbi olan müraciət yerinə yetirilir:

```
Driver:=Detect;
InitGraph (Driver, Mode, 'C:\TP\BGI');
```

2) **GraphResult** funksiyası Integer tipli qiymət qaytarır, bu qiymət qrafik prosedurlara axırıncı müraciətin nəticəsini bildirir. Belə ki, əgər səhv yoxdursa, funksiya sıfır qiymətini, əks halda isə mənfi qiymət qaytaracaqdır.

3) **Function GraphErrorMsg (Code:integer): string;** funksiyası string tipli qiymət qaytarır ki, bu qiymətdə səhv koduna uyğun mətni məlumat verilir. Burada Code-GraphResult funksiyasının qaytardığı səhv kodudur.

4) **Procedure CloseGraph;** proseduru adapterin qrafik reji-mindəki işini başa çatdırır və ekranın mətni iş rejimini bərpa edir.

5) **Procedure RestoreCRTMode;** proseduru mətn iş rejiminə qısa müddətli qayıdışı təmin edir.

6) **Function GetGraphMode:integer;** funksiyası Integer tipli qiymət qaytarır və bu qiymətdə qrafik adapterin təyin olunmuş iş rejiminin kodu verilir.

7) **Procedure SetGraphMode (Mode:Integer);** proseduru adapterin yeni qrafik rejimini təyin edir. Burada **Mode**-təyin edilən rejimin kodudur.

8) **Procedure DetectGraph (var Driver, Mode: Integer);** proseduru drayverin tipini və onun iş rejimini təyin edir. Burada **Driver**-drayverin tipi, Mode-isə iş rejimidir.

9) **Function GetDriverName:String;** funksiyası **string** tipli qiymət qaytarır ki, burada yüklənmiş qrafik drayverin adı olur.

10) **Function GetMaxMode:Integer;** funksiyası adapterin mümkün iş rejimlərinin sayını bildirən **Integer** tipli qiymət qaytarır.

11) **Function GetModeName (ModNumber:integer):string;** funksiyası ekranın imkanlarını və nömrəsinə görə adapterin iş rejiminin adını bildirən **String** tipli qiymət qaytarır. Burada **ModNumber** rejimin nömrəsidir.

12) **Procedure GetModeRange (Drv:integer;var Min,Max:integer);** proseduru verilmiş qrafik adapterin mümkün iş rejimləri diapazonunu təyin edir. Burada **Drv**-adapter tipi, **Min** və **Max**-uyğun olaraq rejim nömrəsinin mümkün aşağı və yuxarı qiymətlərini bildirən **Integer** tipli dəyişənlərdir.

Bir çox qrafik prosedur və funksiyalar ekranın cari mövqe göstəricisindən isitfadə edirlər. Bu göstərici mətni kursordan fərqli olaraq ekranda görsənmir. Bu göstəricinin koordinatları və

ümumiyyətlə qrafik ekrandakı ixtiyari koordinatlar, koordinatları (0,0) olan ekranın yuxarı sol küncünə nəzərən verilirlər.

13) **GetMaxX** və **GetMaxY** funksiyaları cari iş rejimində uyğun olaraq üfüqi və şaquli istiqamətlərdə ekranın maksimal koordinatlarını bildirən **Word** tipli qiymətlər verir.

14) **GetX** və **GetY** funksiyaları göstəricinin uyğun olaraq üfüqi və şaquli isitqamətlərdəki cari koordinatlarını bildirən **Integer** tipli qiymətlər verir. Burada koordinatlar pəncərənin, pəncərə verilməyibsə ekranın yuxarı sol küncünə nəzərən təyin edilirlər.

15) **Procedure SetViewPort (x1,y1,x2,y2: integer;ClipOn:boolean);** proseduru qrafik ekranda düzbucaqlı pəncərə təyin edir. Burada **x1,y1,x2,y2**-koordinatları pəncərənin yuxarı sol **(x1,y1)** küncünün və aşağı sağ **(x2,y2)** küncünün koordinatlarıdır, **ClipOn** isə təsvirin pəncərədə yerləşməyən elementlərinin "kəsilməsini" təyin edən **Boolean** tipli ifadədir. Pəncərnin koordinatları həmişə ekranın yuxarı sol küncünə nəzərən verilir. Burada əgər **ClipOn** parametri **True** qiyməti alırsa, pəncərəyə sığmayan təsvir hissəsi kəsilib atılır, əks halda isə nəzərə alınmır.

16) **Procedure GetViewSettings (var iewInfo: ViewPortType);** proseduru cari qrafik pəncərənin kəsilmə əlamətini və koordinatlarını qaytarır. Burada **ViewInfo**, **ViewPortType** tipli dəyişəndir. Bu tip **Graph** modulunda aşağıdakı kimi təyin edilib:

```
type ViewPortType=record
x1,y1,x2,y2:integer;
clip:boolean
end;
```

17) **Procedure MoveTo (x,y:integer);** proseduru göstərici üçün yeni cari mövqe təyin edir. Burada **x,y**-uyğun olaraq üfüqi və şaquli istiqamətlərdə göstəricinin yeni koordinatlarını müəyyən edir. Koordinatlar pəncərənin, pəncərə təyin edilməyibsə, ekranın yuxarı sol küncünə nəzərən təyin edilirlər.

18) **Procedure MoveRel (DX,DY:integer);** proseduru göstəricinin yeni vəziyyətini nisbi koordinatlarla təyin edir.

Burada **DX,DY** yeni koordinatların uyğun olaraq üfüqi və şaquli istiqamətlərdə artımlarıdır. Artımlar göstəricinin bu prosedura müraciətdən əvvəl mövqeyinə nəzərən verilir.

19) **Procedure ClearDevice;** proseduru qrafik ekranı təmizləyir. Prosedur yerinə yetirildikdən sonra göstərici ekranın yuxarı sol küncündə yerləşdirilir, ekran isə **SetBkColor** prosedurunun verdiyi fon rəngi ilə rənglənir.

20) **Procedure ClearViewPort;** proseduru qrafik pəncərəni təmizləyir, pəncərə müəyyən edilməyibsə bütün ekranı təmizləyir. Göstərici isə pəncərənin yuxarı sol küncünə yerləşdirilir.

21) **Procedure GetAspectRatio(var x,y:Word);** proseduru ekranın tərəflərinin nisbətini qiymətləndirməyə imkan verən iki ədəd qaytarır. Burada **x,y** – **Word** tipli dəyişənlərdir. Bu dəyişənlərin qaytardığı qiymətlər qrafik ekranın tərəflərinin nisbətini piksellərlə qiymətləndirməyə imkan verir. Bu tapılan əmsallar düzgün həndəsi fiqurların çevrə, kvadrat və s. qurulmasında istifadə edilə bilər.

22) **Procedure SetAspectRatio(x,y:Word);** proseduru qrafik ekranın tərəflərinin nisbətinin miqyas əmsalını təyin edir. Burada **x,y** tərəflərin təyin edilən nisbətləridir.

23) **Procedure SetActivePage(PageNum:Word);** proseduru göstərilən videoyaddaş səhifəsini aktiv edir. Burada **PageNum** səhifə nömrəsidir.

24) **Procedure SetVisualPage(PageNum:Word);** proseduru göstərilən nömrəli səhifəni əks etdirir. Burada **PageNum** səhifə nömrəsidir.

25) **Procedure PutPixel(x,y:integer;color: Word);** proseduru göstərilən koordinatlarda verilmiş rəngli nöqtə çıxarır. Burada **x,y** – nöqtənin koordinatları, **color** isə nöqtənin rəngidir.

26) **Function GetPixel(x,y:integer):Word;** funksiyası göstərilən koordinatlı pikselin rəngini bildirən **Word** tipli qiymət qaytarır. Burada **x,y**-pikselin koordinatlarıdır.

27) **Procedure Line(x1,y1,x2,y2:integer);** proseduru başlanğıc **(x1,y1)** və son **(x2,y2)** nöqtələrinin koordinatları göstərilmiş xətt çəkir. Burada **(x1,y1)**-xəttin başlanğıcı-

nın və **(x2,y2)** isə sonunun koordinatlarıdır.

28) **Procedure LineTo (x,y:integer);** proseduru göstəricinin cari mövqeyindən başlayaraq, prosedurda göstərilən yeni koordinatlarına qədər düz xətt çəkir. Burada **x,y**-göstəricinin mövqeyinin yeni koordinatları və deməli xəttin ikinci üç nöqtəsinin kordinatlarıdır.

29) **Procedure LineRel (DX,DY:integer);** proseduru göstəricinin cari mövqeyindən başlayaraq, onun prosedurda göstərilən koordinat artımlarının təyin etdiyi nöqtəyə qədər düz xətt çəkir. Burada **DX,DY**-göstəricinin yeni mövqeyinin koordinatlarını almaq üçün artımdır. **Line**, **LineTo**, **LineRel** prosedurları ilə xətt cari stil və rənglə çəkilir.

30) **Procedure Rectangle(x1,y1,x2,y2:integer);** proseduru yuxarı sol və aşağı sağ təpələrinin koordinatları ilə verilən düzbucaqlı çəkir. Burada **(x1,y1)**–yuxarı sol, **(x2,y2)** isə aşağı sağ təpələrinin koordinatlarıdır. Düzbucaqlı cari xətt stili və rəngi ilə çəkilir.

Misal. Aşağıdakı proqram ekranda bir-birinin daxilində yerləşdirilən 10 düzbucaqlı qurur.

```
uses graph,crt;
var d,r,e,x1,y1,x2,y2,dx,dy:integer;
begin d:=detect; initgraph(d,r,'');
e:=graphresult;
if e<>grok then writeln(grapherrormsg(e))
     else begin dx:=getmaxx div20;
 dy:=getmaxy div 20;
for d:=0 to 9 do
 rectangle(d*dx,d*dy,getmaxx-d*dx,
 getmaxy-d*dy);
if readkey=#0 then d:=ord(readkey);
closegraph
end
end.
```

31) **Procedure drawpoly (n:word; var points);** proseduru sınma nöqtələrinin koordinatları ilə verilən ixtiyari sınıq xətt çəkir. Burada **n** – hər iki kənar nöqtələr də daxil olmaqla sınıq xəttin təpə nöqtələrinin sayını göstərir, **Points** isə

sınma nöqtələrinin koordinatlarından ibarət **PointType** tipli dəyişəndir. Sınma nöqtələrinin koordinatları **Word** tipli qiymətlər cütü ilə verilir. Onlar üçün modulda təyin edilmiş aşağıdakı tipdən isitfadə edilə bilər:

```
type PointType=record
                x,y:Word
                end;
```

32) **Procedure Circle(x,y:integer; r:word);** proseduru çevrə çəkir. Burada **x,y** – çevrənin mərkəzinin koordinatları, **r** – isə onun piksellərlə radiusudur.

33) **Procedure Arc (x,y:integer; BegA,EndA, r:word);** proseduru çevrə qövsü çəkir. Burada **x,y** –mərkəzin koordinatları, **BegA,EndA** – qövsün uyğun olaraq başlanğıc və son qövs dərəcələridir, **r** – radiusdur. Qövslər saat əqrəbinin hərəkətinin əksi istiqamətində çəkilir və dərəcələrlə göstərilir.

34) **Procedure GetArcCoords (var Coords:ArcCoordsType);** proseduru qövsün mərkəzinin, başlanğıcının və sonunun koordinatlarını qaytarır. Burada **Coords, ArcCoordsType** tipli dəyişəndir. Bu dəyişəndə prosedur qövsün mərkəzinin, başlanğıcının və sonunun koordinatlarını verir. Bu tip **Graph** modulunda aşağıdakı kimi təyin edilir:

```
type ArcCoordsType=record
        x,y:integer;
        xs,ys:integer;
        xe,ye:integer;
end;
```

35) **ProcedureEllips(x,y:integer; BegA,EndA, rx,ry:word)** proseduru ellips qurur. Burada **x,y** – mərkəzin koordinatları, **BegA, EndA** – uyğun olaraq qövsün başlanğıc və son bucaqlarıdır, **rx, ry** – ellipsin piksellərlə üfüqi və şaquli oxlarıdır.

36) **Procedure SetColor(Color:word);** proseduru çıxarılan xətt və simvollar üçün cari rəngi təyin edir. Burada **Color** – cari rəng nömrəsidir.

37) **Function GetColor:word;** funksiyası cari rəngin kodunu verən **Word** tipli qiymət qaytarır.

38) **Function GetMax Color:word;** funksiyası **SetColor** proseduraşına müraciət üçün istifadə edilə bilən maksimal rəng kodunu təyin edir.

39) **Procedure SetBkColor(Color:word);** proseduru fonun rəngini təyin edir. Burada **Color** fonun rəngidir.

40) **Function GetBkColor:Word;** funksiyası fonun cari rəngini göstərən **Word** tipli qiymət qaytarır.

41) **Procedure SetPalette(n:word; Color: ShortInt);** proseduru palitradakı bir rəngi yeni rənglə əvəz edir. Burada **n** – palitradakı rəngin nömrəsidir, **Color** isə yeni təyin edilən rəngin nömrəsidir.

42) **Procedure FloodFill(x,y:integer;Border: word);** proseduru ixtiyari qapalı fiquru cari rənglə rəngləyir. Burada **x, y** – qapalı fiqur daxilindəki ixtiyari nöqtənin koordinatları, **Border** isə sərhəd xəttinin rənginin nömrəsidir.

43) **Procedure Bar(x1,y1,x2,y2:integer);** proseduru ekranın düzbucaqlı oblastını cari rənglə rəngləyir. Burada **(x1,y1)** – rənglənən oblastın yuxarı sol küncünün, **(x2,y2)** isə aşağı sağ küncünün koordinatlarıdır.

44) **Procedure Bar3D(x1,y1,x2,y2,D:integer; T:boolean);** proseduru paralelepipedin üçölçülü təsvirini verir və onun ön hissəsini rəngləyir. Burada **(x1,y1)** – ön hissənin yuxarı sol küncünün, **(x2,y2)** isə aşağı sağ küncünün koordinatlarıdır, **D** – üçölçülü təsvirin piksellərlə dərinliyini təyin edir, **T** isə fiqurun yuxarı tilinin təsvir üsulunu təyin edir. Belə ki, **T** parametri **True** qiyməti aldıqda bu til əks etdirilir, əks halda isə gizlədilir.

45) **Procedure FillPoly(N:Word;var Coords);** proseduru qapalı çoxbucaqlını xətlə əhatələyir və rəngləyir. Burada **N**-qapalı çoxbucaqlının təpələrinin sayı, **Coords**-isə **PointType** tipli dəyişən olub, təpə nöqtələrinin kordinatlarını verir.

46) **Procedure FillEllipse(x,y,rx,ry:inte-**

ger); proseduru ellipsi xətlə əhatələyib, onu rəngləyir. Burada **x,y**-ellipsin mərkəzinin koordinatları, **rx, ry** isə ellipsin piksellərlə üfüqi və şaquli radiuslarıdır.

47) **Procedure Sector(x,y:integer; BegA, EndA,rx,ry:Word);** proseduru ellips sektorunu çəkib, rəngləyir. Burada x,y-sektorun mərkəzinin koordinatları, **rx,ry**-sektorun piksellərlə üfüqi və şaquli radiusları, **BegA, EndA** isə elleps sektorunun uyğun olaraq başlanğıc və son bucaqlarıdır.

5.19. Turbo Pascal dilinin proqramlaşdırma mühiti

Turbo Pascal dilinin proqramlaşdırma mühiti TP adlı kataloqda yerləşdirilir. Burada Turbo Pascal-ı çağırmaq üçün **TURBO.EXE** faylını tapıb açmaq lazımdır. Nəticədə ekrana proqramlaşdırma sisteminin işçi stolu çıxarılır. Pəncərənin birinci sətrində mühitin baş menyusu adlanan bölmələr ardıcıllığı verilir. Menyu aşağıdakı bölmələrdən ibarətdir:

1) **File** (fayl) – fayllarla iş üçün və sistemdən çıxış üçün nəzərdə tutlub;

2) **Edit** (redaktə etmək) – sətirlərdə ola biləcək səhvlərin düzəldilməsi və müvəqqəti bufer yaddaşı ilə iş üçün nəzərdə tutulub;

3) **Search** (axtarmaq) – mətnin, prosedurun, funksiyanın və ya səhv olan yerin axtarılmasını təmin edir;

4) **Run** (iş) – proqramın yerinə yetirilməsini təmin edir;

5) **Compile** (kompilyasiya) – proqramın kompilyasiyasını təmin edir;

6) **Debug** (tənzimləmə) – proqramın tənzimlənməsini təmin edir;

7) **Tools** (alətlər) – köməkçi proqramların (utilitlərin) çağırılmasını təmin edir;

8) **Options** (variantlar) – proqramlaşdırma mühitinin qurulması;

9) **Window** (pəncərə) – pəncərələrlə işi təmin edir;

10) **Help** (kömək) – arayışlar bölməsinə müraciəti təmin edir.

Hər bir menyu bölməsi öz növbəsində alt bölmələrdən iba-

rətdir ki, onların köməyi ilə proqramlaşdırma mühitində proqramlarla işi təmin etmək olur. Proqram mətnlərinin redaktə etmə iş rejimindən, menyu bölmələrindən istifadə rejiminə keçmək üçün F10 düyməsindən, geriyə qayıtmaq üçün isə **ESC** düyməsindən istifadə olunur. Pəncərənin aşağı sətrində əsas funksional düymələrin təyinatı haqqında qısa arayışlar verilir. Ekranın yerdə qalan ikiqat çərçivəyə alınmış hissəsi proqram mətnlərinin daxil edilməsi və üzərində iş aparılmasını təmin edən redaktor pəncərəsidir. Bu pəncərənin yuxarı sətrində proqram mətnin oxunduğu disk faylının adı gətirilir ki, (yeni fayla **NONAMEOO.PAS** adı verilir). Turbo Pascal proqramlaşdırma mühitində eyni zamanda müxtəlif pəncərələrdə verilmək şərti ilə bir neçə proqramla işləmək olur. Mühit eyni zamanda redaktorun 9-a qədər pəncərəsində iş aparılmasına imkan verir. Turbo Pascal proqramlaşdırma mühitində redaktorun pəncərəsi ilə yanaşı tənzimləmə rejiminin pəncərəsi, proqramın yerinə yetirilmə nəticələrinin verilməsi pəncərəsi, arayışlar xidmətinin pəncərəsi də olur. *F1-F10* funksional düymələri, eləcə də onların *Alt, Ctrl və Shift* düymələri ilə kombinasiyaları Turbo Pascal proqramlaşdırma mühitinin idarə edilməsində isitfadə edilir. Bu düymələrdən bəzilərinin təyinatını verək:

1) *F1*-daxili arayışlar xidmətinə arayış üçün müraciəti təmin edir;

2) *F2*-redaktə edilən proqram mətnin disk faylına yazılışını təmin edir;

3) *F3*-disk faylındakı proqram mətnini redaktor pəncərəsinə çıxarır;

4) *F4*-tənzimləmə rejimində istifadə edilir: ondan proqramın yerinə yetirilməsini başlamaq və ya davam etdirmək üçün və kursorun durduğu sətrin yerinə yetirilməsindən əvvəl dayanmaq üçün istifadə olunur;

5) *F5*-aktiv pəncərəni bütün ekran boyunca açır;

6) *F6*-növbəti pəncərəni aktiv etmək üçün;

7) *F7*-tənzimləmə rejimində istifadə edilir: proqramın növbəti sətrinin yerinə yetirilməsi üçün, sətirdə prosedura (funksiyaya) müraciət varsa, bu prosedura girmək və onun birinci operatorunun yerinə yetirilməsindən əvvəl dayanmaq üçün istifadə edilir;

8) *F8*-tənzimləmə rejimində istifadə edilir: proqramın

növbəti sətrinin yerinə yetirilməsi üçün, sətirdə prosedura (funksiyaya) müraciət varsa, bu prosedura girib, onu yerinə yetirmək üçün istifadə edilir;

9) *F9*-proqramı kompilyasiyadan keçirmək, lakin yerinə yetirməmək;

10) *F10*-baş menyunun köməyi ilə iş rejiminin dialoq seçiminə keçidi təmin edir;

11) *Ctrl-F9* kombinasiyası ilə proqramın yerinə yetirilməsi: proqramın kompilyasiyasının aparılması, onun operativ yaddaşa yüklənib yerinə yetirilməsi və sonda yenidən Turbo Pascal mühitinə qayıdışı təmin edilir;

12) *Alt-F5* isə redaktor pəncərəsini, proqramın yerinə yetirilmə nəticələrinin çıxarılması pəncərəsi ilə əvəz edir;

13) *Ctrl-Del* – redaktorun buferinin təmizlənməsini təmin edir;

14) *Ctrl-Ins* – ayrılmış blokun redaktorun buferində saxlanılmasını təmin edir;

15) *Alt-X* – Turbo Pascaldan çıxışı təmin edir;

16) *Alt-F3* – aktiv pəncərəni bağlayır.

Beləliklə, birinci növbədə *Ctrl-F9* ilə proqramın işi yoxlanılır və *Alt-X* ilə Turbo Pascal-dan çıxış yerinə yetirilir. *F2* və *F3* istifadəçiyə kataloqu ilə işi təmin edəcək və *Alt-F5* ilə proqramın yerinə yetirilmə nəticələrini ekrana çıxarmaq olur.

Turbo Pascal proqramlaşdırma mühitinin mətn redaktoru proqram mətnlərinin yaradılması və redaktə edilməsi üçün geniş imkanlar yaradır. Proqramlaşdırma mühitinin redaktə etmə rejimində olması, redaktor pəncərəsində kursorun olması ilə müəyyən edilir. Redaktə etmə rejimi Turbo Pascal yüklənən kimi avtomatik olaraq işə hazır olur. Bu rejimdən Turbo Pascal-ın ixtiyari digər rejiminə funksional düymələrin köməyi ilə və ya baş menyudan lazım olan rejimin seçilməsi ilə keçmək olur. Menyudan seçimə keçmək üçün F10, çıxış üçün isə *Esc* düyməsindən istifadə edilir. Proqram mətni klaviaturadan daxil edilir, hər sətrin sonunda *Enter* düyməsi sıxmaqla yeni sətrə keçid yerinə yetirilir. Hər bir fayldakı simvolların sayı 64535-i aşmamalıdır və Turbo Pascal-ın kompilyatoru proqram sətrindəki ilk 126 simvolu qəbul edir. Turbo Pascal redaktorunda istifadə edilən əsas əmrlər aşağıdakılardır:

1) *Page Up*– bir səhifə yuxarıya yerdəyişmə
2) *Page Down* – bir səhifə aşağıya yerdəyişmə
3) *Home* – cari sətrin əvvəlinə keçid
4) *End* – cari sətrin sonuna keçid
5) *Ctrl-Page Up* – mətnin əvvəlinə keçid
6) *Ctrl-Page Down* mətnin sonuna keçid
7) *BackSpace* – kursordan soldakı simvolu silir
8) *Delete* - kursordan sağdakı simvolu silir
9) *Ctrl-Y* – kursorun durduğu sətri silir
10) *Enter* – yeni sətirə keçidi, sətrin bölünməsini təmin edir.

Turbo Pascal proqramlaşdırma mühitindəki əsas iş qaydalarına baxaq. Qeyd etdiyimiz kimi Turbo Pascal yüklənən kimi proqramlaşdırma mühiti proqram mühitinin redaktə etmə rejiminə keçir. Bu rejimdə yeni proqram qurmaq və ya mövcud proqramı redaktə etmək olur. Proqram mətnləri mühitdən kənarda fayllarda saxlanılır. Turbo Pascal-da işi başa çatdırdıqdan sonra proqram mətnini növbəti dəfə istifadə etmək üçün disk faylında saxlamaq olur. Disk faylı ilə proqramlaşdırma mühiti arasında verilənlər mübadiləsini həyata keçirmək üçün F2 (fayla verilənləri yazmaq üçün) və *F3* (fayldan verilənləri oxumaq üçün) düymələrindən istifadə edilir. Yeni proqram yaradılarkən, mühit bu proqram mətninin hansı adlı faylda yerləşdiriləcəyini bilmədiyindən ona **NONAME00.PAS** standart adını verir. Proqram mətninin faylda saxlanması üçün *F2* düyməsini sıxmaq lazımdır. Bu anda mühit proqram adını yoxlayır, bu ad standart **Noname** adı olduqda bu adın dəyişdirilməsi haqqında **SAVE FILE AS** formalı sorğu verilir. Bu sorğudan aşağıda faylın adının daxil edilməsi üçün sahə yerləşir ki, burada lazım olan adı yığıb, *Enter* düyməsini sıxmaq lazımdır. Nəticədə proqram mətni həmin adlı faylda saxlanılır. Əgər fayl adında ad genişlənməsi verilməyibsə, mühit fayla standart **Pas** ad genişlənməsi verir. Turbo Pascal-da işi sona çatdıran zaman burada redaktorda fayla yazılmamış proqram mətni qalırsa, mühit onu yaddaşda saxlanılması haqqında sorğu verir:

NONAME.PAS has been modified. Save?

Proqram mətnini faylda saxlamaq lazımdırsa, **Y**(**Yes** – Hə), lazım deyilsə, **N** (**No** – Yox) düyməsini sıxmaq lazımdır.

Proqram mətnini qurduqdan sonra onu yerinə yetirmək, yəni proqramı kompilyasiya etmək, ehtiyac varsa onu standart prosedur və funksiyalar kitabxanası ilə əlaqələndirmək, operativ yaddaşa yükləyib idarəetməni proqram mühitinə vermək lazımdır. Bu əməliyyat *Ctrl-F9* əmri ilə yerinə yetirilir. Əgər proqram mətnində hər hansı bir səhv yoxdursa, onda proqram mətni ardıcıl olaraq yerinə yetirilir və bu zaman ekrandakı kiçik bir pəncərədə kompilyasiyadan keçmiş sətirlərin sayı və operativ yaddaşın həcmi haqqında məlumatlar çıxır. Yüklənmiş proqrama idarəetməni verməmişdən əvvəl mühit ekranı təmizləyir, proqramın işləməsi pəncərəsini ekrana çıxarır, proqramın işi sona çatdıqdan sonra isə kompüterin idarə edilməsini yenidən üzərinə götürüb ekranda redaktorun pəncərəsini bərpa edir. Əgər proqramın yerinə yetirilməsinin hər hansı bir mərhələsində mühit səhv taparsa, o işi dayandırır, redaktorun pəncərəsini bərpa edir və kursoru səhv tapılmış proqram sətri üzərinə yerləşdirir. Bu zaman redaktorun yuxarı sətrində səhvin səbəbləri haqqında məlumat verilir. Səhvi düzəldib, proqramın yerinə yetirilməsini davam etdirmək lazımdır.

VI FƏSİL

C++ ALQORİTMİK DİLİ

6.1. Dilin əlifbası. Sabitlər

Dilin əlifbasına aşağıdakılar aiddir:

1) latın əlifbasının böyük (A-Z) və kiçik (a-z) hərfləri;

2) 0-9 ərəb rəqəmləri;

3) xüsusi işarələr: " { } , | [ ] ( ) + - / % \ ; ' : ? < = > _ ! & # ~ ^ . *

Əlifbanın simvollarından dilin aşağıdakı konstruksiyaları qurulur:

1) identifikatorlar;

2) işçi sözlər;

3) sabitlər;

4) əməliyyat işarələri;

5) ayrıcılar;

İdentifikator – birinci simvolu rəqəm olmayan, latın əlifbasının hərflərindən, onluq rəqəmlərdən və qeyd etmə _ işarəsindən ibarət ola bilən simvollar ardıcıllığıdır. Məsələn, **copy_32**, **alfa_24**, **x1**, **y2**, **alfa**, **ALFA** və s. Burada böyük və kiçik hərflər fərqləndirildiyindən, axırıncı iki identifikator müxtəlifdir. İdentifikatorun uzunluğu ixtiyari ola bilər, buradan yalnız birinci 32 simvol qəbul edilir.

İşçi sözlər, dildə xüsusi tətbiq üçün nəzərdə tutulmuş identifikatorlardır. C++ dilinin işçi sözləri aşağıdakılardır:

asm, **auto**, **break**, **case**, **catch**, **char**, **class**, **const**, **continue**, **default**, **delete**, **do**, **double**, **else**, **enum**, **extern**, **float**, **for**, **friend**, **goto**, **if**, **inline**, **int**, **long**, **new**, **operator**, **private**, **protected**, **public**, **register**, **return**, **short**, **signed**, **sizeof**, **static**, **struct**, **switch**, **template**, **this**, **throw**, **try**, **typedef**, **union**, **unsigned**, **virtual**, **void**, **volatile**, **while**.

Sabit qeyd olunmuş ədədi, sətri və ya simvol qiyməti ifadə edir. Sabitlər beş qrupa bölünür: tam, həqiqi (sürüşən onluq nöqtə ilə) sadalanan, simvol və sətir.

Tam sabitlər onluq, səkkizlik və onaltılıq ola bilir. Onluq tam sabit, bu ədəd sıfır deyilsə, sıfırla başlamayan onluq rəqəmlər ardıcıllığı kimi təyin edilib: 16, 49757, 0, 41, 73 və s. Müsbət tam sabitlərin dəyişmə diapazonu 0 – 4294967295 arasındadır. Mənfi tam sabitlər, işarənin dəyişdirilməsi əməliyyatı tətbiq olunmuş, işarəsiz sabitlərdir. Mənfi onluq sabitlərin mütləq qiyməti 2147483648-i aşmamalıdır.

Tam sabitlərin tipləri

Verilənlərin tipi	Sabitlərin qiymət diapazonları		
	Onluq	Səkkizlik	Onaltılıq
Int	0 – 32767	00 – 077777	0×0000 – 0×7FF
Unsigned int	0 – 32767	0100000 – 0177777	0×8000 – 0xFFFF
Long	32768 – 2147483647	0200000 – 017777777777	0×10000 – 0×7FFFFFF
Insigned long	2147483648 – 4294967295	020000000000 – 037777777777	0×80000000 – 0×FFFFFFF

Səkkizlik tam sabitlər həmişə sıfırla başlayır. Məsələn, 016 ədədi onluq say sistemində 14 qiymətini alır. Əgər səkkizlik sabitin yazılışında verilməsi mümkün olmayan 8 və ya 9 rəqəminə rast gəlinərsə, bu səhv kimi qəbul ediləcək. Müsbət səkkizlik sabitlər üçün mümkün qiymətlər diapazonu 00 – 037777777777 arasındadır. Mənfi səkkizlik sabitlər üçün isə mütləq qiymət 020000000000-ı aşmamalıdır.

Qarşısında 0× qoyulan onaltılıq rəqəmlər ardıcıllığı onaltılıq sabit sayılır. Onaltılıq rəqəmlərə onluq rəqəmlərdən başqa a (və ya A)-dan f (və ya F)-ə qədər latın hərfləri daxildir. Beləliklə, 0×16, onluq 22 qiymətinə, 0×F isə onluq 15 qiymətinə bərabərdir. Müsbət onaltılıq sabitlər üçün mümkün qiymətlər diapazonu 0×0 – 0×FFFFFFFF arasındadır. Mənfi səkkizlik sabitlər üçün isə mütləq qiymət 0×80000000-ı aşmamalıdır.

Qeyd edək ki, istifadəçi C++ dilində sabitin tipini aşkar şəkildə dəyişdirə bilər. Bunun üçün **L** və ya **l** (**long**) və **U** və ya **u** (**unsigned**) hərflərindən istifadə edilir. Məsələn, 65L sabiti long ti-pini alacaq, lakin əslində 65 qiyməti int tipinə aiddir. Bir sabit

üçün ixtiyari ardıcıllıqla iki **U** (**u**) və **L** (**l**) hərflərindən istifadə etmək olar. Məsələn, **0×22Ul**, **0×11Lu**, **0×55lu** sabitləri **unsigned long** tipinə aid olacaqlar.

Həqiqi sabitlər, yəni sürüşən onluq nöqtəli sabitlər aşağıdakı altı hissədən ibarət ola bilər: tam hissə (onluq tam sabit), onluq nöqtə, kəsr hissə (onluq tam sabit), eksponent əlaməti **e** və ya **E** simvolu, onluq tərtib göstəricisi (işarəsi də ola bilən onluq tam sabit), **F** (və ya **f**) yaxud **L** (və ya **l**) simvolları. Həqiqi sabitlərin yazılışında eyni zamanda olmamaq şərti ilə tam və ya kəsr hissə, onluq nöqtə və ya tərtib göstəricisi ilə eksponent simvolu, eləcə də **F**, **L** simvolları verilməyə də bilər. Məsələn, **66.**; **.0**; **.15**; **3.1415F**; **5.12e-2**; **4E+7L**; **3.77** .

Burada **F** (**f**) və ya **L** (**l**) simvolları verilmədikdə həqiqi sabitlər double verilənlər tipinə uyğun gəlir. Əgər bura **f** və ya **F** simvolları əlavə olunarsa, sabit float tipli olur. Sabitin ifadəsində **L** və ya **l** simvolları istifadə edildikdə, long double sabit tipli olur.

Həqiqi tipli verilənlər

Verilənlər tipi	Ölçü (bitlərlə)	Qiymətlər diapazonu
float	32	3.4E–38 - 3.4E+38
double	64	1.7E–308 - 1.7E+308
long double	80	3.4E–4932 - 1.1E+4932

Sadalanan sabitlər enum işçi sözünün köməyi ilə daxil edilir. Faktiki olaraq bu sabitlər özəl və istifadə üçün əlverişli işarələmələri olan int tipli tam sabitlərdir. İşarələnmə kimi işçi sözlərlə üst-üstə düşməyən ixtiyari identifikatorlardan, adlardan istifadə olunur. *Məsələn*,

```
enum{one =1,two =2,three =3};
```

Burada **enum** – verilənlərin sadalanan tipini təyin edən işçi söz, **one**, **two**, **three** isə 1, 2, 3 sabitlərinin işarələnməsi üçün istifadəçinin daxil etdiyi şərti adlardır. Bu cür təyindən sonra proqramda, məsələn, 2 sabit əvəzinə onun two işarəsindən də istifadə etmək olar. Əgər təyinatda «=» işarəsini verməyib, sadalanan sabitlərə onların qiymətlərini mənimsətməsək, onda sadalanan sabitləri avtomatik olaraq soldan başlayaraq 0,1,2,...

qiymətlərini alacaq. Məsələn, **enum{zero, one, two, three};** nəticəsində sadalanan sabitlər **zero==0**; **one==1**; **two==2**, **three==3** qiymətlərini alacaqlar. Sadalanan sabitlərin hər adımda bir vahid artırılması qaydası, onlardan birincisinə konkret qiymət mənimsədildiyi halda da doğrudur. *Məsələn*,

enum{ten=10,three=3,four,five,six};

təyini **ten==10, three==3, four==4, five==5, six==6** sabitlərini verəcəkdir. Sadalanan sabitlərin adları bir-birindən fərqlənməlidir, lakin müxtəltf sabitlər eyni qiymətlər ala bilər. Məsələn, **enum{zero,nought=0,one,two,p=2, three};** təyini **zero==0, nought==0, one==1, two==2, p==2, three==3** sabitlərini verəcək. Qeyd edək ki, sadalanan sabitlərin aldığı qiymətlər ifadə şəklində də verilə bilər. Məsələn, **enum{two=2,four=two*2}** təyini **two==2, four==4** sabitlərini verəcəkdir.

Sadalanan sabitlər üçün verilmiş siyahıya uyğun tipin adı da daxil edilə bilər. Tipin adı enum sözü ilə fiqurlu mötərizə arasında verilən ixtiyari identifikatordur. Məsələn, **enum week{sunday,monday,tuesday,wednesday,thursday, friday,saturday};** burada **sunday==0, monday==1** və s. təyin olunmaqla yanaşı, week sadalanan tip adı, daxil edilir və bu addan sonralar proqramda istifadə etmək olar.

Simvol sabitləri – apostrof işarəsi arasına alınmış bir və ya iki simvoldan ibarətdir. Bir simvollu sabitlər standart char tiplidirlər. Məsələn, **'z'**, **'*'**, **'\012'**, **'\0'**, **'\n'** – birsimvollu, **'db'**, **'\x07\x07'**, **'\n\t'** – ikisimvollu sabitlərdir. Qeyd edək ki, burada istifadə edilən **'\'** işarəsi aşağıdakı hallarda verilir:

1) qrafik təsviri olmayan kodların yazılışı zamanı;
2) (**'**) apostrof simvolunu verərkən;
3) (****) simvolunu verərkən;
4) (**?**) sual işarəsini verərkən;
5) (**"**) dırnaq işarəsini verərkən.

Ümumiyyətlə, «\» işarəsi ilə başlayan bütün mümkün simvollar ardıcıllığı aşağıdakılardır:

1) **\a-bel(audible bell)** – səs siqnalı;

2) **\b-bs (backspace)** – bir addım geri;
3) **\f-ff (form feed)** – səhifəni keçirmək;
4) **\n-lf (line feed)** – sətri keçirmək;
5) **\r-cr (carriage return)** – yeni sətrə keçid;
6) **\t-ht (horizontal tab)** – üfüqi tabulyasiya;
7) **\v-vt (vertical tab)** – şaquli tabulyasiya;
8) **\\-\ (backslash)** – əks çəpəki xətt;
9) **\'-' (single quote)** – apostrof;
10) **\"-" (double quote)** – ikiqat dırnaq;
11) **\?-? (question mark)** – sual işarəsi;
12) **\000** – simvolun səkkizlik kodu;
13) **\xhh** – simvolun onaltılıq kodu.

Sətir və ya sətri sabit – ikiqat dırnaq arasına alınmış simvollar ardıcıllığına deyilir. Proqramda ardıcıl verilən sətirlər nəticədə birləşdirilir. Dırnaqlar sətrə aid deyil və yalnız sətrin proqram daxilində məhdudlaşdırıcılarıdır. Sətirdə yalnız bir simvol da ola bilər. Məsələn, "A". Bu halda sətrin uzunluğu 2-yə bərabərdir. Sətir boş da ola bilər: " ", bu halda sətrin uzunluğu 1-ə bərabərdir.

6.2. Əməliyyatlar. Ayırıcılar

Əməliyyat işarələri ifadələri formalaşdırmağa və sonradan hesablamağa imkan verir. C++ dilində aşağıdakı əməliyyat işarələri təyin edilib:

[ ]; ( ); .; ->; ++; --; &; *; +; -; ~; !; sizeof; /; %; <<; >>; <; >; <=; >=; ==; !=; ^; |; &&; ||; ?:; =; *=; /=; %=; +=; -=; <<=; >>=; &=; ^=; |=; ', #; ##; ::; .*; ->*; new; delete; typeid.

Bu işarələrə uyğun əməliyyatların standart imkanlarını qeyd edək.

Unar əməliyyatlar:

1) **&** – operandın ünvanının alınması;
2) ***** – operanda ünvanı üzrə müraciət;
3) **-** - unar mənfi hesabi operandın işarəsinin dəyişdirilməsi;

4) **+** - unar müsbət;

5) **!** – operandın qiymətinin məntiqi inkarı, əgər operand sıfırdan fərqli, yəni həqiqətdirsə, nəticə 0 olacaq, operand sıfır, yəni yalandırsa, nəticə 1 olacaq.

Beləliklə, **!1==0**; **!2==0**; **!(-5)==0**; **!0==1** olar.

6) **++** operandın bir vahid artırılması, burada operand sabit ola bilməz;

7) **--** operandın bir vahid azaldılması;

8) **sizeof** – operandın ölçüsünün (baytlarla) təyini. İki formatı mövcuddur: **sizeof** *unar ifadə* və **sizeof** (*tip*).

Binar əməliyyatlar. Bu əməliyyatlar aşağıdakı qruplara bölünür:

1) additiv
2) multiplikativ
3) yerdəyişmə
4) dərəcələr üzrə
5) münasibət əməliyyatları
6) məntiqi
7) mənimsətmə
8) strukturlaşdırılmış obyektin elementlərinin seçimi
9) siniflərin elementləri ilə əməliyyatlar
10) vergül işarəsi əməliyyat kimi
11) mötərizələr əməliyyat kimi.

Additiv əməliyyatlar:

1) **+** binar toplama (hesabi ifadələrin toplanması)
2) **–** binar çıxma (hesabi ifadələrin çıxılması)

Multiplikativ əməliyyatlar:

1) ***** – hesabi ifadələrin vurulması;

2) **/** – hesabi ifadələrin bölünməsi. Tam tipli ifadələrin bölünməsi zamanı nəticə tam ədədə qədər yuvarlaqlaşdırılır, məsələn, 20/3 ifadəsi 6-ya, -20/3, -(20)/3, 20/(-3) ifadələri isə -6-ya bərabərdir.

3) **%** – tam tipli ifadələrin bölünməsi zamanı qalıq həddinin alınması.

Yerdəyişmə əməliyyatları (bu əməliyyatlar yalnız tam tipli ifadələr üçün təyin olunub). Bu əməliyyatlar üçün format aşağıdakı kimidir: sol ifadə_yerədəyişmə əməliyyat işarəsi_sağ ifadə.

1) << soldakı tam tipli ifadənin bitlərlə qiymətinin sağdakı tam tipli ifadə qədər sola yerdəyişməsi;

2) >> sağdakı tam tipli ifadənin bitlərlə qiymətinin soldakı tam tipli ifadə qədər sağa yerdəyişməsi.

Dərəcələr üzrə əməliyyatlar:

1) **&** – tam tipli ifadənin bitlərlə qiymətinin dərəcələr üzrə konyunksiyası (və);

2) **|** – tam tipli ifadənin bitlərlə qiymətinin dərəcələr üzrə dizyunksiyası (və ya);

3) **^** – tam tipli ifadənin bitlərlə qiymətinin dərəcələr üzrə dizyunksiyasının ləğv edilməsi.

Dərəcələr üzrə əməliyyatların və yerdəyişmə əməliyyatlarının xüsusiyyətlərini açıqlayan bir proqram quraq:

```
#include<iostream.h>
void main( )
{cout<<"\n4<<2 ="<<(4<<2);
cout<<"\t5>>1 ="<<(5>>1);
cout<<"\n6&5 ="<<(6&5);
cout<<"\t6|5 ="<<(6|5);
cout<<"\t6^5 ="<<(6^5);
}
```

Proqramın yerinə yetirilmə nəticələri:

4<<2 bərabərdir **16**; **5>>1** bərabərdir **2**; **6&5** bərabərdir **4**; **6|5** bərabərdir **7**; **6^5** bərabərdir **3**.

Qeyd edək ki, 4 rəqəmi üçün ikilik kod 100, 5 üçün 101, 6 üçün 110 və s. olduğundan 100 kodunun sola 2 mövqe yerdəyişməsi nəticəsində 10000 alınır ki, onun da onluq qiyməti 16-dır. Sonrakı əməliyyatlar analoji aparılır. Nəzərə alaq ki, sola n sayda mövqe yerdəyişməsi uyğun ifadənin 2^n-ə bölünməsi (qalıq həddi atılmaqla) deməkdir.

Münasibət əməliyyatları (Müqayisə):

1) **<** kiçikdir;

2) **>** böyükdür;

3) **<=** kiçik bərabərdir;

4) **>=** böyük bərabərdir;

5) **==**bərabərdir;

6) **!=** bərabər deyil.

Məntiqi əməliyyatlar:

1) **&&** hesabi ifadə və ya münasibətlərin konyunksiyası. Nəticə ya 0 (yalan), ya da 1 (həqiqət) ola bilər;

2) **||** hesabi ifadə və ya münasibətlərin dizyunksiyası. Nəticə ya 0 (yalan), yada 1 (həqiqət) ola bilər.

Mənimsətmə əməliyyatları:

1) **=** sağdakı ifadə qiymətini soldakı dəyişən adına mənimsədir, məsələn, **P=10.3-2*x;**

2) ***=** ifadədə iştirak edən hər iki operandın hasili soldakı operanda mənimsədilir, məsələn, **P*=2** ifadəsi **P=P*2** ilə ekvivalentdir.

3) **/=** ifadədəki solda duran operandın sağdakı operanda nisbəti soldakı operanda mənimsədilir. Məsələn, **P/=3.5** ifadəsi **P=P/3.5** ilə ekvivalentdir;

4) **%=** soldakı tam operandın sağdakı tam operanda bölünməsi nəticəsində alınan qalıq soldakı operanda mənimsədilir, məsələn, **N%=3** ifadəsi **N=N%3** ilə ekvivalentdir.

5) **+=** ifadədəki hər iki operandın cəmi soldakı operanda mənimsədilir, məsələn, **A+=B** ilə **A=A+B** ekvivalentdir.

6) **-=** ifadədəki sol və sağ operandların fərqi soldakı operanda mənimsədilir, məsələn, **x-=4.3-y** ilə **x=x-(4.3-y)** ekvivalentdir.

7) **<<=** soldakı tam tipli operandın bitlərlə qiymətinin sağdakı tam tipli qiymət qədər sola yerdəyişməsinin qiyməti soldakı operanda mənimsədilir, məsələn, a **<<=** 4 ilə $a = a$ **<<** 4 ekvivalentdir.

8) **>>=** soldakı tam tipli operandın bitlərlə qiymətinin sağdakı tam tipli qiymət qədər sağa yerdəyişməsinin qiyməti soldakı operanda mənimsədilir, məsələn, a **>>=** 4 ilə $a = a$ **>>** 4 ekvivalentdir.

9) **&=** soldakı tam tipli operandın bitlərlə qiymətinin dərəcələr üzrə sağdakı tam tipli operandın bitlərlə qiyməti ilə konyunksiyasının ifadəsi soldakı operanda mənimsədilir, məsələn, e**&=** 44 ilə $e = e$ & 44 ekvivalentdir.

10) **|=** soldakı tam tipli operandın bitlərlə qiymətinin dərəcələr üzrə sağdakı tam tipli operandın bitlərlə qiyməti ilə dizyunk-

siyasının ifadəsi soldakı operanda mənimsədilir, məsələn, $a|=b$ ilə $a=a|b$ ekvivalentdir.

11) ^= hər iki tam operandın bitlərlə qiymətlərinə dərəcələr üzrə dizyunksiyasının ləğv edilməsinin tətbiqinin nəticəsi soldakı operanda mənimsədilir, məsələn, $z\hat{}=x+y$ ilə $z=z\hat{}(x+y)$ ekvivalentdir.

Bu əməliyyatları bir misal üzərində göstərək.

```
//P2.CPP
# include<iostream.h>
void main()
{int k;
cout<<"\n\n k=35/4="<<(k=35/4);
cout<<"\t k/=1+1+2="<<(k/=1+1+2);
cout<<"\n k*=5-2="<<(k*=5-2);
cout<<"\t k%=3+2="<<(k%=3+2);
cout<<"\n k+=21/3="<<(k+=21/3);
cout<<"\t k-=6-6/2="<<(k-=6-6/2);
cout<<"\n k<<=2="<<(k<<=2);
cout<<"\t k>>=6-5="<<(k>>=6-5);
cout<<"\n k&=9+4="<<(k&=9+4);
cout<<"\t k|=8-2="<<(k|=8-2);
cout<<"\n k^=10="<<(k^=10);
}
```

Nəticədə alarıq:

k=35/4 bərabərdir **8**; **k/=1+1+2** bərabərdir **2**;
k*=5-2 bərabərdir **6**; **k%=3+2** bərabərdir **1**;
k+=21/3 bərabərdir 8; **k-=6-6/2** bərabərdir **5**;
k<<=2 bərabərdir **20**; **k>>=6-5** bərabərdir **10**;
k&=9+4 bərabərdir **8**; **k|=8-2** bərabərdir**14**;
k^=10 bərabərdir **4**.

Strukturlaşmış obyektin elementlərinin seçimi əməliyyatları:

1) . (nöqtə) strukturlaşmış obyektin elementinin birbaşa seçimi, seçimin formatı: strukturlaşmış obyektin adı. elementin adı;

2) -> strukturlaşmış obyektin elementinin dolayısı yolla seçimi, seçimin formatı: strukturlaşmış obyektin göstəricisi -> ele-

mentin adı.

Siniflərin elementləri ilə əməliyyatlar:

1) **.*** obyektin adı və elementə göstərici ilə sinif elementinə birbaşa müraciət;

2) **->*** obyektə göstərici və elementə göstərici ilə sinif elementinə dolayısı yolla müraciət;

3) **::** görünüş oblastına müraciət əməliyyatı iki formaya, binar və unar formalara malikdir.

Binar forma sinif elementinə keçməyə imkan verir, unar əməliyyat isə müəyyən funksiya üçün xaric olan yaddaş oblastına keçməyə imkan verir.

Vergül işarəsi əməliyyat kimi: **,**(vergül) işarəsi ifadədə hesablamaları soldan sağa doğru qruplaşdırır, hesablamanın nəticəsinin tipi qrupdakı axırıncı sağ ifadənin tipi qəbul edilir.

Misal.

```
#include<iostream.h>
void main()
{int d;
cout<<"\n d=4,d*2="<<(d=4,d*2);
cout<<",d="<<d;
}
```

Nəticədə alarıq: **d=4,d*2** bərabərdir **8,d** bərabərdir**4.**

Uyğun olaraq ( ) yumru və [ ] kvadrat mötərizələr funksiyalara müraciət və massiv elementlərinin indekslərinin ifadəsi üçün istifadə edilərkən binar əməliyyat rolunu oynayır.

Funksiyaya müraciət zamanı hökmən yumru mötərizədən istifadə edilir: funksiyanın adı (arqumentlər siyahısı). Massivlərdə indekslərin ifadəsi üçün kvadrat mötərizədən aşağıdakı formatda istifadə edilir:

massivin adı **[***indeks***]**.

C++ dilində massiv indeksləri sıfırdan başlayaraq dəyişir, məsələn, üç elementli int **z[3]** massivi **z[0]**, **z[1]**, **z[2]** elementlərindən ibarət olacaq.

Misal.

```
//P4.CPP
#include<iostream.h>
void main()
{char x[]="BAKI";
```

```
int i=0;
while(x[i]!='\0')
cout<<"\n"<<x[i++];
}
```

Nəticədə «BAKI» sözü yuxarıdan aşağıya doğru sütun kimi çap olunacaq.

Şərti əməliyyat. Bu əməliyyatın ümumi yazılış forması aşağıdakı kimidir:

ifadə 1 ? ifadə 2: ifadə 3

Burada əvvəlcə *ifadə 1* hesablanır, əgər onun qiyməti doğru, yəni sıfırdan fərqlidirsə, onda *ifadə 2*-nin qiyməti hesablanır və bu qiymət nəticə olur. Əgər *ifadə 1*-in aldığı qiymət sıfır olursa, onda nəticə kimi *ifadə 3*-ün qiyməti qəbul edilir. Məsələn, `x<0?-x:x;` Bu əməliyyat `x` dəyişəninin mütləq qiymətini təyin edir.

Tipin aşkar təyini əməliyyatının iki müxtəlif forması var. Birinci kanonik forma:

(tipin adı) operand,

ikinci funksional forma:

tipin adı (operand).

Məsələn,

`(long)1, (char)1, long(2), double(2).`

New və delete əməliyyatları.

`new` *tip adı*

və ya

`new` *tip adı_qiymət*

əməliyyatı əsas yaddaşda tip adı ilə təyin edilən verilənlər tipinin ölçülərinə uyğun sahə ayırır. Ayrılmış sahəyə qiymət verilir, lakin bu məcburi element deyil. Əməliyyatın tətbiqi sintaksisi aşağıdakı kimidir:

göstərici = `new` tip adı_qiymət,

burada məcburi olmayan qiymət yumru mötərizələrdə verilən ifadədir, göstəricinin tipi isə tip adı ilə verilən tiplə üst-üstə düşməlidir.

`New` və `delete` əməliyyatlarının tətbiqi zamanı göstəricilərdən istifadə olunur, lakin əvvəlcədən hər bir göstərici təyin olunmalıdır.

Göstəricinin təyini aşağıdakı kimidir:

*tip*göstəricinin adı;*

burada göstəricinin adı-identifikator, yəni dəyişən adıdır, tipin yerində isə **int**, **long**, **float**, **double**, **char** standart tipləri ola bilər. Məsələn, **int*h;** təyini üçün yaza bilərik **h=new int(15);** gələcəkdə bu ayrılmış yaddaş sahəsinə keçid üçün ***h** ifadəsini vermək kifayətdir.

New əməliyyatı ilə ayrılmış yaddaş sahəsini azad etmək üçün **delete** göstərici; operatorundan istifadə olunur, burada göstərici **new** əməliyyatında verilən göstəricidir.

Məsələn, **delete h;**

C++ dilində əməliyyatların yerinə yetirilmə ardıcıllığı aşağıdakı kimidir:

ardıcıllıq	Əməliyyatlar	assosiativlik
1	**( ) [ ] -> :: .**	→
2	**! ' ~ + - ++ -- & *** (tip) **size of new delete** tip ( )	←
3	**. * -> ***	→
4	*** / %** (multiplikativ binar əməliyyatlar)	→
5	**+ -** (additiv binar əməliyyatlar)	→
6	**<< >>**	→
7	**< <= >= >**	→
8	**== !=**	→
9	**&**	→
10	**^**	→
11	**|**	→
12	**&&**	→
13	**||**	→
14	**? !** (şərti əməliyyat)	←
15	**= *= /= %= += -= &= ^= |= <<= >>=**	←
16	**,** (vergül əməliyyatı)	→

Ayrıcılar dilin əlifbasına aiddir:

[] () { } , ; : ... * = # &

Kvadrat mötərizələr '[ ]'bir və çox ölçülü massivlərin indekslərini məhdudlaşdırmaq üçün istifadə olunur.

Məsələn,

//birölçülü beş elementli massiv:

```
int A[]={1,3,5,7,9};
```

//e- ikiölçülü massiv-3×2 ölçülü matris:

```
int e[3][2];
```

yumru mötərizələrdən '( )' aşağıdakı hallarda istifadə olunur:

1) şərt operatorunda şərti ifadələri ayırır:

Məsələn, `if(x<0)x=-x;`

2) ixtiyari funksiyanın təyini, təsviri və çağrılmasında uyğun olaraq formal və faktiki parametrləri ayırırlar. *Məsələn*,

```
float F(float x,int k)// funksiyanın təyini
{funksiyanın gövdəsi}
float F(float,int);// funksiyanın təsviri
.....
F(z,n); // funksiyaya müraciət
```

3) funksiyaya göstəricinin təyini zamanı istifadə olunur. *Məsələn*, `int(*func)(void);`

4) ifadədə əməliyyatların təbii yerinə yetirilmə ardıcıllığını dəyişdirməyə imkan verir. *Məsələn*, `y=(a+b)/c;`

5) dövr operatorlarında istifadə olunur. *Məsələn*,

`for (i=0,j=1;i<j;i+=2,j++)` dövrün gövdəsi;

6) tipin aşkar təyini əməliyyatında istifadə olunur, məsələn,

```
long i=12L;int j;
```

Fiqurlu mötərizələr '{ }'

Qurma operatorun və ya blokun əvvəlini və sonunu işarə edir. Məsələn, şərt operatorunda qurma operatordan aşağıdakı kimi istifadə etmək olar: `if (d>x) {d--; x++;}`

Fiqurlu mötərizələrdən struktur, birləşmə və siniflərin təyinində elementlər siyahısının ayrılması üçün istifadə olunur. Fiqurlu mötərizələrdən həmçinin massiv və strukturların təyinii zamanı qiymətlərin mənimsədilməsində istifadə olunur. *Məsələn*,

```
int M[]={1,2,3,4,5,6,7,8,9,10}
```

Vergül "," siyahılardakı elementləri bir-birindən ayırır. Eləcə də dövr operatorunun başlığında ayırıcı kimi vergüldən istifadə olunur. *Məsələn*,

```
for (x=p1,y=p2,i=2;i<n;
        z=x+y,x=y,y=z,i++);
```

Vergüldən ayırıcı kimi həmçinin eyni tipli dəyişənlərin təsviri və təyini zamanı da istifadə olunur. *Məsələn,* `int i, n;float x,y,z;`

Nöqtə vergül "`;`" işarəsi hər bir operatorun, təyinatın (funksiya təyinatından başqa) və təsvirin sonunda verilir. Sonunda nöqtə vergül işarəsi verilən ixtiyari mümkün ifadə operator kimi qəbul olunur. Bu işarə boş sətirdən sonra qoyulduqda, həmin sətir boş operator kimi qəbul edilir.

İki nöqtə "`:`" işarəsi nişan və onun təyin etdiyi operatoru bir-birindən ayırmaq üçün istifadə olunur:

nişan: operator;

Məsələn, `5:x=a*b`. Nişan identifikator olduğundan aşağıdakı yazılışlarda doğrudur:

```
xyz: a=(b-c)*(d-c); cc:z*=1;
```

Nöqtələr işarəsi "..." boş yer qoyulmadan bir-birnin ardınca verilən nöqtələrdir. Bu işarədən funksiyanın təyini və təsviri zamanı parametrlərin dəyişən sayda olduğunu göstərmək üçün istifadə olunur.

Ulduz "`*`"" işarəsi vurma əməliyyatını göstərmək, "`=`" bərabər işarəsi mənimsətmə əməliyyatını yerinə yetirmək üçün istifadə olunur. Nömrə "`#`" işarəsi prosessorun əmrlərini işarə etmək üçün, "`&`" işarəsi isə dəyişənlərin təyini zamanı istifadə olunur. *Məsələn,*

```
int B;
int & A=B;
```

6.3. Verilənlərin tipləri. Törəmə tiplər. Obyektlər

Yuxarıda təyin etdiyimiz sabitlərə dəyişənlər uyğun gəlir. Dəyişən adı kimi identifikatordan istifadə edilir. Dəyişənin ala biləcəyi qiymətlər çoxluğu, adətən onunla eyni tipli olan sabitlər çoxluğu ilə üst-üstə düşür. Beləliklə, həqiqi, tam, simvol tipli dəyişənlər daxil edilir. Qeyd edək ki, simvol (char) tiplər bəzən tam tiplərə aid edilir. Tam və həqiqi tiplər hesabi tiplər qəbul edilir. Hesabi tip (simvol tip də daxil olmaqla) skalyar tiplərin xüsusi halıdır. Skalyar tiplərə hesabi tiplərdən başqa göstəricilər, istinadlar və sadalanmalar da aiddir. Dəyişənlər təsvirlərin və təyin

etmələrin köməyi ilə tiplərə ayrılır. Təsvirdən fərqli olaraq təyin etmələr obyekti (məsələn, dəyişən) daxil etməklə yanaşı, onun üçün yaddaşda yer ayrılması haqqında kompilyatora göstəriş verir. Əsas tip dəyişənlərin təsvir və təyin etmələri üçün aşağıdakı işçi sözlərdən istifadə edilir:

1) **char** (simvol)
2) **short** (qısa tam)
3) **int** (tam)
4) **long** (uzun tam)
5) **float** (həqiqi)
6) **double** (ikiqat dəqiqlikli həqiqi)
7) **void** (qiymətin verilməməsi)

Dəyişənlərin təyini zamanı onlara başlanğıc qiymətlər də vermək olar. *Məsələn*, **int x=5; double pi=3.14159265358979323 85; char nl='\n'; long fl=0L;** və s. Tipin ifadəsində eyni zamanda bir neçə işçi sözdən istifadə etmək olar. Məsələn, **long double z,s;** təyini yüksək dəqiqlikli həqiqi tip **z** və **s** dəyişənlərini daxil edir, lakin onlara aşkar formada qiymət mənimsətmir. Təsvirdə eyni zamanda bir neçə dəyişən adı vermək olar. *Məsələn*, **int x,t,z; char a,b,c;** və s.

Hesabi və ya simvol tipli dəyişənlərin təsvirində onların dəyişmə diapazonunu dəqiq təyin etmək üçün **unsigned** (işarəsiz) və **signed** (işarəli) işçi sözlərindən istifadə olunur. *Məsələn*,

unsigned int i,j,k; (qiymət 0-65535 aralığındadır),

unsigned long L,M,N; (qiymət 0-4294967295 aralığındadır),

unsigned char c,s; (qiymət 0-255 aralığındadır).

Tiplərin təyinində **int**, **char**, **short**, **long** işçi sözləri **signed int**, **signed char**, **signed short**, **signed long** sözləri ilə ekvivalentdirlər. Buna görə də təyin və təsvirlərdə **signed** sözü adətən istifadə edilmir. Yalnız **unsigned** sözünün verilməsi **unsigned int** sözünə ekvivalentdir.

Verilənlərin əsas tipləri

Verilənlərin tipi	*Ölçü (bit)*	*Qiymətlər diapazonu*	*Tipin təyinatı*
unsigned char	8	0 ... 255	Kiçik tam ədədlər və simvol kodları
char	8	-128 ... 127	Çox kiçik tam ədədlər
enum	16	-32768 ... 32767	Tam qiymətlərin sadalanan külliyatı
unsigned int	16	0 ... 65535	Böyük tam ədədlər və dövr sayğacları
short int	16	-32768 ... 32767	Çox da böyük olmayan tam ədədlər və sayğaclar
int	16	-32768 ... 32767	"__"
unsigned long	32	0 ... 4294967295	Astronomik məsafələr
long	32	-2147483648 2147483647	Böyük ədədlər
float	32	3.4E-38 ... 3.4E+38	Elmi hesablamalar (7 əhəmiyyətli rəqəm)
double	64	1.7E-308...1.7E+308	Elmi hesablamalar (15 əhəmiyyətli rəqəm)
long double	80	3.4E-4932 ... ... 1.1E+4932	Maliyyə hesabları (19 əhəmiyyətli rəqəm)

Mürəkkəb tip təsvirlərin verilməsi üçün **typedef** işçi sözündən istifadə olunur. *Məsələn,*

```
Typedef unsigned char CH;
CH S;
```

təsviri ilə yeni **CH** tipi daxil edilir ki, bu tip **unsigned char** tipi və bu tipdən olan **S** dəyişəni üçün qısa işarələmədir.

Baza **char, int, signed, double, long, unsigned, float, short, void** tiplərindən "*", "&", "[ ]", "( )" əməliyyatlarının köməyi ilə törəmə tipləri qurmaq olar. Mümkün tipləri **type** adı ilə işarə edib, bəzi törəmə tiplərin formatlarına baxaq:

1) **type** ad[ ] – **type** tipli obyektlər massivi. Məsələn, **long int M[6]** – **long int** tipli altı obyekti işarə edir, həmin obyektlərə **M[0], M[1], M[2], M[3], M[4], M[5]** indeksli dəyişənlərinin köməyi ilə müraciət etmək olur.

2) **type1** ad (**type2**) **-type2** tipli arqumenti olan və type1 tipli qiymət qaytaran funksiya. Məsələn, **int f1(void);** – arqumentlər tələb etməyən və **int** tipli qiymət qaytaran funksiya, **void f2 (double);** – **double** tipli arqument qəbul edən və qiymət qaytarmayan funksiya.

3) **type***ad-**type** tipli obyektə göstərici. *Məsələn*, **char*p;** – **char** tipli obyektlərə **p** göstəricisini təyin edir.

4) **type***ad[ ]; **type** tipli obyektlərə göstəricilər massivi.

5) **type** (*ad) [ ]; – **type** tipli obyektlər massivinə göstərici.

6) **type1***ad**(type2)**; – **type2** tipli arqument alan və **type1** tipli obyektə göstərici qaytaran funksiya.

7) **type1**(*ad) **(type2)**; – **type2** tipli parametr alan və **type1** tipli qiymət qaytaran funksiyaya göstərici.

8) **type1*** (*ad) **(type2)**; – **type2** tipli parametr alan və **type1** tipli obyektə göstərici qaytaran funksiyaya göstərici.

9) **type&**ad=**type** tipli obyektin adı; – **type** tipli obyektə istinad.

10) **type1**(&ad) **(type2)**; – **type2** tipli parametr qəbul edən və verilmiş **type1** tipli qiymət qaytaran funksiyaya istinad.

11) **struct** ad{**type1** ad1; **type2** ad2;}; – **type1** və **type2** tipli komponentləri olan struktur tipi.

12) **union** ad**{type1** ad1**; type2** ad2**;}**; – birləşmə tipi.

13) **class** ad**{type1** ad1**; type2** ad2**;(type3);}**; – **type1** tipli obyekt və **type3** tipli arqumenti olan **type2** tipli funksiya kimi komponentləri olan sinif.

Bütün mümkün törəmə tipləri skalyar (**scalar**) tiplərə, aqreqat (**agregate**) tiplərə və funksiyalara (**function**) bölünürlər. Skalyar tiplərə hesabi tiplər, sadalanan tiplər, göstəricilər və istinadlar aiddir. Aqreqat tiplərə və ya struktur tiplərə massivlər, strukturlar, birləşmələr və siniflər aiddir.

C++ dilində obyekt, obyektyönümlü proqramlaşdırmanın termini kimi qəbul edilib və obyekt hər hansı sinfə aid edilir. Dəyişən-yaddaşın adlı oblastı kimi obyektin xüsusi halıdır. **V=E** mənimsətmə əməliyyatı **V** dəyişən adından və hər hansı **E** ifadəsindən ibarətdir. Dəyişən adı daha ümumi anlayışın "solda verilməsi mümkün olan ifadə" (left value expression) və ya l-ifadə anlayışının xüsusi halıdır. l-ifadə hər hansı obyektə ümumi hal üçün istinadı təyin edir. Bu cür istinadın xüsusi halı dəyişən adıdır. Beləliklə, obyekt yaddaşın bir oblastı kimi təyin edilir. Obyektlə müəyyən qiymət bağlı olduğundan, obyekt üçün l-ifadə ilə yanaşı tip də verilir. Tip obyekt üçün yaddaşda tələb olunan miqdarda yer ayırır, obyektlə aparıla bilən əməliyyatlar külliyatını təyin edir.

l-ifadələrə aşağıdakılar aiddir: skalyar, hesabi və simvol dəyişənlərin adları, massivlərə aid dəyişən adları, göstərici adları, strukturlaşmış verilənlərin (struktur, birləşmə, sinif) elementlərinin dəqiqləşdirilmiş adları (struktur adı.element adı;), struktur verilənlərin elementlərinin dolayısı yolla seçimini təmin edən ifadələr (obyektə göstərici **->** element adı;), yumru mötərizəyə alınan l-ifadələr, obyektlərə istinadlar, '*****' ad dəyişdirmə əməliyyatlı ifadələr, obyektlərə istinadlar qaytaran funksiya çağırışları.

l-ifadələrlə yanaşı sağda verilməsi mümkün olan ifadələr də təyin edilib ki, onlardan mənimsətmə operatorunun sol tərəfində istifadə etmək olmaz, məsələn, funksiya adı, sabit adı.

Obyektlər üçün tiplərdən başqa, aşkar və ya susmaqla aşağıdakılar təyin edilir: yaddaş sinifi (obyektin yerləşdirilməsini verir), identifikator obyekti ilə əlaqəli olan fəaliyyət oblastı, obyektin görünüşü, obyekt və onların adlarının mövcudluq davamiyyəti, əlaqələndirmə tipi.

Yaddaş sinifi, obyektin yaddaşda yerləşməsi qaydasını və mövcudluq zamanım təyin edir. Yaddaş sinfinin aşkar verilməsi üçün obyektin təsvirində aşağıdakılardan istifadə edilir:

1) **auto** (avtomatik verilən lokal yaddaş)-blok obyektlərinin, məsələn, funksiyanın gövdəsində təyini zamanı istifadə edilə bilər;

2) **register** (avtomatik verilən registr yaddaşı)-obyekt qiymətlərinin yerləşdirilməsi üçün əsas yaddaş hissələrindən deyil, registrlərdən istifadə edilir;

3) **static**-obyekti, onun təyin olunduğu proqram mətninin verildiyi fayl daxilində müəyyən edir. Bu yaddaş sinifi dəyişən və funksiyalara verilə bilər;

4) **extern** – sinfinin obyekti qlobaldır, yəni, proqramın bütün modullarında istifadə edilə bilər.

İdentifikator obyekti ilə əlaqəli olan fəaliyyət oblastı-identifikatorun onunla əlaqəli olan obyektə keçid üçün istifadə edilə bilən proqram hissəsidir. Fəaliyyət oblastı, obyektlərin təyin edildiyi və identifikatorların təsvir edildiyi yerdən asılıdır. Onlar blokda, funksiyada, funksiya prototipində faylda (modulda) və sinifdə verilə bilər. Əgər identifikator blokda təsvir edilibsə, onda fəaliyyət oblastı-təsvir nöqtəsindən blokun sonunadək təyin edilir. Blok funksiyanın gövdəsi olduğu halda, burada təsvir olunmuş obyektlərlə yanaşı, funksiya başlığında verilmiş formal parametrlər də təyin edilir. Beləliklə, funksiyanın təyinində formal parametrlərin fəaliyyət oblastı funksiya gövdəsidir. Məsələn, aşağıdakı funksiya öz arqumentinin qiymətinin faktorialını hesablayır.

```
long fact(int z)
{long m=1;if (z<0)return 0;
for(int i=1;i<z;m=++i*m);
return m;
}
```

Funksiya prototipi, formal parametrlər siyahısında verilmiş identifikatorların fəaliyyət oblastıdır. Bu fəaliyyət oblastının sonu funksiya prototipinin sonu ilə üst-üstə düşür. *Məsələn*,

```
//k1.cpp
#include<iostream.h>
long fact(int z)
{long m=1;if(z<0)return 0;
```

```
for(int i=1;i<z;m=++i*m);return m;
}
main()
{int j=1,k=3;
long fact(int k=0);for(;j<=k;j++)
cout<<"\n arg="<<j<<
                "arg!="<<fact(j);
}
```

Nəticədə alırıq: **arg=1 arg!=1**
arg=2 arg!=2
arg=3 arg!=6

Proqram (modul) mətni verilmiş fayl bütün qlobal, yəni ixtiyari funksiya və siniflərdən kənar təsvir edilmiş obyekt adları üçün fəaliyyət oblastıdır. Hər bir qlobal ad təsvir nöqtəsindən faylın sonuna qədər təyin edilir.

Obyektin görünüşü anlayışı, bir-birinin daxilində verilən bloklarda (və ya funksiyalarda) identifikatorun təkrar təyininin mümkünlüyü ilə əlaqədar daxil edilir. Bu halda adla obyekt arasındakı başlanğıc əlaqə pozulur. *Məsələn,*

```
//K2.CPP
#include<iostream.h>
void main()
{char cc[]="Num";
float pi=3.1415926;
cout<<"\n pi="<<pi;
{double pi=3.1415926535897932385;
cout<<'\n'<<cc"double pi="<<pi;
}
cout<<'\n'<<cc"float pi="<<pi;
}
```

Nəticədə alırıq: **pi=3.1415926**
Num double pi=3.1415926535897932385
Num float pi=3.1415926

Obyektlərin mövcudluğunun davamiyyəti, proqramdakı identifikatorlara yaddaşdakı konkret obyektlərin uyğun gəldiyi perioddur. Əlaqələndirmə tipi isə başlanğıc mətni bir neçə fayllarda (modullarda) yerləşdirilən proqramdakı konkret obyekt və ya funksiya ilə identifikator arasındakı uyğunluğu təyin edir.

6.4. Təsvirlər və təyinlər. İfadələr. Tiplərin çevrilməsi

Aşağıdakı obyektləri təyin və təsvir etmək olar:

1) dəyişənləri;

2) funksiyaları;

3) sinifləri, onların elementlərini və bu elementlərə göstəriciləri;

4) typedef ilə istifadəçinin daxil etdiyi tipləri;

5) struktur, birləşmə və sadalamaların tipləri və adlarını;

6) struktur və birləşmələrin elementlərini;

7) verilmiş tip obyektlərin massivlərini;

8) sadalanan sabitləri;

9) operator nişanlarını;

10) preprosessorun makroslarını;

11) verilmiş tip və ya obyektlərə göstəriciləri;

12) verilmiş tip funksiya və ya obyektlərə istinadları;

13) verilmiş tip sabitləri.

Verilmiş tip dəyişənlərin təyini aşağıdakı formata malikdir:

s m tip ad1qiymət1, ad2 qiymət2,...;

burada **s-auto**, **static**, **extern**, **register**, və ya **typedef** yaddaş sinfi göstəricilərindən biridir, **m-const** və ya **volatile** modifikatorlarından biridir, tip dildə təyin olunmuş əsas tiplərdən biridir, ad-identifikatordur, qiymət isə uyğun obyektin başlanğıc qiymətini təyin edir. Qiymət aşağıdakı formatda mənimsədilir:

= başlanğıc qiymət

və ya (başlanğıc qiymət)

Adətən başlanğıc qiymət kimi sabitlərdən istifadə olunur. Mötərizəli formadan yalnız funksiya daxilində istifadə edilməsinə icazə verilir. Qeyd edək ki, təsvirlər yalnız aşağıdakı hallarda təyin hesab edilir:

1) dəyişəni təsvir edirsə;

2) başlanğıc qiymət mənimsədirsə;

3) funksiyanı tam təsvir edirsə (funksiyanın gövdəsi də daxil olmaqla);

4) birləşmə və ya strukturu təsvir edirsə (elementləri də daxil olmaqla);

5) sinfi təsvir edirsə (elementləri də daxil olmaqla).

Təsvir aşağıdakı hallarda təyin etmə hesab edilə bilməz:

1) funksiyanın prototipini təsvir edirsə;

2) **extern** yaddaş sinifi varsa;

3) sinfin statik elementi təsvir olunursa;

4) sinfin adı təsvir edilirsə;

5) **typedef** sözü ilə istifadəçinin daxil etdiyi tip təsvir edilirsə.

Təsvirlərə aşağıdakı misallar göstərmək olar: **extern int g**; (xaricdə verilən dəyişən), **float f (int, double)**; (funksiya prototipi); **struct st**; (struktur adı).

Təyin etmələrə aşağıdakı misallar göstərmək olar:**char sm**; (dəyişən təyini), **float d=10.0**; **double Euler (2.718282)**; (başlanğıc qiyməti ilə təyin edilən dəyişən), **const float pi=3.14159**; (sabit təyini), **float x2 (float x) {return x*x;}**; (funksiya təyini), **struct {char a; int b;} st**; (struktur təyini), **enum {zero,one,two}**; (sadalanmanın təyini).

Const və **volatile** modifikatorları təyin edilən obyektlərin sabit və ya dəyişən olub-olmaması haqqında məlumat verir. Əgər dəyişən **const** ilə təsvir edilibsə, o proqramın digər modullarında istifadə edilə bilməz və onun qiyməti proqramın yerinə yetirildiyi müddət ərzində dəyişdirilə bilməz. Ona qiymət mənimsətmək üçün yeganə imkan, onun təyin edilməsi zamanı olur. **Volatile** modifikatoru, obyektin qiymətini, ona hər dəfə müraciət zamanı dəyişdirmək mümkün olduğunu bildirir.

İfadə, hesablama müəyyən edən operandlar, ayırıcılar və əməliyyat işarələri ardıcıllığıdır. Əməliyyatların operandlara tətbiq edilməsi qaydası, əməliyyatların yerinə yetirilmə üstünlüyü və assosiativliyi ilə müəyyən edilir. Əməliyyatların yerinə yetirilmə ardıcıllığını dəyişdirmək üçün yumru mötərizələrdən istifadə olunur. Ümumi halda unar əməliyyatlar, şərti əməliyyat və mənimsətmə əməliyyatı sağdan sola doğru, yerdə qalan əməliyyatlar isə soldan sağa doğru yerinə yetirilir. Belə ki, **x=y=z** əməliyyatı **x=(y=z)**, **x+y-z** əməliyyatı isə **(x+y)-z** deməkdir.

İxtiyari ifadənin əsas başlanğıc elementi başlanğıc ifadədir. Onlara aşağıdakılar aiddir:

1) sabit;
2) ad;
3) (ifadə);
4) :: identifikator;
5) :: funksiya-əməliyyat adı;
6) this.

Öz növbəsində sabitlərə aiddir:

1) tam sabit;
2) simvol sabit;
3) sadalanan sabit;
4) həqiqi sabit;
5) sətir sabit (sətir).

Adlara aiddir:

1) identifikator-yalnız uyğun təyin etmə vasitəsilə daxil edildikdə ad kimi istifadə edilə bilər, məsələn, identifikator-dəyişən adı;

2) funksiya-əməliyyat adı-əməliyyatların işinin genişləndirilməsi ilə əlaqədar daxil edilə bilər (sinif anlayışında);

3) sinif adı-sinfin elementinə müraciəti təmin edir. Beləliklə, "sinif" anlayışı daxil etmədən, ad kimi yalnız identifikatordan istifadə etmək olar. Burada (ifadə)-rekursiyaya malik, yumru mötərizələrə alınmış ixtiyari ifadədir.

Funksiyaya müraciətin ümumi şəkli aşağıdakı kimidir:

FM (ifadələr siyahısı), burada FM-funksiya adı və ya funksiyaya göstərici və ya funksiyaya istinaddır. İfadələr siyahısı faktiki parametrlərin siyahısıdır.

Tiplərin aşkar çevrilməsi üçün **type** (ifadələr siyahısı) ifadəsindən istifadə olunur. Əgər burada siyahıda birdən artıq ifadə varsa, tip sinif olmalı, ifadə isə sinif konstruktoru olmalıdır. Siyahıda ifadə yeganədirsə, **type**-isə sadə tip adıdırsa, onda bu cür aşkar tip çevirməsinin ümumi forması: sadə tip adı (ifadə) şəklində olur. Məsələn, **int(3)**, **float(3/7)**, **int('B')**. Bu yazılış sadə adı olmayan tiplərə şamil edilə bilməz. Məsələn, **unsigned long(x/3+7)** və ya **char*(0777)** yazılışları mümkün deyil. Qeyd edək ki, tiplərin aşkar çevrilməsi üçün tətbiq olunan bu funksional yazılışla yanaşı kanonik əməliyyatdan da istifadə edilə bilər. Məsələn, bu əməliyyatla yazmaq olar: **(int)3**, **(float)3/7**, **(int)'A'**.

Kanonik əməliyyat mürəkkəb işarələnməyə malik tiplərə də tətbiq oluna bilər. Məsələn, **(unsigned long) (x/3+7)** və ya **(char*) 0777**.

Mürəkkəb adlı tiplərin aşkar çevrilməsi üçün **typedef**-dən istifadə etmək olur. Məsələn, **typedef unsigned long int x; typedef char *p;**

İfadələrin hesablanması zamanı bəzi əməliyyatlar, operandların uyğun tipdən olmasını tələb edir, tip üzərinə qoyulan şərt ödənilmirsə, lazımi tip çevirmələri məcburi aparılır. C++ dilində tip çevrilmələri arasında aşağıdakıları qeyd etmək olar:

1) hesabi ifadələrdə operandların çevrilməsi;

2) göstəricilərin çevrilməsi;

3) istinadların çevrilməsi;

4) sinif elementlərinə göstəricilərin çevrilməsi.

Hesabi ifadələrdə operandların çevrilməsi mərhələlərini qeyd edək:

1) Bütün qısa tam tiplər ondan qısa olmayan digər tam tiplərə çevrilir. Sonra əməliyyatda iştirak edən hər iki qiymət **int** və ya **float** və ya **double** tipini alır.

2) Əgər operandlardan biri **long double** tiplidirsə, onda ikinci də **long double** tipinə gətirilir.

3) Əgər operandlardan biri **double** tiplidirsə, onda digər operand da **double** tipinə gətirilir.

4) Operandlardan biri **float** tiplidirsə, digəri də **float** tipinə gətirilir.

5) Əgər operandlardan biri **long int**, digəri **unsigned int** tiplidirsə, əgər **long int** tipi bütün **unsigned int** tipli qiymətləri ifadə edə bilirsə, axırıncı **long int** tipinə gətirilir, əks halda hər iki operand **unsigned long** int tipinə çevrilir.

6) Operandlardan biri **long** tiplidirsə, digər operand da **long** tipinə gətirilir.

7) Operandlardan biri **unsigned** tiplidirsə, digəri də **unsigned** tipinə gətirilir.

8) Nəhayət 2)-7) bəndləri ödənilmirsə hər iki operand **int** tipinə gətirilir.

Standart hesabi çevrilmələrin qaydaları

Başlanğıc tip	char	unsigned char	signed char	short	unsigned short	enum
Çevrildiyi tip	int	int	int	int	unsigned int	int

6.5. Proqramın strukturu. Standart funksiyalar. Giriş-çıxış prosedurları

C++ dilində proqramın ümumi strukturu aşağıdakı kimidir:

preprosessorun direktivləri
funksiya 1-in təyini
funksiya 2-nin təyini
.....
funksiya n-nin təyini.

Funksiyalar arasında hökmən main adlı baş funksiya iştirak edir. Sadə proqram yalnız baş funksiyadan ibarət olub, aşağıdakı struktura malikdir:

preprosessorun direktivləri
void main ( )
{obyektlərin təyini;
yerinə yetirilən operatorlar;
}

Misal. a, b, c tərəfləri ilə verilmiş üçbucağın sahəsini Heron düsturu ilə hesablayaq : $S = \sqrt{p(p-a)(p-b)(p-c)}$,
$p=(a+b+c)/2$.

```
#include<stdio.h>
#include<math.h>
void man()
{float a,b,c,p,s;
printf("\na=");scanf("%f",&a);
printf("\nb=");scanf("%f",&b);
printf("\nc=");scanf("%f",&c);
p=(a+b+c)/2;s=sqrt(p*(p-a)*(p-b)*(p-c));
printf("\ns=%f",s);
}
```

Riyazi standart funksiyalar – math.h faylı

Funksiya	*Təyinatı*
`int abs (int i);`	i tam arqumentinin mütləq qiymətini qaytarır.
`double acos (double x);`	Arkkosinusun funksiyası, burada arqument -1-dən +1 diapazonunda olur.
`double asin (double x);`	Arksinusun funksiyası, burada arqument -1-dən +1 diapazonunda olur.
`double atan (double x);`	Arktangens funksiyası.
`double atan 2 (double y, double x);`	y/x qiyməti üçün arktangens funksiyası.
`double cos (double x);`	Kosinus funksiyası. Bucaq (arqument) radianlarla verilir.
`double cosh (double x);`	x-ın hiperbolik kosinusunun qiymətini qaytarır.
`double exp (double x);`	e^x qiymətini hesablayır.
`double fabs (double x);`	İkiqat dəqiqlikli x arqumentinin mütləq qiymətini qaytarır.
`double floor (double x);`	x qiymətini aşmayan ən böyük tam ədədi tapır.
`double fmod (double x, double y);`	x-ın y-ə tam bölünməsindən qalan qalıq hədddin tapır.
`double hypot (double x, double y);`	x,y katetlərinin qiymətləri üzrə düzbucaqlı üçbucağın z hipotenuzunu hesablayır ($z^2=x^2+y^2$).
`long labs(long x);`	x tam arqumentinin mütləq qiymətini qaytarır.
`double ldexp (double v,int e);`	v 2^e ifadəsinin qiymətini qaytarır.

double log (double x);	ln x natural loqarifminin qiymətini qaytarır.
double log 10 (double x);	$\log_{10} x$ onluq loqarifminin qiymətini qaytarır.
double pow (double x,double y);	x^y-in qiymətini qaytarır.
double pow 10 (int p);	10^p-in qiymətini qaytarır.
double sin (double x);	Sinus funksiyası. Arqument radianlarla verilir.
double sinh (double x);	x üçün hiperbolik sinusun qiymətini qaytarır.
double sqrt (double x);	$\sqrt{x}$ kvadrat kökünün müsbət qiymətini qaytarır.
double tan (double x);	Tangens funksiyası. Arqument radianlarla verilir.
double tanh (double x);	x üçün hiperbolik tangensin qiymətini qaytarır.

Ekrana formatlı çıxarış operatoru **printf** ( ) aşağıdakı struktura malikdir:

printf (format sətri, arqumentlər siyahısı) ;

Burada format sətiri ikiqat dırnaq işarəsi ilə məhdudlaşdırılıb, yəni mətni sabitdir. Bu mətndə ixtiyari mətn və formatı idarə edən simvollar verilə bilər. Arqumentlər siyahısı boş da ola bilər, yaxud ekrana çıxarılan ifadələrdən (xüsusi halda sabit və dəyişənlərdən) ibarət ola bilər. Sətirdəki mətn daxil edildiyi kimi çıxarılır. İdarəedici simvollar ekrana çıxarılan işarələrin yerləşdirilmə qaydasına təsir edir. İdarəedici simvolun əlaməti "\" işarəsidir. Əvvəl qeyd etdiyimiz kimi, \n-yeni sətrə keçidi, \t-üfiqi tabulyasiyanı, \r-kursorun yeni sətrin əvvəlinə qaytarılmasını, \a-siqnal-zəngi, \b-bir mövqe geriyə qayıdışı, \f-səhifənin çevrilməsini, \v-isə şaquli tabulyasiyanı müəyyən edir.

Format çıxarılan kəmiyyətin xarici ifadə formasını təyin edir. Onlardan bəzilərini qeyd edək:**%c**-simvol, **%s**-sətir, **%d**-onluq tam ədəd, (**int** tipi), **%u**-işarəsiz onluq tam ədəd(**unsigned** tipi), **%f**-qeyd olunmuş onluq nöqtə formalı həqiqi ədəd, **%e**-

sürüşən onluq nöqtəli həqiqi ədəd. Məsələn, aşağıdakı operatorların yerinə yetirilməsi

```
float m,p;int k;
m=84.3;k=-73;p=43.57;
printf("\nm=%f\tk=%d\tp=%e",m,k,p);
```

aşağıdakı nəticələri verir:

```
m=84.299999 k=-73 p=4.35700e+01
```

Formata sahənin eni və dəqiqlik kimi ədədi parametrlər əlavə edilə bilər. Sahə eni kəmiyyət üçün ekranda ayrılan mövqelərin ümumi sayını, dəqiqlik isə onluq nöqtədən sonra kəsr hissə üçün ayrılan mövqelərin sayını təyin edir. Bu parametrlər % işarəsi ilə format simvolu arasında yerləşdirilir və bir-birindən nöqtə işarəsi ilə ayrılır. *Məsələn,*

```
printf("\nm=%5.2f\tk=%5d\tp=%8.2e\tp=%11.4e",
m,k,p,p);
```

Nəticədə ekranda alarıq:

```
m=84.30   k=-73    p=43.57    p=4.3570e+01
```

Əgər göstərilən sahə eni hüdudlarına çıxarılan qiymət yerləşmirsə, onda bu parametr nəzərə alınmır və kəmiyyət olduğu kimi tam şəkildə çıxarılır. Formata aşağıdakı variantlı modifikatorlar əlavə edilə bilər: **%ld**–long int tipinin çıxarışı, **%hu** – **short unsigned** tipinin çıxarışı, **%Lf** – **long double** tipinin çıxarışı.

Klaviaturadan formatlı daxiletmə operatoru **scanf( )** aşağıdakı struktura malikdir:

scanf (format sətri, arqumentlər siyahısı) ;

Bu funksiya klaviaturadan daxil edilən simvolların oxunmasını və onların tipinə uyğun daxili ifadə formasına çevrilməsini təmin edir. Burada scanf ( ) funksiyasında format sətri və arqumentlər siyahısı hökmən iştirak edir. Klaviaturadan daxil edilən və **scanf** ( ) funksiyası tərəfindən qəbul edilən simvollar ardıcıllığını giriş axını adlandırırlar. Burada **scanf** ( ) funksiyası həmin axını ayrı-ayrı daxil edilən kəmiyyətlərə ayırır, göstərilən tip və formata uyğunlaşdırır, arqumentlər siyahısında göstərilmiş dəyişənlərə mənimsədir. Arqumentlər siyahısı daxil edilən dəyişənlərin ardıcıllığını verir, hər bir dəyişən qarşısında **&** işarəsi qoyulur. Bu dəyişənin ünvanının qəbul edilməsi əməliyyatının işarəsidir. Format sətiri dırnaq işarəsi arasına alınır (**printf**-də olduğu kimi) və formatdan ibarət olur. Hər bir format qarşısında **%** işarəsi olur, bu işarədən sonra isə * sahə eni və modifikator verilə

bilər. Onlar arasında yalnız format olması məcburidir. Burada aşağıdakı formatlardan istifadə edilə bilər:

1) **d**-onluq tam ədədlər üçün (int tipi);

2) **u**-işarəsiz onluq tam ədədlər üçün (**unsigned int** tipi);

3) **f**-qeyd olunmuş onluq nöqtəli həqiqi ədədlər üçün (**float** tipi);

4) **e**-sürüşən onluq nöqtəli həqiqi ədədlər üçün (**float** tipi).

Burada * işarəsi giriş axınında müəyyən sayda simvolları buraxmağa imkan verir. Sahə eni giriş axınındakı daxil edilən dəyişənə uyğun simvolların sayını təyin etməyə imkan verən müsbət tam ədəddir. Burada **printf( )** funksiyasında olduğu kimi **h,l,L** modifikatorlarından istifadə etmək olar. Onlar aşağıdakı qaydada istifadə olunur:

1) **hol** – **short** int tipli qiymətlərin daxil edilməsi üçün;

2) **ld** – **long int** tipli qiymətlərin daxil edilməsi üçün;

3) **lf,le** – qeyd olunmuş və sürüşən onluq nöqtəli **double** tipli qiymətlərin daxil edilməsi üçün;

4) **Lf,Le** – qeyd olunmuş və sürüşən onluq nöqtəli **long double** tipli qiymətlərin daxil edilməsi üçün.

Giriş axınında müxtəlif qiymətlər arasında ayırıcı kimi ixtiyari sayda probel qoyula bilər. Burada yalnız *Enter* düyməsi sıxıldıqdan sonra daxil edilən qiymətlər uyğun dəyişənlərə mənimsədilir. Buna qədər isə giriş axını redaktə edilə bilər.

C++ dilində proqramlaşdırma zamanı **stdio.h** faylı ilə daxil edilən C dilinin standart kitabxanasının giriş-çıxış vasitələrindən istifadə etməklə yanaşı C++ dilinin öz giriş-çıxış vasitələrindən də istifadə etmək olar. Bu **iostream.h** faylı ilə daxil edilən siniflər ki-tabxanasıdır. Bu kitabxanada obyekt kimi aşağıdakı standart simvol axınları müəyyən edilib:

1) **cin** – klaviaturadan giriş üçün standart axın;

2) **cout** – ekrana çıxış üçün standart axın.

Verilənlərin daxil edilməsi **cin** axınından verilənlərin çıxarılması və uyğun dəyişənlərə mənimsədilməsi kimi ifadə olunur. C++ dilində standart axından çıxarış əməliyyatı müəyyən olunub və bu əməliyyat **>>** işarəsi ilə ifadə olunur. Məsələn, **x** dəyişəninə qiymətlərin daxil edilməsi, **cin >> x;** operatoru ilə yerinə

yetirilir.

Verilənlərin çıxarılması isə cout standart axınına çıxarılan qiymətlərin yerləşdirilməsi kimi ifadə olunur. Burada ikiqat dırnaq işarəsi arasında mətnlər və ifadə qiymətləri çıxarıla bilər. Axına yerləşdirilmə əməliyyatı << işarəsi ilə göstərilir.

Məsələn,

```
cout<<a+b;
cout<<"\n cavab ="<<y;
cout<<"x="<<x<<"y="<<y<<"z="<<z;
```

Yuxarıda baxdığımız misalı C++ dilinin giriş-çıxış axınları vasitəsilə aşağıdakı qaydada yaza bilərik.

```
#include<iostream.h>
#include<math.h>
void main()
{float a,b,c,p,s;
cout<<"\na=";cin>>a;
cout<<"\nb=";cin>>b;
cout<<"\nc=";cin>>c;
p=(a+b+c)/2;s=sqrt(p*(p-a)*(p-b)*p-c));
cout<<"\ns="<<s;
}
```

6.6. C++ dilinin operatorları

C++ dilinin hər bir operatoru nöqtə vergül işarəsi ilə sona çatır. Sonunda nöqtə vergül işarəsi qoyulan ixtiyari ifadə operator kimi qəbul edilir. Çox vaxt operator-ifadə heç bir qiymət qaytarmayan funksiyaların çağırılması üçün istifadə olunur. Eləcədə operator-ifadə funksiya çağırışı ilə yanaşı, mənimsəmə ifadəsi ola bilir. Mənimsəmələr əməliyyatlara aid olduğundan və bunlar ifadələrin formalaşdırılmasında istifadə olunduğundan C++ dilində ayrıca mənimsəmə operatoru yoxdur. Burada mənimsəmə operatoru operator-ifadənin xüsusi halıdır.

Operatorun xüsusi halı boş operatordur. Bu operator boş sətirin sonunda nöqtə vergül işarəsinin qoyulması ilə alınır. Boş operator heç bir əməliyyat yerinə yetirmir.

Hər bir operatorun qarşısında, ondan iki nöqtə işarəsi ilə ayrılan nişan qoyula bilər. Nişan kimi istifadəçi tərəfindən seçilən

ixtiyari simvollar ardıcıllığından istifadə olunur. Məsələn, `5:y=7; ABC:x=4*x;` və s.

Boş operatorun da qarşısında nişan qoymaq mümkün olduğundan, boş operatorların köməyi ilə proqramın ixtiyari yerində nişanlar qoymaq olar.

Fiqurlu mötərizələr daxilində verilən operatorlar ardıcıllığına qurma operator deyilir. Əgər fiqurlu mötərizələr daxilində verilən operatorlar ardıcıllığı arasında təyin etmələr və təsvirlər varsa, onda qurma operator bloka çevrilir. Burada blok da, qurma operator da müstəqil operator rolunu oynayır. Lakin blokun, qurma operatorun sonunda nöqtə vergül işarəsi qoyulmur, onlar fiqurlu mötərizələrlə məhdudlanır. Blokun və qurma operatorun daxilindəki bütün operatorların sonunda nöqtə vergül işarəsi qoyulur. *Məsələn,*

`{int a; char b='0'; a=(int)b; }` (blok),
`{func(z+1.0,22); e=4*x-1;}` (qurma operator).

Seçki operatorları. Bu operatorlara şərt operatoru (**if...else**) və variant (**switch**) operatoru aiddir. Onlardan hər biri proqramın yerinə yetirilmə yolunun seçilməsi üçün nəzərdə tutulub.

Şərt operatorunun ümumi şəkli aşağıdakı kimidir:

if (ifadə) oprator1 **else** operator2;

Burada ifadə skalyar olub, hesabi tipli və ya göstərici tipli ola bilər. Əgər ifadə sıfırdan fərqlidirsə və ya boş olmayan göstəricidirsə, onda ifadə doğru hesab olunur və operator1 yerinə yetirilir. Əks halda operator2 yerinə yetirilir. Burada operatorlar kimi təsvirlərdən və təyin etmələrdən istifadə etmək olmaz. Lakin burada qurma operatorlardan və bloklardan istifadə edilə bilər. *Məsələn,*

```
if(x>0){x=-x;f(x*2);}
else{int i=2;x*=i;f(x);}
```

Şərt operatorunun aşağıdakı qısa formasından da istifadə olunur:

if (ifadə) operator1;

Bu halda ifadə sıfra bərabər olduqda, yəni yoxlanılan şərt yalan olarsa, heç bir əməliyyat yerinə yetirilmir. *Məsələn,*

```
if(a<0)a=-a;
```

Burada operator1 və operator2 də öz növbəsində digər şərt

operatoru da ola bilər, nəticədə şərtlərin yoxlanması zəncirini qurmaq olar. Bu zəncirdə həm tam formalı, həm də qısa formalı şərt operatorları iştirak edə bilər. *Məsələn*,

```
int max3(int x,int y,int z)
{if(x<y)
if(y<z)return z;
else return y;
else
if(x<z)return z;
else return x;
}
```

Variant operatorunun ümumi şəkli aşağıdakı kimidir:

switch (keçirici ifadə)
{**case** sabit ifadə1: operatorlar1;
case sabit ifadə2: operatorlar2;
...
case sabit ifadə n: opratorlar n;
default:operatorlar;
}

Bu operator yerinə yetirilərkən əvvəlcə keçirici ifadə hesablanır, sonra isə **switch** idarəedici sözü idarəetməni **case** sözü ilə qeyd olunmuş və sabit ifadəsinin qiyməti **switch** sözündəki keçirici ifadənin qiyməti ilə üst-üstə düşən operatorlara verir. Burada keçirici ifadə həmişə ya tam sabit ya da tam qiymət alan hesabi ifadə olmalıdır. Eləcə də **case** xidməti sözündən sonra gələn sabit ifadələrin tipləri **switch** sözündəki keçirici ifadənin tipinə gətirilməlidir. Bütün sabit ifadələr müxtəlif qiymətlər almalıdır, lakin eyni tipli olmalıdırlar. Burada **switch**(...) konstruksiyasından sonra fiqurlu mötərizələrdə yerləşdirilmiş ixtiyari operator aşağıdakı formalı bir və ya bir neçə nişanlarla qeyd oluna bilər:

case sabit ifadə :

Əgər keçirici ifadənin aldığı qiymət, operatordakı heç bir sabit ifadənin qiyməti ilə üst-üstə düşmürsə, onda idarəetmə **default** sözü ilə qeyd olunmuş operatorlara verilir. Operatorda yalnız bir **default** sözü ola bilər, lakin bu söz operatorda verilməyə də bilər. Operatorlarda **default** sözü olmadıqda, keçirici ifadənin qiyməti heç bir sabit ifadənin qiyməti ilə üst-üstə

düşmədikdə, burada heç bir operator yerinə yetirilmir. Burada **case** sabit ifadə **j:** və **default:** nişanları operatorların yerinə yetirilmə ardıcıllığını dəyişdirmir. Əgər operatorda keçid və çıxış nəzərdə tutulmayıbsa, onda burada idarəetmənin verildiyi nişandan başlayaraq ardıcıl olaraq bütün operatorlar yerinə yetiriləcəkdir.

Misal.

```
\\p4.cpp.
#include<iostream.h>
void main( )
{int ic;cin>>ic;cout<<'\n';
switch(ic)
{case0:case1:cout<<"one,";
case2:case3:cout<<"three,";
case4:case5:cout<<"five,";
case6:case7:cout<<"seven,";
case8:case9:cout<<"nine.";
                        break;
            default:cout<<"Error!";
}
}
```

Burada əgər **ic**-nın qiyməti 4 olarsa, cavab **five**, **seven**, **nine**, **ic**-ya **x** qiyməti verilsə, cavab **Error**! olacaqdır. Qeyd edək ki, burada **break** operatoru variant operatorlarından çıxışı təmin edir. Məsələn, proqramda hər bir rəqəmin çıxışında **break** operatoru verilsə idi, onda proqram yalnız bir tək rəqəm adı çap edəcəkdir.

Dövr operatorları. C++ dilində aşağıdakı tip dövr operatorları təyin edilib:

1) şərt qabaqcadan yoxlanılan dövr operatoru:

while (ifadə–şərt)

dövrün gövdəsi

2) şərt sonradan yoxlanılan dövr operatoru:

do

dövrün gövdəsi

while(ifadə–şərt);

3) iterasiyalı dövr operatoru:

for (dövrün başlanğıc qiymətləri;
ifadə-şərt; ifadələr siyahısı)
dövrün gövdəsi.

Burada dövrün gövdəsi təsvir və ya təyinetmə ola bilməz. Dövrün gövdəsi ya bir operatordan (o cümlədən boş operatordan), ya qurma operatordan, ya da blokdan (fiqurlu mötərizə daxilində) ibarət olur. Şərt-ifadə dövrdə aparılan iterasiyaların yerinə yetirilməsi şərtini təyin edən skalyar (adətən münasibət və ya hesabi ifadə) ifadədir. İterasiyalı dövr operatorunda dövrün başlanğıc qiymətlərinin sonunda nöqtə vergül işarəsi qoyulur, yəni bu qiymətlər, ondan sonra gələn şərt-ifadədən bu işarə ilə ayrılır. Başlanğıc qiymətlər və şərt-ifadə olmadıqda belə uyğun nöqtə vergül işarələri qoyulmalıdır. Nəhayət, ifadələr siyahısı bir-birindən vergüllə ayrılan skalyar ifadələr ardıcıllığıdır.

Dövrün yerinə yetirilməsi aşağıdakı hallarda dayandırıla bilər:

1) yoxlanılan şərt-ifadə sıfır qiymətini alsın;

2) dövr daxilində (**break**, **goto**, **return**) keçid operatorları idarəetməni dövrdən kənara versin.

Şərt qabaqcadan yoxlanılan dövr operatoru yerinə yetirilərkən əvvəlcə şərt-ifadə yoxlanılır, əgər onun qiyməti sıfırdan fərqlidirsə, onda dövrün gövdəsi yerinə yetirilir. Sonra şərt-ifadənin yoxlanılması və dövrün gövdəsinin yerinə yetirilməsi şərt-ifadə sıfır qiymətini alana qədər təkrarlanır. Yoxlanılan şərt-ifadə kimi çox vaxt münasibət əməliyyatlarından istifadə olunur. Məsələn, birinci k sayda natural ədədin kvadratları cəminin tapılması üçün yaza bilərik:

```
int i=0;int s=0;
while(i<k)
s+=++i*i;
```

Şərt qabaqcadan yoxlanılan dövr operatorundan istifadə edərkən, şərt-ifadənin dəyişdirilməsi üçün dövrün gövdəsini təşkil edən operatorlar şərtə təsir göstərməlidirlər.

Şərt sonradan yoxlanılan dövr operatoru yerinə yetirilərkən əvvəlcə dövrün gövdəsini təşkil edən operatorlar yerinə yetirilir. Sonra şərt-ifadə yoxlanılır, əgər şərt-ifadənin aldığı qiymət sıfırdan fərqlidirsə, onda dövrün gövdəsi yenidən yerinə yetirilir. *Məsələn,*

```
int i=0;int s=0;
do
s+=++i*i;
while(i<k);
```

İterasiyalı dövr operatorunda dövrün başlanğıc qiymətləri, bir-birindən vergüllə ayrılan təsvirlər (təyin etmələr) və ifadələr ardıcıllığından ibarətdir. Dövrün başlanğıc qiymətlərinə aid olan ifadələr dövrə giriş zamanı yalnız bir dəfə hesablanır. Adətən burada sayğacların və dövr parametrlərinin başlanğıc qiymətləri verilir. Şərt-ifadə isə əvvəlki dövr operatorlarında olduğu kimidir, yəni onun qiyməti sıfıra bərabər olduqda dövrün yerinə yetirilməsi dayandırılır. Bu operatorda şərt-ifadə verilmədikdə, onun sonunda qoyulan nöqtə vergül işarəsi şərt-ifadənin verilməməsinə baxmayaraq saxlanılır. Şərt-ifadənin verilməsi, burada onun həmişə doğru qiymət aldığını, yəni ödənildiyini bildirir. Qeyd edək ki, başlanğıc qiymətlər operatorda verilmədikdə onların sonunda gələn nöqtə vergül işarəsi saxlanılır. İfadələr siyahısında verilən ifadələr, dövrün gövdəsini təşkil edən operatorlar yerinə yetirildikdən sonra şərt-ifadənin növbəti dəfə yoxlanılmasına qədər hər iterasiyada hesablanırlar. Dövrün gövdəsi blok, ayrıca operator, qurma operator və boş operator ola bilər.

Birinci k sayda natural ədədlərin kvadratları cəmini tapmaq üçün iterasiyalı dövr operatorundan aşağıdakı kimi istifadə etmək olar:

```
for(int i=1,s=0;i<=k;i++)s+=i*i;
for(int i=0,s=0;i<=k;s+=++i*i);
for(int i=0,s=0;i<=k;)s+=++i*i;
for(int i=0,s=0;i<=k;)
{int j;j=++i;s+=j*j;}
```

Burada ikinci variantda dövrün gövdəsi boş operatordur, üçüncü variantda isə ifadələr siyahısı verilməyibdir.

Beləliklə, iterasiyalı dövr operatoru yerinə yetirilərkən, əvvəlcə dövrün obyektlərinə onların başlanğıc qiymətləri mənimsədilir, sonra şərt-ifadə yoxlanılır, əgər şərt-ifadə sıfırdan fərqli qiymət alırsa, dövrün gövdəsini təşkil edən operatorlar hesablanır. Daha sonra ifadələr siyahısında verilmiş ifadələr hesablanır. Bundan sonra şərt-ifadə yenidən yoxlanılır və yuxarıda təsvir etdiyimiz proses təkrar edilir.

Qeyd edək ki, burada şərt-ifadə onun hesablanması nəticəsində, yaxud dövrün gövdəsindəki operatorların təsiri hesabına dəyişir.

Proqramda müxtəlif tip operatorlar bir-birinin daxilində yerləşdirilə bilər. Bu halda daxildə verilən for dövr operatorunun başlanğıc qiymətlərin mənimsədildiyi hissədə, xarici dövr operatorunda təyin olunmuş dəyişən adı verilə bilər. *Məsələn,*

```
#include<iostream.h>
void main(void)
{for(int i=0;i<3;i++)
        {cout<<"\n i="<< i;
cout<<",daxili dovr:";
for(int i=6;i>3;i--)cout<<"i="<<i;
cout<<".\n sonra:i="<<i<<".";
}
}
```

Tam N ədədinin faktorialının hesablanması məsələsinin həllini şərt qabaqcadan yoxlanılan dövr operatorundan istifadə etməklə (müqayisə üçün həm Turbo Pascal, həm də C++ dillərində) verək:

```
var f:longint; i,n:integer;
begin write('n=');
readln(n);f:=1;i:=1;
while i<=n do
begin f:=f*i;i:=i+1 end;
writeln(n,'!=',f)
end.

#include<iostream.h>
void main( )
{long int f; int i,n;
cout <<"n=",cin>>n;
f=i=1;
while(i<=n) f=f*i++;
cout<<"\n"<<n<<"!="<<f;
}
```

Verilmiş ε dəqiqliyi ilə 1+1/2+1/3+... harmonik sırasının cəmini tapaq:

```
var n:integer;
s,eps:real;
begin readln(eps);
n:=1;s:=0;
while (1/n>eps) and
(n<maxint) do
begin s:=s+1/n; n:=n+1
end;
writeln('sum=',s)
end.
```

```
#include<iostream.h>
#include<limits.h>
void main()
{int n=1;
double s=0,eps;
cin>>eps;
while (1.0/n>eps &&
n<int_max)
s+=1./n++;
cout<<"\nsum="<<s;
}
```

Tam N ədədinin faktorialının hesablanması məsələsini şərt sonradan yoxlanılan dövr operatorundan istifadə etməklə həlli edək:

```
var f:longint;
i,n:integer;
begin write('n=');
readln(n);f:=1;i:=1;
repeat f:=f*i;i:=i+1
until i>n;
writeln(n,'!=',f)
end.
```

```
#include<iosteram.h>
void main( )
{long int f;int i,n;
cout<<"n=";cin>>n;
f=i=1; do f*=i++;
while(i<=n);
cout<<"n\n"<<n<<"i="<<f;
}
```

Faktorialın hesablanması məsələsinin həllini iterasiyalı dövr operatorundan istifadə etməklə aşağıdakı variantlarda vermək olar:

```
f:=1;for(i=1;i<=n;i++)f=f*i;
for(f=1,i=1;i<=n;i++)f=f*i;
f=1;i=1;for(;i<=n;i++)f=f*i;
for(f=1,i=1;i<=n;f=f*i,i++);
for(f=1,i=1;i<=n;f*=i++);
```

Harmonik sıranın ε dəqiqliyi ilə hesablanması məsələsini isə aşağıdakı qaydalarla vermək olar:

```
for(n=1,s=0;1.0/n>eps && n<int_max;n++)
s+=1.0/n;
```

və ya

```
for(n=1,s=0;1.0/n>eps && n<int_max;
s+=1.0/n++);
```

Aşağıdakı proqram fraqmenti isə C++ dilində bir-birinin daxilində verilən dövrlərə misal ola bilər:

```
for (x=2;x<=9;x++)
for (y=2;y<=9;y++)
cout<<"\n"<<x<<"*"<<y<<"="<<x*y;
```

Nəticədə aşağıdakını alarıq:

2*2=4
2*3=6
.....
9*8=72
9*9=81

Keçid operatorları. Keçid operatorlarına aşağıdakılar aiddir:

1) şərtsiz keçid operatoru (**goto**);

2) funksiyadan qayıdış operatoru (**return**);

3) dövrdən və variant operatorundan çıxış operatoru (**break**);

4) dövrün növbəti iterasiyasına keçid operatoru (**continue**).

Şərtsiz keçid operatorunun ümumi şəkli aşağıdakı kimidir:

goto identifikator;

burada identifikator – şərtsiz keçid operatoru istifadə olunan funksiyada yerləşən operatorun nişanının adıdır. Şərtsiz keçid operatoru ilə funksiya daxilində idarəetmə, nişanlanmış ixtiyari operatora verilə bilər. Lakin obyektlərə başlanğıc qiymətlər mənimsəyən təsvirlər üzərindən keçid yerinə yetirmək mümkün deyil. Daxildəki tam bloklar üzərindən isə keçid yerinə yetirmək mümkündür. Ümümiyyətlə, şərtsiz keçid operatorundan istifadə edərkən aşağıdakılar nəzərə alınmalıdır:

1) blok daxilinə kənardan daxil olmaq məsləhət edilmir;

2) şərt operatorunun daxilinə, yəni **if** və **else** işçi sözlərindən sonra gələn operatorlara idarəetməni vermək olmaz;

3) variant (**switch**) operatorunun daxilinə xaricdən keçmək olmaz;

4) dövr operatorunun daxilinə idarəetməni vermək olmaz.

Funksiyadan qayıdış operatorunun ümumi şəkli aşağıdakı

kimidir:

return ifadə;

və ya

return;

Əgər bu operatorda ifadə iştirak edirsə, o, yalnız skalyar ifadə ola bilər. Məsələn, aşağıdakı funksiya öz arqumentinin kubunu hesablayır və onun qiymətini qaytarır:

```
float cube(float x){return x*x*x;}
```

Break operatoru dövrdən və variant operatorundan məcburi çıxışı təmin edir. **Break** operatoru dövr operatorunun və ya variant operatorunun yerinə yetirilməsini dayandırır və idarəetməni dövr və ya variant operatorundan sonra gələn birinci operatora verir. **Break** operatorundan dövr və variant operatorundan başqa yerdə istifadə etmək olmaz.

Dövrdə təkrarlamaların davam etdirilməsini təyin edən şərti dövrün əvvəlində (**for**, **while** dövrləri) və sonunda (**do** dövrü) yox, dövrün ortasında yoxlamaq tələb edildikdə, dövrün gövdəsində break operatorundan istifadə etmək məqsədəuyğun olur. Bu halda dövrün gövdəsi aşağıdakı struktura malik olur:

```
{operatorlar
if (şərt) break;
 operatorlar
}
```

Məsələn, əgər tam **i**, **j** dəyişənlərinə elə başlanğıc qiymətlər verilərsə ki, **i<j** şərti ödənsin, onda aşağıdakı dövr onların ədədi ortasından kiçik olmayan, ən kiçik tam ədədi təyin edir:

```
while(i<j)
{i++;
if(i==j)break;
j--;
}
```

Variant və dövr operatorlarında **break** çıxışından istifadəyə misal olaraq, simvol massivində sıfır və vahidlərin sayını hesablayan proqrama baxaq:

```
#include<iostream.h>
void main(void)
{char c[]="ABC100111";
```

```
int k0=0,k1=0;
for(int i=0;c[i]!='\0';i++)
switch(c[i])
{case'0':k0++;break;
case'1':k1++;break;
default:break;
}
cout<<"\n Burada "<<k0<<"sıfır,"<<k1<<"vahid
var";
}
```

Nəticədə alarıq: Burada 2 sıfır, 4 vahid var.

Continue operatorundan yalnız dövr operatorlarında istifadə olunur. Bu operatorun köməyi ilə cari iterasiya başa çatdırılır və dövrün davam etdirilməsi, yəni növbəti iterasiyaya başlamaq şərtinin yoxlanılmasına keçirilir. Məsələn, birölçülü massivdəki yalnız müsbət elementlərin ədədi ortasını hesablayan proqram fraqmentinə baxaq:

```
for(s=0.0,k=0,i=0;i<n;i++)
{if(x[i]<=0.0)continue;
k++;s+=x[i];
}
if(k>0)s=s/k;
```

6.7. Obyekt ünvanları və göstəriciləri

Göstəricilər iki tip olur: göstəricilər-dəyişənlər və göstəricilər-sabitlər. Göstəricilərin qiymətləri konkret tip obyektlər üçün ayrılmış yaddaş hissələrinin ünvanlarıdır. Buna görə də göstəricilərin təsvir və təyin etmələrində həmişə ona uyğun olan tip gətirilir. Bu informasiya sonradan göstəricinin köməyi ilə yadda saxlanılan obyektə tam şəkildə keçməyə imkan verir.

Gösətiricilər iki kateqoriyaya bölünür: obyekt göstəriciləri və funksiya göstəriciləri. Göstəricilərin bu iki kateqoriyaya bölünməsi, onlardan istifadə qaydalarının və onların xassələrinin müxtəlifliyi ilə izah olunur. Məsələn, funksiya göstəricilərinə hesabi əməliyyatları tətbiq etmək olmaz, obyekt göstəricilərindən isə bəzi hesabi ifadələrdə istifadə edilə bilər.

Əvvəlcə obyekt göstəricilərinə baxaq. Sadə halda obyekt göstərici–dəyişənin təsvir və təyin etmələrinin ümumi şəkli aşağıdakı kimidir:

type * göstərici adı;

burada type-tip işarəsi, göstərici adı isə identifikatordur, * işarəsi müraciətin təyin edilməsi əməliyyatını bildirir. Məsələn, **int*i1p,*i2p,*i3p,i;** burada **i1p,i2p,i3p** tam tipli obyektlərə göstəricilərdir, **i**-isə tam tipli dəyişəndir.

Göstəricinin təyin edilməsi zamanı çox vaxt ona mənimsədilən qiymətini də vermək məqsədəuyğun olur. Bu halda təyin etmənin formatı aşağıdakı kimi ola bilər:

type* göstəricinin adı mənimsədilən qiymət;

type* göstəricini adı=mənimsədilən ifadə;

type* göstəricinin adı (mənimsədilən ifadə);

Mənimsədilən ifadə kimi sabit ifadədən istifadə olunur, onun xüsusi halı aşağıdakılar ola bilər:

1) yaddaş hissəsinin aşkar verilən ünvanı;

2) qiymətə malik göstərici;

3) obyektin ünvanını '**&**' əməliyyatı ilə təyin etməyə imkan verən ifadə. *Məsələn*, **char cc='d'; char*pc=&cc; char*p;** və s.

Göstəricinin təyini zamanı həm göstərici, həm də onun qiyməti sabit kimi elan edilə bilər. Bunun üçün const modifikatorundan istifadə olunur:

type const*const göstərici adı mənimsədilən qiymət;

Göstəricinin təyini zamanı tip kimi dilin əsas tiplərindən və törəmə tiplərdən istifadə edilə bilər. Dilin törəmə tiplərinə massivlər, funksiyalar, göstəricilər, istinadlar, sabitlər, siniflər, strukturlar, birləşmələr və nəhayət istifadəçinin təyin etdiyi tiplər aiddir. Əsas tiplər isə aşağıdakı sözlərlə müəyyən edilirlər: **char**, **int**, **float**, **long**, **double**, **short**, **unsigned**, **signed**, **void**.

Göstəricilər üzərində aşağıdakı əməliyyatlar aparıla bilər:

1) ünvan üzrə müraciət əməliyyatı (*);

2) tiplərin çevrilməsi;

3) mənimsətmə;

4) ünvanın alınması (&);

5) toplama və çıxma (additiv əməliyyatlar);

6) inkrement və ya avtomatik artırma (++);
7) dekrement və ya avtomatik azaltma;
8) müqayisə əməliyyatları.

6.8. Massivlər

Massiv – yaddaşın kəsilməz oblastını tutan eyni tipli elementlərin strukturudur. Massivlərin təsvir formatı aşağıdakı kimidir:

elementlərin tipi ad [sabit ifadə];

burada elementlərin tipi-mümkün əsas və ya törəmə tip, ad-identifikator, sabit ifadə isə massivin ölçüsünü, yəni buradakı elementlərin sayını təyin edir. Məsələn, int A[10]; təsviri ilə tam tipli 10 elementdən ibarət A adlı massiv elan edilir. Massiv elementləri indeksli adlarla işarələnirlər:

ad[indeks]

İndekslərin aşağı sərhəddi sıfıra bərabərdir. Məsələn,

A[0], A[1], A[2],..., A[9].

Turbo Pascal dilindən fərqli olaraq C++ dilində indekslər üçün ixtiyari dəyişmə diapazonu vermək olmaz. Təsvirdə verilmiş massiv ölçüsü, həmişə indeksin maksimal qiymətindən bir vahid çox olur. Əgər massivin elanında, onun elementlərinin qiymətləri mənimsədilirsə, bu zaman massivin ölçüləri aşkar verilməyə də bilər. Massiv statik və ya xaricdə yerləşmiş olduqda, onun elementlərinə qiymətlər avtomatik olaraq verilir. Bu halda massivin bütün elementlərinə sıfır qiyməti mənimsədilir.

Məsələn,

```
void f(void)
{static float F[4];long double A[10];
}
void main()
{extern int B[];
.....
f();
.....
}
int B[8];
```

Massiv elementlərinə qiymətlərinin aşkar mənimsədilməsi,

onun təyini zamanı mümkündür və iki üsulla həyata keçirilir: massivin ölçüsü kvadrat mötərizə daxilində verilməklə və massivin ölçüsü verilməmək şərti ilə. *Məsələn*,

```
char A1[]={'A','B','C','D'};
int B[6]={10,20,30,40};
char S[]="ABCD";
```

Kvadrat mötərizədə sabit ifadə verilmədikdə massivin elementlərinin başlanğıc qiymətlərinin daxil edilməsi məcburidir. Əgər massivin ölçüsü aşkar verilirsə, onda başlanğıc qiymətlərin siyahısındakı elementlərin sayı massivin ölçüsündən böyük olmamalıdır. Massivin təyin edilmədiyi, yalnız təsvir edildiyi halda başlanğıc qiymətlərin siyahısını vermək olmaz. Massivin təsvirində onun ölçüsü də verilməyə bilər. *Məsələn*,

```
extern float E[];
```

Misal. Həqiqi tipli elementlərdən ibarət massivi daxil edib, elementlərinin ədədi ortasını tapmalı.

```
#include<iostream.h>
#include<conio.h>
void main()
{const n=10;int i;double A[n];S;
clrscr();
for(i=0;i<n;i++){cout<<"A["<<i<<"]=";
cin>>A[i];}
S=0;for(i=0;i<n;i++)S=S+A[i];
S=S/n;
cout<<"\nS="<<S;
}
```

Misal. Verilmiş massivi elementlərinin artımı boyunca çeşidləmək tələb olunur.

```
#include<iostream.h>
#include<conio.h>
void main()
{int x[]={6,4,9,3,2,1,5,7,8,10};
int i,j,n,A;
clrscr();
n=sizeof(x)/sizeof(x[0]);
for(i=0;i<n-1;i++)
for(j=0;j<n-1-i;j++)
```

```
if(x[j]>x[j+1]){A=x[j];x[j]=x[j+1];
x[j+1]=A;}
for(i=0;i<n,i++)cout<<x[i]<<" ";
}
```

Çoxölçülü massivlərin təyinində ümumi halda elementlərin tipi, ölçü və elementlərin sayı haqqında məlumatlar verilməlidir:

type massiv adı**[k1][k2]... [kN];**

burada type-mümkün tip (əsas və ya qurma), massiv adı-identifikator, **N**-massivin ölçüsü, **k1,... kN**-isə hər ölçüdəki elementlərin sayıdır. Məsələn, **float A[5][10];**-hər biri on dənə həqiqi ədəddən ibarət massiv olan beş elementli massivi elan edir. Bu massivin ayrı-ayrı elementləri iki indeksli adlarla işarələnir:**A[0][0],A[0][1],...,A[4][9],int B[4][3][5];** isə hər biri tam ədədlərdən ibarət 3×5 ölçülü massivlər olan dörd elementli üçölçülü massivi elan edir. Qeyd edək ki, Turbo Pascal dilində olduğu kimi, çoxölçülü massivlərdə elementlər yaddaşda aşağıdakı qaydada yerləşir: əvvəlcə sonuncu indeks dəyişir, sonra sonuncudan əvvəl gələn indeks dəyişir və s., və bu zaman birinci indeks yalnız bir dəfə dəyişir. Çoxölçülü massivlərin təsviri zamanı, onun başlanğıc qiymətləri də mənimsədilə bilər. *Məsələn,*

```
int M[3][3]={11,12,13,
             21,22,23,
             31,32,33};
```

Misal. Pifaqor matrisi formasında vurma cədvəlinin ekrana çıxarılması və hesablanlanmasını təmin edən proqram quraq.

```
#include<stdio.h>
#include<conio.h>
void main()
{int i,j,A[10][10];
clrscr();
for(i=1;i<=9;i++)
{for(j=1;j<=9;j++)
    {A[i][j]=i*j;
     printf("%5d",A[i][j]);
     }
     printf("\n");
     }
}
```

6.9. Funksiyalar, göstəricilər, istinadlar

C++ dilində alt proqramların yalnız bir tipindən funksiyalardan istifadə olunur. Burada ümumiyyətlə alt proqram anlayışından istifadə olunmur, çünki C++ dilində funksiya əsas proqram vahidi, yerinə yetirilən minimal proqram moduludur. İxtiyari proqram hökmən **main** adlı əsas funksiyaya malikdir. Bundan əlavə proqramda əsas olmayan ixtiyari sayda funksiyalar ola bilər və onlar proqramda alt proqram rolunu oynayır. Proqramın bütün funksiya adlarına avtomatik olaraq **extern** yaddaş sinfi mənimsədilir, beləliklə proqramın hər bir funksiyası qloballaşır, yəni proqramın modullarından və ixtiyari yerindən onlara müraciət etmək olur. Modulda bunun üçün funksiya ona birinci müraciətə qədər təyin edilib, təsvir olunmalıdır. Beləliklə, C++ dilində hər bir proqram, qabaqcadan təyin və təsvir edilməsi tələb olunan funksiyalar ardıcıllığıdır. Funksiyanın təyinində ona müraciət zamanı yerinə yetiriləcək əməliyyatlar, funksiyanın adı, tipi (funksiyanın qaytardığı nəticənin tipi) və formal parametrlərin (arqumentlərin) siyahısı verilir. Burada hər bir formal parametr tipi göstərilməklə sadalanır.

Funksiyanın təyini aşağıdakı formata malikdir:

funksiyanın tipi funksiyanın adı (formal parametrlərin siyahısı)

(funksiyanın gövdəsi).

Burada funksiyanın tipi-funksiyanın qaytardığı nəticənin tipi (əgər funksiya heç bir qiymət qaytarmırsa, onda onun üçün **void** tipi göstərilir), funksiyanın adı–istifadəçinin verdiyi identifikatordur və ya əsas funksiya üçün **main** adıdır. Formal parametrlər siyahısı – ya boş olur, ya hər biri üçün tipi göstərməklə funksiyanın formal parametrlərinin adları siyahısıdır, ya da **void** tipidir.

Funksiyanın təyinində hər bir parametrin verilmə formatı aşağıdakı kimidir:

parametrin tipi parametrin adı

və ya parametrin tipi parametrin adı=susmaqla verilən qiymət

Formatdan göründüyü kimi parametr üçün susmaqla veri-

lən qiymət mənimsədilə bilər.

Funksiyanın gövdəsi – həmişə ya blokdan, ya da qurma operatordan, yəni fiqurlu mötərizə daxilində verilən təsvir və operatorlar ardıcıllığından ibarət olur. Funksiya gövdəsinin ən vacib operatoru müraciət nöqtəsinə qayıdış operatorudur:

return ifadə;

və ya

return;

burada ifadə – funksiyanın qaytardığı qiyməti təyin edir, məhz bu qiymət funksiyaya müraciətin nəticəsi olacaqdır. Qaytarılan qiymətin tipi, funksiyanın tipi ilə təyin edilir. Əgər funksiya heç bir qiymət qaytarmırsa, yəni **void** tiplidirsə, onda **return** operatorunda ifadə verilmir. Bu halda funksiya gövdəsində **return** operatorunun özü də verilməyə bilər. Funksiyanın gövdəsində bir neçə **return** operatoru ola bilər. **Return** operatorundan **main** funksiyasında da istifadə edilə bilər. Funksiya heç bir əməliyyat yerinə yetirmədikdə və heç bir qiymət qaytarmadıqda belə onun gövdəsi daxili boş olan {}fiqurlu mötərizələrlə göstərilməlidir.

Funksiya təyininə aid misallara baxaq:

```
1)float min(float a,float b)
{if(a<b)return a;return b;
}
2) float cube(float x)
{return x*x*x;
}
3)int max(int n,int m)
{return n<m? m:n;
}
```

Funksiyaya müraciətin (funksiyanın çağırılmasının) formatı aşağıdakı şəkildədir:

funksiyanın adı (faktiki parametrlərin siyahısı)

C++ dilində funksiyaya müraciət ifadədir. Bu ifadədə yumru mötərizələr əməliyyat işarəsi rolunu oynayır və funksiya ilə faktiki parametrlər onun üçün operandlardır. Bu əməliyyatın yerinə yetirilmə üstünlüyü ifadələrdə ən yüksək olduğundan funksiyaların hesablanması digər əməliyyatlardan əvvəl yerinə yetirilir. Funksiyaya müraciətin yerinə yetirilməsinin nəticəsi, tipi

funksiyanın tipinə uyğun olan funksiyanın qaytardığı qiymət olur. Funksiyanın formal və faktiki parametrlərinin tipi, sayı və yerləşmə ardıcıllığı arasında qarşılıqlı birqiymətli uyğunluq olmalıdır. Müraciət yerinə yetirildikdə faktiki parametrlər uyğun formal parametrlərin yerinə qoyulur və hesablama bu konkret qiymətlər üçün aparılır. Turbo Pascal dili ilə müqayisə aparsaq, deyə bilərik ki, C++ dilində yalnız qiymət-parametrlər mümkündür. Buna görə də funksiyanın yerinə yetirilməsi, faktiki parametr kimi daxil edilmiş dəyişmələrin qiymətini dəyişdirə bilməz.

Beləliklə, faktiki parametrlərin siyahısı ya boş olur, ya **void** tipi olur, ya da bir-birindən vergüllə ayrılan ifadələrdən ibarət faktiki parametrlərin siyahısıdır.

Misal. **max(a,b,c)**-nin tapılması məsələsini **max(a,b,c)=max(max(a,b),c)** şəklində ifadə edək.

```
#include<iostream.h>
int MAX(int x,int y)
{if(x>y)return x;
else return y;
}
void main()
{int a,b,c,d;
cout<<"a,b,c:";
cin>>a>>b>>c;
d=MAX(MAX(a,b),c);
cout<<"\nmax(a,b,c)="<<d;
}
```

Funksiyaya müraciət zamanı, formal parametrlər, faktiki parametrlərlə əvəz olunur. Burada formal və faktiki parametrlərin tipləri arasında ciddi uyğunluq olmalıdır. Buna görə də funksiyaya birinci müraciətə qədər, onun tipi (nəticənin tipi) və bütün parametrlərinin tipləri haqqında məlumat olan təyin və ya təsvir (prototip) verilməlidir. Funksiyanın prototipi (təsviri) funksiyanın təyininin başlığı ilə tam üst-üstə düşə bilər, fərq yalnız prototipin sonunda nöqtə vergül işarəsinin olmasındadır:

funksiyanın tipi funksiyanın adı (formal parametrlərin siyahısı);

Burada həmçinin formal parametrlərin adlarını da verma-

mək olar. Yuxarıda misal kimi verdiyimiz funksiyaların uyğun prototiplərini verək:

```
1)float min(float a,float b);
2)float cube(float x);
3)int max(int,int m);
```

Proqramda prototiplər olduqda müraciət olunan funksiyalarla, bu funksiyalara müraciətlərin eyni bir faylda (modulda) olması məcburi deyildir.

Misal. Yuxarıda baxdığımız misalı (max(a,b,c)-nin tapılması) funksiya prototipindən istifadə etməklə həll edək:

```
#include<iostream.h>
int MAX(int,int);
void main()
{int a,b,c,d;
cout<<"a,b,c:";
cin>>a>>b>>c;
d=MAX(MAX(a,b),c);
cout<<"\n max(a,b,c)="<<d;
}
int MAX(int x,int y)
{if(x>y)return x;
else return y;
}
```

Misal. İki ədədin cəmi, fərqi və hasili üçün ən böyük ortaq bölənin tapılması məsələsini müqayisə üçün həm Turbo Pascal, həm də C++ dilərində yazaq.

```
program K1;
var a,b,z:integer;
function K2(m,n:integer):integer;
begin while m<>n do
if m>n then m:=m-n else n:=n-m;
K2:=m end;
begin write('a=');readln(a);
write('b=');readln(b);
z:=K2(K2(a+b,abs(a-b)),a*b);
```

```
writeln('z=',z)
end.
#include<iostream.h>
#include<math.h>
int K2(int,int);
void main()
{int a,b,z;
cout<<"a=";cin>>a;
cout<<"b=";cin>>b;
z=K2(K2(a+b,abs(a-b)),a*b);
cout<<"z="<<z;
}
int K2(int m,int n)
{while(m!=n)
{if(m>n)m=m-n;
else n=n-m;
}
return m;
}
```

Ayrı-ayrı fayllarda saxlanılan köməkçi funksiyalara kitabxana funksiyaları deyilir. Standart kitabxanalar C++ dilinin proqramlaşdırma sisteminə daxilidir. Bundan əlavə istifadəçi öz kitabxana funksiyalarını yarada bilər. Bu cür kitabxanalardan olan funksiyalardan istifadə etməmişdən əvvəl onları aşağıdakı əmrin köməyi ilə elan etmək tələb olunur:

#include<faylın adı>

burada faylın adı – bu və ya digər standart funksiyalar qrupunun prototiplərindən ibarət mətn faylının adıdır.

Məsələn: **#include<iostream.h>** - əmri ilə proqrama verilənlərin giriş-çıxışını təmin edən funksiyaların kitabxana siniflərinin təsviri daxil edilir, **#include<math.h>** - əmri isə riyazi funksiyalardan istifadəni təmin edir. Bütün başlıq faylları h (ingiliscə header sözündən) ad genişlənməsinə malikdir.

Ümümiyyətlə, C++ dilində aşağıdakı başlıq faylları mövcuddur: **alloc.h, complex.h, conio.h, constrea.h, cstring.h, ctype.h, dos.h, except.h, fcntl.h, float.h, fstream.h, graphics.h,io.h, iomanip.h,**

iostream.h, match.h,mem.h, new.h, process.h, signal.h, stdarg.h, stdarg.h, stdlib.h, stdio.h, string.h, strstrea.h,sys\stat.h, typeinfo.h.

Hər bir başlıq faylı müəyyən standart funksiyalardan istifadə edilməsini təmin edir və proqrama ehtiyac olduqda daxil edilirlər. Standart funksiyalar kitabxanalarında olduğu kimi istifadəçinin hazırladığı müxtəlif proqram modullarında yerləşdirilmiş kifayət qədər çox saylı funksiyaların prototiplərini, xarici obyektlərini (dəyişən, massiv və s.), təsvirlərini ayrıca bir faylda yerləşdirirlər və aşağıdakı əmrlə

#include "fayl adı"

proqrama daxil edirlər. Burada fayl adı bu funksiya prototiplərinin və təsvirlərin yerləşdirildiyi faylın adıdır. Məsələn, yuxarıda baxdığımız misallardakı funksiyalar üçün aşağıdakı başlıq faylı yazmaq olar:

```
//example.HPP
float min(float a,float b);
float cube(float x=1);
int max(int,int m=0);
```

Turbo Pascal dilində olduğu kimi C++ dilində də funksiyaların rekursiya təyininə icazə verilir. Məsələn, müsbət tam ədədin faktorialının hesablanması üçün yaza bilərik:

```
long fact(int k)
{if(k<0)return0;
if(k==0)return1;
return n*fact(k-1);
}
```

C++ dilində qüvvətə yüksəltmə əməli təyin olunmadığından sıfırdan fərqli həqiqi ədədin tam tərtibinin hesablanması üçün aşağıdakı rekursiv funksiya xeyirli ola bilər:

```
double expo(double a,int n)
{if(n==0)return1;
if(a==0)return0;
if(n>0)return a*expo(a,n-1);
if(n<0)return expo(a,n+1)/a;
}
```

Qeyd edək ki, massivlər də funksiyaların parametri ola bilər və funksiyalar massivə göstəricini nəticə kimi qaytara bilər.

Məsələn, aşağıdakı proqramda hər biri birölçülü parametr massiv kimi ifadə olunan iki vektor arasında qalan bucağın kosinusunu hesablayan funksiyadan istifadə olunur:

```
#include<iostream.h>
#include<math.h>
float cosinus(int n,float x[],float y[])
{float a=0,b=0,c=0;
for(int i=0; i<n;i++)
{a+=x[i]*y[i];
b+==x[i]*x[i];
c+=y[i]*y[i];
}
return a/sqrt(double(b*c))
}
void main()
{float E[]={1,1,1,1,1,1,1};
float G[]={-1,-1,-1,-1,-1,-1,-1};
cout<<"n kosinus="<<cosinus(7,E,G);
}
```

Funksiyaya göstərici aşağıdakı formada təyin olunur:

funksiyanın tipi(*göstəricinin adı)(parametrlərin tipi);

Məsələn, `int(*f1Ptr)(char);-char` tipli parametrli və qaytardığı nəticənin tipi `int` olan funksiyaya `f1Ptr` göstəricisinin təyinidir.

Funksiyaya göstərici vasitəsilə funksiyanı aşağıdakı formada çağırmaq olar:

(*göstəricinin adı)(faktiki parametrlərin siyahısı);

Misal.

```
#include<iostream.h>
int add(int n,int m){return n+m;}
int div(int n,int m){return n/m;}
int mult(int n,int m){return n*m;}
int subt(int n,int m){return n-m;}
void main()
{int(*par)(int,int);
int a=6,b=2;char c='+';
while(c!='')
{cout<<"\n arqument:a="<<a<<",b="<<b;
```

```
cout<<".rezult c=\'"<<c<<"\'"<<"=";
swith(c)
{case'+':par=add;c='/';break;
case'-':par=subt;c=' ';break;
case'*':par=mult;c='-';break;
case'/':par=div;c='*';break;
}
cout<<(a=(*par)(a,b));
}
}
```

C++ dilində istinad artıq mövcud obyektin başqa adla təyinidir. İstinad aşağıdakı formada təyin edilir:

type& istinadın adı=ifadə;

və ya

type& istinadın adı(ifadə);

Məsələn, `int L=666;`

`int&RL=L;`

Funksiyaya göstərici kimi funksiyaya istinad da təyin edilir:

funksiyanın tipi(&istinadın adı)(parametrlərin tipi) mənimsədilən ifadə;

Burada funksiyanın tipi – funksiyanın nəticə kimi qaytardığı qiymətin tipi, parametrlərin tipi – istinad etmək mümkün olan funksiyaların parametrlərinin tipi, mənimsədilən ifadə – istinad olunan funksiya ilə eyni olan artıq məlum funksiyanın adından ibarətdir.

Misal. Massivin maksimal qiymətli elementinə istinad edən funksiya təyin edilir.

```
#include<iostream.h>
int& max(int n,int d[])
int im=0;
for(int i=1;i<n;i++)
im=d[im]>d[i]?im:i;
return d[im];
}
void main()
{int n=4;
int x[]={10,20,30,14};
cout<<"\n max(n,x)="<<max(n,x);
```

```
max(n,x)=0;
for(int i=0;i<n;i++)
cout<<"x["<<i<<"]="<<x[i];
}
```

6.10. Sətirlər

C++ dilində Turbo Pascal dilində olduğu kimi verilənlərin xüsusi sətir tipi mövcud deyil. Burada sonuncu simvolu daxili kodu sıfıra bərabər \0 simvolu olan simvollar massivindən simvollar sətiri təşkil olunur. Turbo Pascal dilindən fərqli olaraq C++ dilində simvollar massivinin uzunluğuna məhdudiyyət (Turbo Pascal-da məhdudiyyət 255 simvoldur) qoyulmur. Sətir simvol massivi kimi təsvir olunur. Məsələn,

```
char STR[255];
```

Təsvir zamanı sətrə onun qiymətləri də mənimsədilə bilər. Bunun üçün iki üsul var: sətir sabiti vasitəsilə və simvollar siyahısı ilə:

```
char A[10]="C++dili";
char A[]="C++dili";
char
A[10]={'c','+','+','d','i','l','i','\0'};
```

Sətrin ayrı-ayrı simvolları indeksli adlarla ifadə olunur. Məsələn, `A[0]='C';A[5]='l';` və s. Sətrlərin emalı buradakı simvolların başdan sona qədər bir-bir yoxlanılması ilə əlaqədar olur. Bu cür yoxlamanın sona çatma əlaməti sıfır simvolunun tapılmasıdır. Aşağıdakı proqramda sətirdəki simvollar ardıcıl olaraq ulduz işarəsi ilə əvəz olunur.

```
#include<stdio.h>
#include<conio.h>
void main()
{char S[]="abcd";int i=0;clrscr();
puts(s);
while(S[i])
{S[i++]='*';puts(S);}
}
```

Bu proqramda `S[i]`, sıfır simvolunun qiymətini alana qədər dövr təkrar yerinə yetirilir. Sətrin ekrana çıxarılması üçün `stdio` standart kitabxanasında `puts()` funksiyasından istifadə

olunur. Bu funksiyada arqument kimi sətrin adı verilir. Bu kitabxanada sətrin klaviaturadan daxil edilməsi üçün **gets()** adlı funksiyadan istifadə olunur. Arqument kimi burada daxiletmə yerinə yetirilən sətrin adı verilir. C++ dilinin standart kitabxanaları arasında sətirlərin emalı üçün istifadə edilən başlıq faylı **string.h** olan kitabxana da var. Bu kitabxanadan olan **strlen()** funksiyası sətrin uzunluğunu təyin etməyə imkan verir.

Misal. Simvol sətrini daxil edib, sonra bu sətri tərsinə çıxarmalı. Məsələn, "abcdef" sətrini daxil etdikdə nəticədə "fedcba" sətrini alacağıq.

```
#include<stdio.h>
#include<string.h>
#include<conio.h>
void main()
{char c,S[10],int i;clrscr();gets(S);
for(i=0;i<=(strlen(S)-1)/2;i++)
{c=S[i];S[i]=S[strlen(s)-i-1];
S[strlen(S)-i-1]=c;}
puts(S);
}
```

6.11. Strukturlar və birləşmələr

C++ dilində struktur anlayışı Turbo Pascal dilindəki yazılış (record) anlayışına analojidir. Struktur – müxtəlif tipli elementlərin adlandırılmış külliyatından ibarət, verilənlərin strukturlaşmış tipidir. Struktur tipi adətən informasiya sistemlərinin, verilənlər bazasının hazırlanması üçün istifadə olunur.

Struktur tipinin təsvir formatı aşağıdakı kimidir:

struct tipin adı

{elementlərin təyini};

Sonda həmişə nöqtə vergül işarəsi qoyulur. Hər bir struktur, strukturun elementləri adlanan bir və ya bir neçə obyektdən (dəyişənlər, massivlər, göstəricilər, strukturlar və s.) ibarət olur. Məsələn, ixtiyari kitab üçün biblioqrafiya vərəqəsində aşağıdakı informasiya olur:

1) müəllifin soyadı, adı, atasının adı;
2) kitabın adı;
3) nəşr olunan yeri;
4) nəşriyyat;
5) nəşr ili;
6) səhifələrin sayı.

Biblioqrafik vərəqəyə daxil olan verilənlər haqqında məlumatları struktur vasitəsilə aşağıdakı struktur tip şəklində göstərmək olar:

```
struct card{char*author;
            char*title;
            char*city;
            char*firm;
            int year;
            int pages;
           };
```

Bu cür təyin, struktur tip adlanan yeni törəmə tip daxil edir. Bu misalda həmin tipin adı **card** kimi təyin edilib. Strukturun elementləri həm baza tipli, həm də törəmə tipli ola bilər. Məsələn, baxdığımız misalda **card** strukturunda **int** baza tipli və **char*** törəmə tipli elementlər var. Struktur tipi təyin etdik-dən sonra konkret strukturlaşmış obyektləri təyin edib, təsvir et-mək olar, məsələn:

```
card x1,x2,x3;
```

burada **x1,x2,x3** – struktur tipinin dəyişənləridir. Struktur dəyişənlərinin təsvirinin digər variantları da var. Struktur tipinin adını vermədən, dəyişənləri birbaşa təsvir etmək olar. Məsələn, qrupdakı tələbələr haqqında məlumatı əlaqələndirən aşağıdakı strukturda olduğu kimi:

```
struct stud{char fam[30];
            int kurs;
            char grup[3];
           }stud1,stud2,stud3;
```

Konkret strukturun elementlərinə müraciət etmək üçün aşağıdakı formaya malik dəqiqləşdirilmiş addan istifadə olunur:

strukturun adı. struktur elementinin adı;

Məsələn, **stud1.fam;stud1.kurs;**

Strukturun təyini zamanı, onun elementlərinin başlanğıc

qiymətlərini də daxil etmək olar. *Məsələn,*

```
card x1={"Aliyev A.Y.","Informatika",
Baku","Baku University",2007,300};
```

Bu cür təyin aşağıdakı operatorlar ardıcıllığı ilə ekvivalentdir:

```
card x1;
x1.author="Aliyev A.Y.";
x1.tile="Informatika";
x1.city="Baku";
x1.firm="Baku University";
x1.year=2007;
x1.pages=300;
```

Struktur tipin təyini bu tipin konkret strukturlarının təyini ilə müşayət oluna bilər:

```
struct PRIM{char*name;long s;}A,B,C;
```

Burada **PRIM** adlı struktur tip və eyni daxili quruluşa malik **A,B,C** struktur təyin edilib. Struktur tipin adı, tip adlarının bütün hüquqlarına malik olduğundan strukturlara göstəricilər təyin etməyə icazə verilir:

struktur tipin adı * struktura göstəricinin adı;

Adətən, təyin edilən göstəriciyə başlanğıc qiymətlər də mənimsədilə bilər. Struktura göstəricinin qiyməti həmin tipli strukturun ünvanı ola bilər. Məsələn, yuxarıda baxdığımız misal üçün struktura göstəricini aşağıdakı kimi vermək olar:

```
stud*pst,stud1;
```

Burada `pst=& stud1;` mənimsəməsini apardıqdan sonra **stud1** struktur dəyişənin hər bir elementinə aşağıdakı üç qayda ilə müraciət etmək olur (məsələn, fam sahəsi üçün):

stud1.fam və ya **(*pst) .fam** və ya **pst->fam**

Axırıncı variantda struktur elementinə keçiddə "**->**" əməliyyat işarəsindən istifadə olunur. Eyni qayda ilə strukturların digər elementlərinə müraciət etmək olar. Axırıncı iki müraciət forması-nı ümumi şəkildə aşağıdakı kimi göstərmək olar:

göstəricinin adı **->** struktur elementinin adı

və

(*göstəricinin adı). strukturun elementinin adı

Digər obyektlərdə olduğu kimi, strukturlar üçün istinadlar

təyin edilə bilər:

struktur tipin adı&
struktura istinadın adı
başlanğıc qiymətlərinin mənimsədilməsi.

Məsələn, yuxarıda verdiyimiz **PRIM** struktur tipi üçün A və B strukturlarına istinadları aşağıdakı kimi vermək olar:

```
PRIM& refA=A;
PRIM& refB(B);
```

Burada **refA** və **refB** istinadlarına başlanğıc qiymətlər müxtəlif qaydalarla mənimsədilib. Bu cür təyinlərdən sonra **refA**, A struktur adının sinonimi, **refB**-də, **B** strukturunun başqa adıdır. İndi **A.sum** ilə **refA.sum.*B.name** ilə **refB.name;** və **A.name** ilə **refA.name** müraciətləri öz araların-da ekvivalentdirlər.

Qeyd edək ki, strukturların elementlərinin özləri də struktur tipə aid ola bilər. Burada eləcə də struktur massivlərindən istifadə edilə bilər. Məsələn, 100 tələbə haqqındakı məlumat aşağıdakı kimi təsvir olunan massivdə də saxlanıla bilər:

```
stud stud1[100];
```

Onda ayrı-ayrı tələbələr haqqındakı məlumatlar **stud1[1].fam; stud1[5].kurs;** və s. kimi işarə edilə bilər. Məsələn, 50-ci tələbənin soyadının birinci hərfini təyin etmək üçün **stud1[50].fam[0];** yazmaq lazımdır.

Misal. N sayda tələbə haqqında məlumat verilən strukturdan ən yüksək bal toplamış tələbələrin soyadlarını təyin etməli.

```
#include<stdio.h>
#include<conio.h>
void main()
{
const N=30;int i,max;
struct student{char fam[15];
               int kurs;
               char grup[3];
               int bal;
               }
student stud[N];clrscr();
for(i=0;i<N;i++)
```

```
    {printf("\n"soyadi:");scanf("%s",&stud[i].
fam);
    printf("Kurs:");scanf("%d",&stud[i].kurs);
    printf("Grup:");scanf(%s",&stud[i].grup);
    printf("Ball:");scanf("%f",&stud[i].bal);
    }
    max=0;
    for(i=0;i<N;i++)
    if(stud[i].bal>max)max=stud[i].bal;
    printf("\n Maksimal bal toplayan",max);
    for(i=0;i<N;i++)
    if(stud[i].bal==max)printf("\n%s",stud[i].
fam);
    }
```

Birləşmə – verilənlərin daha bir struktur tiplərindəndir. Birləşmə struktura oxşardır və öz təsvirində strukturdan yalnız onunla fərqlənir ki, burada **struct** sözü əvəzinə **union**-dan istifadə olunur:

union tipin adı

{birləşmə elementləri};

Məsələn, `union mix{double d;long E[2],int K[4];};`

Birləşmə tipini daxil etməklə, konkret birləşmələri, onların massivlərini, bu birləşmələrə göstərici və istinadları təyin etmək olar. *Məsələn*,

```
mix mA,mB[4];
mix*pmix;
mix& rmix=mA;
```

Birləşmə elementlərinə müraciət etmək üçün aşağıdakı qaydalardan istifadə etmək olar:

1) dəqiqləşdirilmiş adla:
birləşmənin adı. elementin adı;
2) göstərici ilə:
birləşməyə göstərici–>elementin adı;
və ya
(*birləşməyə göstərici). elementin adı;
3) istinadla:
birləşəməyə istinad. elementin adı;

Məsələn,

```
mA.d=78.8;
mB[2].E[1]=100L;
pmix=&mB[0];pmix->E[0]=88;
cin>>(*pmix).K[1];
cin>>rmix.E[0];
```

6.12. Proqram mətninin emal vasitələri

C++ dilində proqramlaşdırma mühitinə və ya dilin kompilyatoruna hökmən preprosessor daxil olur. Preprosessorun təyinatı, proqramın başlanğıc mətninin onun kompilyasiyasına qədər emalını təmin etməkdir. C++ dilində preprosessor emalı ardıcıl yerinə yetirilən aşağıdakı mərhələlərdən ibarət olur:

1) bütün sistemdən asılı işarələr standart kodlarla əvəz olunur;

2) '\' və "sətrin sonu" simvolları cütləri götürülür və nəticədə başlanğıc faylın növbəti sətri, bu simvolların olduğu sətrə birləşdirilir;

3)proqram mətnində preprosessorun əmrləri oxunur;

4) preprosessorun əmrləri yerinə yetirilir;

5) simvol sabit və sətirlərdə *Esc*-ardıcıllıqlar,(məsələn '\n') uyğun ədədi kodlarla əvəz olunur;

6) ayrı-ayrı yerləşən simvol sətirləri birləşdirilir.

Preprosessoru idarə etmək üçün əmrlərdən istifadə olunur və hər bir əmr yeni sətirdən verilir, qarşısında # işarəsi qoyulur. Bu əmrlər aşağıdakılardır:

#define,#include,#undef,#if,#ifdef,#ifndef ,#else,#endif,#elif,#line,#error,#pragma,#.

Burada

1) **#define** əmri müəyyən simvollar ardıcıllığı ilə əvəz ediləcək preprosessor identifikatorlarını təyin edir;

2) **#include** əmri seçilmiş fayldan mətni proqram mətninə daxil etməyə imkan verir;

3) **#undef** əmri, **#define** əmrinin fəaliyyətini ləğv edir;

4) **#if,#ifdef,#ifndef** əmrləri, **#else, #endif, #elif** əmrləri ilə birlikdə proqram mətninin şərti emalını təşkil

edir (burada şərtilik ondadır ki, proqramın bütün mətni deyil, yalnız bu əmrlərlə qeyd olunmuş hissələri kompilyasiyadan keçir);

5) **#line** əmri proqramla fayldakı sətirlərin nömrələnməsini idarə etməyə imkan verir (burada faylın adı və sətrin başlanğıc nömrəsi **#line** əmrində göstərilir);

6) **#error** əmri səhv olduğu halda, bu səhv haqqında məlumat almağa imkan verir;

7) **#pragma** əmri kompilyatorun konkret işindən asılı olaraq əməliyyatlar yerinə yetirir;

8) # əmri boş əmr olduğundan heç bir əməliyyat yerinə yetirmir.

İdentifikatorun qabaqcadan hazırlanmış simvollar ardıcıllığı ilə əvəz edilməsi üçün **#define** identifikator əvəz edilən sətir konstruksiyasından istifadə edilir:

Məsələn,

başlanğıc mətn	emalın nəticəsi
#define begin{	**void main()**
#define end}	**{**
void main()	**operatorlar**
begin	
operatorlar	**}**
end	

Burada **begin-end** operatorlar mötərizəsi C++ dilindəki uyğun {} standart mötərizələri ilə əvəz olunacaqdır.

Proqram mətnində əvəzləməni aşağıdakı əmrlə ləğv etmək olar:

#undef identifikator

Fayldan mətni daxil etmək üçün aşağıdakı əmrdən istifadə olunur:

#include<faylın adı**>**

və ya

#include "faylın adı"

Birinci halda preprosessor, faylı standart sistem kataloqlarında axtarır, ikinci halda isə əvvəlcə istifadəçinin cari kataloqunda sonra isə standart sistem kataloqlarında axtarır.

C++ dilində ixtiyari proqramın əvvəlində

#include<iostream.h> əmrini verdikdə preprosessor proqramla giriş-çıxış kitabxanası arasında əlaqə yaradır. Qeyd edək ki, **iostream.h** faylı standart sistem kataloqlarında axtarılır. Burada **.h** – ad genişlənməsi, proqramın başlığında yəni operatorların yerinə yetirilməsinə qədər yerləşdirilən fayllarda istifadə olunur.

Proqram mətninin şərti emalını təşkil etmək üçün aşağıdakı əmrlərdən istifadə olunur:

1) **#if** sabit ifadə;
2) **#ifdef** idetifikator;
3) **#ifndef** identifikator;
4) **#else;**
5) **#endif;**
6) **#elif.**

Bunlardan birinci üç əmr şərtlərin yoxlanılmasını, sonrakı iki əmr isə yoxlanılan şərtin yerinə yetirilmə diapazonunu təyin edir. Proqramın başlanğıc mətninin emalı zamanı budaqlanmanın təşkili üçün **#elif** sabit ifadə əmrindən istifadə edilir.

Sətirlərin nömrələnməsi üçün

#line sabit

əmrindən istifadə olunur, bu əmr sətirləri nömrələməklə yanaşı fayla ad da verə bilər:

#line sabit"faylın adı"

Burada **#error** əmri səhv olduqda diaqnostik məlumat mətninin verilməsini təmin edir.

6.13. Siniflər

Sinif – tip verilənlər və funksiyalardan element kimi istifadə edən struktur tipdir. Sinif tipi obyektlər üçün təyin edilir. Siniflərin elanı, strukturların elanına oxşardır. Elan **class** sözü ilə başlayır, onun ardınca isə sinifə verilən ad gəlir. Sadə halda sinifin elanı aşağıdakı şəkildədir:

class ad
{tip 1 dəyişən 1
tip 2 dəyişən 2
.....

```
public
funksiya 1;
funksiya 2;
.....
};
```

Siniflərin, strukturlardan əsas fərqi ondadır ki, sinifin bütün elementləri susmaqla əvvəlcədən qapalı hesab olunur və onlara yalnız bu sinfin hədləri olan funksiyalar yiyələnə bilər. Lakin sinfin elementlərinə yiyələnmə rejimi, onun aşkar göstərilməsi ilə dəyişdirilə bilər. Bunun üçün sinfin elementləri qarşısında uyğun rejim adı verilir. Bunlar aşağıdakılardır:

1)private (xüsusi)

2)public (ümumi)

3)protected (qorunan)

Private yiyələnmə rejimi bildirir ki, bu elementdən yalnız elə bu sinfin funksiyaları istifadə edə bilər. Bu rejim susmaqla əvvəlcədən verilir. **Public** rejiminə uyğun elementlər proqramın digər yerlərində də istifadə oluna bilər. **Protected** rejimi siniflərin varislik xassələri ilə bağlıdır və bu rejimə sonra baxılacaqdır. Çox vaxt verilənlər (dəyişənlər) üçün **private**, funksiyalar üçü isə **public**, rejimindən istifadə olunur.

Misal kimi adi kəsrlərin **Drob** adlı sinfinə baxaq. Burada kəsrin qiyməti iki tam ədədin (surət və məxrəc) strukturu kimi təyin edilib. Kəsrlə iş üsulları isə aşağıdakılardır: kəsrlərin daxil edilməsi – **Vvod** funksiyası, surət və məxrəcin ən böyük ortaq böləninin hesablanması – **NOD** funksiyası, kəsrin ixtisar edilməsi – **Sokr** funksiyası, kəsrin tam tərtibindən qüvvətə yüksəldilməsi – **Stepen** funksiyası və nəhayət kəsrin ekrana çıxarılması – **Print** funksiyası. Burada uyğun sinif aşağıdakı kimi elan edilir:

```
class Drob{
        Frac A;
        public:
        void Vvod(void);
        int Nod(void);
        void Sokr(void);
        void Stepen(int N);
        void Print(void);
```

```
};
```

Burada, nəzərə alınır ki, bu sinfə nəzərən qlobal olaraq aşağıdakı struktur elan edilib:

```
Struct Frac{int P;int Q;};
```

Beləliklə, beş üsul, beş funksiya ilə ifadə olunub və sinfin elanında öz prototipləri ilə göstəriliblər. Sinif hədləri olan funksiyaların təsviri ayrıca aparılır. Burada funksiyanın hansı sinfə aid olduğunu da göstərmək lazımdır. Bunun üçün "::" aid olmaq əməliyyatından istifadə edilir. *Məsələn,*

```
void Drob ::Vvod(void)
{cin>>A.P;
 cin>>A.Q;
}
```

Proqramın əsas hissəsində (əsas funksiyada) **Drob** sinfi müəyyən dəyişənlər üçün tip kimi təyin edilir.Məsələn: **Drob Y;**. Bundan sonra **Y** dəyişəni proqramda uyğun sinfin obyekti kimi qəbul olunur. Həmin obyektin elementlərinə isə müraciət qurma adla yerinə yetirilir.

Məsələn: **Y.Vvod()**

İndi proqrama tam şəkildə baxaq:

```
#include<iostream.h>
#include<math.h>
struct Frac {int P; int Q;};
Frac F;
class Drob {
Frac A;
Public:
void Vvod(void);
int Nod(void);
void Sokr(void);
void Stepen(int N);
void Print(void);
};
void Drob :: Vvod(void)
{ cin >> A.P;
cin >> A.Q;
}
```

```
int Drob::Nod(void)
{ int M,N;
M=abs(A.P);N=A.Q;
while (M!=N)
{ if (M>N)
if (M%N!=0)M=M%N; else M=N;
else if (N%M!=0) N=N%M; else N=M;
}
return M;
}
void Drob::Sokr(void)
{int x;
x=NOD( );
if(A.P!=0){A.P=A.P/x;A.Q=A.Q/x;}
else A.Q=1;
}
void Drob::Stepen(int N)
{int i;
F.P=F.Q=1;
for(i=1;i<=N;i++)
{F.P*=A.P;F.Q*=A.Q;}
}
void Drob::Print(void)
{cout<<"\n"<<A.P<<"/"<<A.Q<<"\n";}
void main(void)
{Drob Y;
cout<<"Drob!"<<"\n";
Y.Vvod( );
Y.Sokr( );
cout<<"ixtisar:"<<"\n";
Y.Print( );
Y.Stepen(2);
cout<<"kvadratı:"<<"\n";
cout<<F.P<<"/"<<F.Q<<"\n";
}
```

Siniflərin təyinində onların obyektlərinə başlanğıc qiymətlərini mənimsətmək üçün konstruktor adlanan xüsusi funksiyadan istifadə etmək olur. Konstruktorun sinfin gövdəsində təyin

etmə formatı aşağıdakı kimidir:

sinfin adı (formal parametrlər siyahısı)
{konstuktorun gövdəsinin operatorları}

Konstruktorun adı sinfin adı ilə üst-üstə düşməlidir. Bu cür funksiya, sinfin hər bir obyektinin **new** operatoru ilə təyini zamanı avtomatik olaraq çağırılır. Konstruktorun əsas təyinatı – obyektlərin qiymətlərinin mənimsədilməsidir. Konstruktor ixtiyari sinif üçün yaradıla bilər, sinifdə bir neçə konstruktor ola bilər. Konstruktorun ünvanını almaq olmaz, konstruktorun parametri, onun öz sinfi ola bilməz. Konstruktorun aşkar çağırılması üçün aşağıdakı qaydalardan istifadə etmək olar:

sinfin adı obyektin adı (konstruktorun faktiki parametrləri)

və ya

sinfin adı (konstruktorun faktiki parametrləri);

Obyektin ləğv edilməsi zamanı konstruktor tərəfindən ayrılmış yaddaş hissəsinin azad edilməsi üçün sinfin destruktor adlanan elementindən istifadə edilir. Onun standart formatı aşağıdakı kimidir:

~sinfin adı ( ) {destruktorun gövdəsinin operatorları};

C++dilində destruktor həmişə "~" simvolu ilə başlayır və onun ardınca probel olmadan sinfin adı yerləşdirilir. Destruktorun parametrləri ola bilməz və o heç bir qiymət qaytarmır. Destruktorun çağırışı sinfin obyekti ləğv edilən kimi avtomatik olaraq yerinə yetirilir. Məsələn, aşağıda sətir obyektləri üçün sinif elan edilir. Burada konstruktor **new** operatoru ilə **string1** göstəricisi üçün yaddaş bloku ayırır. Ayrılmış yaddaş hissəsi destruktorla **delete** operatorunun köməyi ilə azad edilir:

```
class string operation
{char *string1;
int string len;
public:
string operation(char*)
{string1=new char[string len];}
~string operation( )
{delete string1;}
void input data(char*);
```

```
void output data(char*);
};
```

Sinfin təyini zamanı onun gövdəsində sinfin verilənləri və funksiyaları da təyin edilir. Sinfin verilənlərinin təyini zahiri olaraq baza və törəmə tip obyektlərin adi təsvirinə oxşardır. Buna görə də sinfin verilənlərinə onun elementləri demək olar. Verilənlərin adı təyinindən fərqli olaraq, siniflərdə, sinif elementlərinin başlanğıc qiymətlərinin mənimsədilməsinə icazə verilmir.

Sinfin elementləri verilən – elementlərə və funksiya – elementlərə bölünür. Sinfin funksiya elementlərinə göstərici aşağıdakı formatda verilir:

funksiyanın qaytardığı qiymətin tipi
(sinfin adı `::*` üsula göstəricinin adı)
(funksiyanın parametrləri);

Sinfin verilən – elementlərinə göstərici isə aşağıdakı formatda verilir:

verilənlərin tipi (sinfin adı ::* göstəricinin adı);

Sinfə aid funksiya, konkret obyektin verilənlərinin emalı üçün çağırılarkən bu funksiyaya onun çağrıldığı obyektə göstərici avtomatik olaraq verilir. Bu göstərici qeyd olunmuş **this** adına malikdir və sinfin hər bir funksiyasında aşağıdakı formada təyin edilib:

sinfin adı * **const this**=emal edilən obyektin adı;

Məsələn,

```
struct SS
{ int si;char SC;
SS{int in,char cn)
{ this->si=in;this->SC=cn;}
void print(void)
{ cout<<"\n si="<<this->si;
 cout<<"\n SC="<<this->SC;
 }
};
```

Sinfin elementi olmayıb, lakin onun qorunan və xüsusi elementlərinə müraciət edə bilən funksiyalara, sinfin dost funksiyaları deyilir. Funksiyanın sinfin dost funksiyası olması üçün onun sinfin gövdəsində **friend** işçi sözü ilə təsvir edilməsi tələb olunur. Misal olaraq, "nöqtə müstəvidə" və "düz xətt müstəvidə"

siniflərinə dost olan funksiyaya baxaq. Birinci sinifdə nöqtənin müstəvidəki (x,y) koordinatları, ikinci sinifdə isə düz xəttin $Ax+By+C=0$ ümumi tənliyinin A,B,C əmsalları veriləcəkdir. Dost funksiya verilmiş nöqtənin verilmiş düz xəttdən yayınma qiymətini təyin edir. Əgər (a,b) verilən nöqtənin koordinatlarıdırsa, onda A,B,C əmsalları olan düz xətt tənliyi üçün yayınma $Aa+Bb+C$ ifadəsi ilə hesablanır.

```
#include<iostream.h>
class line2;
class point2;
{ float x,y;
public:
point2(float xn=0,float yn=0)
{ x=xn; y=yn;}
friend float uclon(point2,line2);
};
class line2;
{ float A,B,C;
public:
line2(float a,float b,float c)
{ A=a;B=b;C=c;}
friend float uclon(point2,line2);
};
float uclon(pont2 p,line2 l)
{return l.A*p.x+l.B*P.y+l.c;}
void main(void)
{ point2 p(16.0,12.3);
line2 L(10.0,-42.3,24.0);
cout<<"\n Yayınma:";cout<<uclon(P,L);
}
```

C++ dilində ixtiyari standart əməliyyatı istifadəçi tərəfindən daxil edilən yeni verilənlər tipinə tətbiq etmək olur. Bunun üçün standart əməliyyatların əlavə yüklənməsi mexanizmindən istifadə edilir. Əməliyyiatların yeni verilənlər tipinə tətbiq edilməsi üçün istifadəçi "əməliyyat-funksiya" (**operator function**) adlanan xüsusi funksiyanı təyin edir. Əməliyyat-funksiyanın təyini aşağıdakı formata malikdir:

qaytarılan qiymətin tipi operator əməliyyatın işarəsi
(əməliyyat-funksiyanın parametrləri)
{əməliyyat-funksiyanın gövdəsinin operatorları};

Misal. Vektor sinfi Evklid fəzasında üçölçülü vektor təyin edir. Bu sinifdə toplama (+) və mənimisətmə (=) əməliyyatlarını yükləyərək, onları üçölçülü vektorlara tətbiq edəcəyik.

```
#include<iostream.h>
class vector
{int x,y,z;
public:
vector operator+(vector t);
vector operator=(vector t);
void show(void);
void assign(int mx,int my,int mz);
}
vector vector::operator+(vector t)
{ vector temp;
temp.x=x+t.x; temp.y=y+t.y; temp.z=z+t.z;
}
vector vector::operator=(vector t)
{x=t.x; y=t.y; z=t.z;
return*this;
}
void vector::show(void)
{ cout<<x<<",";
cout<<y<<",";
cout<<x<<\n';
}
void vector::assign(int mx,int my;int mz)
{x=mx; y=my; z=mz;}
void main(void)
{ vector a,b,c;
a.assign(1,2,3);b.assign(10,10,10);
a.show( );b.show( );
c=a+b; c.show( );
c=a+b+c; c.show;
c=b=a; c.show; b.show;
}
```

Müxtəlif siniflərin obyektləri və siniflərin özləri varislik münasibətlərində ola bilərlər. Artıq mövcud siniflər əsasında yeni siniflər yaratmaq mümkün olur. Mövcud siniflər adətən baza (yaradan) siniflər, bu baza sinifləri əsasında qurulan yeni siniflər törəmə (yaradılan), yaxud varis-siniflər və ya sadəcə varislər adlanırlar. Varis-siniflər, baza siniflərin bütün verilənlərinə və üsullarına malik olur, eləcə də bundan əlavə öz elementləri (verilənlər və üsullar) ilə də tamamlanırlar.

Varis – siniflərin təyin və təsvirində, bu sinfin verilənləri və üsulları hansı baza siniflərindən alındığını göstərmək üçün burada həmin baza siniflərin adları sadalanmalıdır. *Məsələn,*

```
class S:X,Y,Z {...};
```

burada **S** sinfi **X,Y,Z** siniflərindən törənib.

Misal. Aşağıdakı misalda **circsqrt** adlı varis sinif (kvadratda çevrə sinfi) **circ** ('çevrə' sinfi) və **square** ('kvadrat' sinfi) adlı baza sinifləri əsasında qurulur.

```
//circ.cpp
#include<iostream.h>
class circ
{ int xc,yc,rc;
public:
circ(int xi, int yi, int ri)
{xc=xi; yc=yi; rc=ri;}
void show( )
{circle(xc,yc,rc);}
void hide( )
{int bk,cc;
bk=getbkcolor( ); cc=getcolor( );
setcolor(bk);
circle(xc,yc,rc); setcolor(cc);
}
};
//square.cpp
#include<graphics.h>
class square
{int xq,yq,lq;
void rissquare(void)
{int d=lq/2; line(xq-d,yq-d,xq+d,yq-d);
```

```
   line(xq-d,yq+d,xq+d,yq+d);
   line(xq-d,yq-d,xq-d,yq+d);
   line(xq+d,yq- d,xq+d, yq+d);
   }
   public:
   square(int xi,int yi,int li)
   {xq=xi; yq=yi; lq=li;}
void show( )
{rissquare( ):}
void hide( )
{int bk,cc;bk=getbkcolor( );cc=getcolor( );
setcolor(bk);rissquare( );setcolor(cc);
}
};
#include<conio.h>
#include"square.cpp"
#include"circ.cp"
class circsqrt:public circ,public square
{public:
circsqrt(int xi,int yi,int ri);
circ(xi,yi,ri),square(xi,yi,2*ri)
{}
void show(void)
{circ::show( );square::show( );}
void hide( )
{square::hide( );circ::hide( );}
};
void main( )
{int dr=detect,mod;
initgraph(&dr,&mod,"c:\\borlandc\\bgi");
circsqrtA1(100,100,60);
circsqrt F4(400,300,50);
A1.show( ); getch( ); F4.show( ); getch( );
F4.hide( ); getch( ); A1.hide( );getch( );
closegraph( );
}
```

6.14. C++ dilində giriş-çıxış prosedurları

C++ dilində qurulan proqramlarda eyni hüquqlu olaraq iki giriş-çıxış prosedurları kitabxanasından C (ANSI C standartı) dilinin funksiyalarının standart kitabxanasından və məxsusi olaraq C++ dili üçün yaradılmış siniflər kitabxanasından istifadə edilir. C dilinin funksiyalar kitabxanası, proqram başlığına **stdio.h** faylı daxil edilməklə, proqramda istifadə edilə bilər.

Qeyd edək ki, proqramlaşdırmada giriş-çıxış prosesi operativ yaddaşla, xarici qurğular (klaviatura, displey, maqnit informasiya daşıyıcıları və s.) arasındakı informasiya mübadiləsi prosesidir. Giriş informasiyanın xarici qurğulardan operativ yaddaşa köçürülməsi, çıxış isə informasiyanın operativ yaddaşdan xarici qurğulara çıxarılmasıdır. Displey, printer kimi qurğular yalnız çıxış, klaviatura isə giriş üçün nəzərdə tutulub. Maqnit informasiya daşıyıcıları (disklər, maqnit lentlər və s.) həm giriş, həm də çıxış üçün istifadə edilir. İxtiyari giriş-çıxış əməliyyatı, faylların mübadiləsi əməliyyatı kimi müəyyən olunur. Giriş – fayldan operativ yaddaşa informasiyanın oxunması, çıxış isə – operativ yaddaşdan informasiyanın fayla yazılmasıdır.

C/C++ dillərində giriş-çıxış əməliyyatları axın anlayışı ilə bağlıdır. Axın – giriş-çıxış prosesində ötürülən bayt ardıcıllığıdır. Axın hər hansı xarici qurğu və ya diskdəki faylla əlaqənməlidir, yəni axın hər hansı qurğuya və ya fayla yönəldilməlidir.

C++ dilində **iostream.h** (**stream** – axın, **"i"** – **input**-giriş sözünün, **"o"** – isə **output** – çıxış sözünün qısaldılmasıdır) başlıq faylı buradakı giriş-çıxış vasitələri ilə verilənlərin axından çıxarılmasını və axına daxil edilməsini təmin edir. Axınla informasiya mübadiləsi zamanı çox vaxt əsas yaddaşın köməkçi hissəsindən – axın buferindən istifadə olunur.

Proqramın çıxaracağı verilənlər, xarici qurğuya ötürülənə qədər buferdə yerləşdirilirlər. Eləcə də verilənlərin daxil edilməsi zamanı, onlar əvvəl buferdə yerləşdirilir və yalnız bundan sonra yerinə yetirilən proqramın yaddaş obalstına ötürülürlər. Proqramlarda istifadə olunan axınlar məntiqi olaraq aşağıdakı üç tipə bölünür:

1) informasiyanın oxunduğu giriş axınları;
2) verilənlərin daxil edildiyi çıxış axınları;

3) həm oxunmanı, həm yazılışı təmin edən iki istiqamətli axınlar.

Giriş-çıxış kitabxanasının bütün axınları ardıcıldırlar, yəni zamanın hər bir anı üçün axın üçün yazılışın və (və ya) oxunmanın mövqeləri təyin edilib və bu mövqelər informasiya mübadiləsindən sonra ötürülmüş verilənlər payının uzunluğu qədər axın üzrə yerdəyişmə edirlər. Axının qoşulduğu qurğunun xüsusiyyətlərindən asılı olaraq axınlar – standart, konsol, sətir və fayl axınlarına bölünürlər. Standart kitabxanalardan (riyazi funksiyalar və ya sətirlər ilə iş kitabxanaları) fərqli olaraq giriş-çıxış kitabxanaları funksiyalar deyil, siniflər kitabxanasıdır.

Axın sinifləri kitabxanası iki baza sinfin **ios** və **streambuf** siniflərinin əsasında qurulub. Burada **streambuf** sinfi istifadəçinin aşkar və ya qeyri-aşkar istifadə etdiyi bütün törəmə siniflərdəki verilənləri buferdə yerləşdirir. Bu sinif yaradılan axınlarla fiziki qurğular arasında əlaqə yaradır. Digər **ios** sinfi isə həm giriş, həm də çıxış üçün ümumi olan elementlərə (verilənlər və üsullar) malikdir. Giriş-çıxışın axın kitabxanası ilə iş zamanı aşağıdakı siniflərdən geniş istifadə olunur:

1) **ios** – baza axın sinfi;
2) **istream** – giriş axınları sinfi;
3) **ostream** – çıxış axınları sinfi;
4) **iostream** – giriş-çıxış axınlarının iki istiqamətli sinfi;
5) **istrstream** – giriş sətir axınları sinfi;
6) **ostrstream** – çıxış sətir axınları sinfi;
7) **strstream** – giriş-çıxış sətir axınlarının iki istiqamətli sinfi;
8) **ifstream** – giriş fayl axınları sinfi;
9) **ofstream** – çıxış fayl axınları sinfi;
10) **constream** – konsol çıxış axınları sinfi.

Axın siniflərini, onların verilənlərini və üsullarını proqrama daxil etmək üçün aşağıdakı başlıq fayllarından istifadə edilir:

1) **iostream.h–ios, istream, ostream, stream** sinifləri üçün;
2) **strstrea.h–istrstream, ostrstream, strstream** sinifləri üçün;
3) **fstream.h–ifstream, ofstream,fstream**

sinifləri üçün;

4) **constrea.h-constream** sinfi üçün.

Qeyd edək ki, **ios** sinfi qalan axın sinifləri üçün baza sinfi olduğundan **strstrea.h,constrea.h** və ya **fstream.h** başlıq fayllarından ixtiyari biri proqrama daxil edirlərsə avtomatik olaraq proqrama **iostream.h** faylı qoşulur. Burada **iostream.h** faylı proqrama **ios, istream, ostream, stream** siniflərinin təsvirlərini daxil etməklə yanaşı, giriş-çıxış standart axınlarının da təyinini təmin edir:

1) **cin-istream** sinfinin obyekti olub, standart buferli giriş axını ilə əlaqəlidir;

2) **cout-ostream** sinfinin obyekti olub, standart buferli çıxış axını ilə əlaqəlidir;

3) **cerr-ostream** sinfinin obyekti olub, səhvlər haqqında məlumatlar göndərilən standart bufersiz çıxış axını ilə əlaqəlidir;

4) **cloq-ostream** sinfinin obyekti olub, səhvlər haqqında məlumatlar göndərilən, standart buferli çıxış axını ilə əlaqəlidir.

Proqrama **iostream.h** faylı daxil edilərkən hər dəfə **cin,cout, cerr, cloq** obyektlərinin formalaşdırılması prosesi gedir, yəni uyğun standart axınlar yaradılır və istifadəçiyə onlarla əlaqədar olan giriş-çıxış vasitələrindən istifadə etmək imkanı verilir. Burada **istream** sinfinin giriş əməliyyatı verilənlərin axından çıxarılması (oxunması) adlanır və sağa yerdəyişmə əməliyyatının "**>>**"simvolu ilə işarə olunur. **Ostream** sinfinin çıxış əməliyyatı verilənlərinin axına daxil edilməsi (yazılması) adlanır və sola yerdəyişmə əməliyyatının "**<<**"simvolu ilə işarə olunur. Burada

cin>> baza tipinin obyektinin adı

cout<< baza tipinin ifadəsi

cerr<< baza tipinin ifadəsi

cloq<< baza tipinin ifadəsi.

Qeyd edək ki, burada **>>** və **<<** əməliyyatları axınlarla yalnız o vaxt əlaqə yaradırlar ki, onlar axın obyektlərindən sağda verilsin. Əks halda onlar yalnız yerdəyişmə əməliyyatını bildirir. Verilənlərin axından alınması və ya axına salınması üçün istifadə

olunan << və >> əməliyyatları buradakı ifadədə əməliyyatların yerinə yetirilmə üstünlüyünü dəyişdirmir. Məsələn, `cout<<2+3+4;` operatorunun yerinə yetirilməsinin nəticəsində ekrana 9 qiyməti veriləcəkdir. Beləliklə, verilənlərin axına daxil edilməsi (çıxarılması) əməliyyatında sağda duran operandlar kimi mötərizəsiz verilən hesabi ifadələrdən istifadə etmək olar. Lakin << əməliyyatından yerinə yetirilmə üstünlüyü daha aşağı olan əməliyyatların iştirak etdiyi ifadələri axından çıxarmaq üçün mötərizələrdən istifadə olunur:

```
cout<<(a+b+c);
cout<<(x<0?-x:x);
```

Axından çıxarışı, əməliyyatlar "zənciri" şəklində ifadə etmək daha rasional olur:

```
(cout<<"\nx*2=")<<x*2;
```

Əgər x=33 olarsa, ekrana x*2=66 çıxarılacaq. K=1 üçün

```
cout<<"\nk*2="<<(k<<1)<<
      "k<<2="<<(k<<2);
```

ekranda alarıq

```
k*2=2  k<<2=4
```

Aşağıdakı operatorların yerinə yetirilməsi nəticəsində

```
int k=1;
cout<<"\nk++="<<k++<<
    " (k+=3)="<<(k+=3)<<"k++="<<k++;
```

ekranda alarıq:

```
k++=5 (k+=3)=5  K++=1.
```

Əməliyyatlar "zəncirindən" verilənlərin axına daxil edilməsi zamanı da istifadə etmək olur. Məsələn, `int i,j,k,l;` üçün `cin>>i>>j>>k>>l;` operatoru bu dəyişənlərə uyğun tam qiymətlərin daxil edilməsini təmin edir. Bu qiymətlər bir-birindən boş yerlərlə ayrılmaqla daxil edilir. Məsələn, **1 2 3 4 <Enter>**. Bu qiymətlərin hər biri bir sətirdə yerləşdirilib, sonlarında **<Enter>** düyməsinin sıxılması ilə də daxil edilə bilər.

Massiv elementlərinin çıxarılması üçün onları aşkar olaraq indeksli adları ilə göstərmək lazımdır:

```
float A[3]={10.0,20.0,30.0};
cout<<A[0]<<" "<<A[1]<<" "<<A[2];
```

Massiv elementlərinin daxil edilməsi üçün aşağıdakı qayda-

dan istifadə etmək olar:

```
double B[5];
for(int i=0; i<5; i++)cin>>B[i];
```

bu zaman klaviaturada məsələn, aşağıdakı qiymətlər ardıcıllığını yığmaq lazımdır:

```
0.01  0.02  0.03  0.04  0.05<Enter>
```

Tam ədədlərin daxil edilməsi və çıxarılması zamanı bu ədədlərin daxili və xarici ifadə uzunluqlarına məhdudiyyətlər qoyulur. Məsələn,

```
int i;cin>>i; cout<<"\ni="<<i;
```

operatorları üçün klaviaturada 123456789 ədədini yığsaq, ekrana bu hal üçün **i=-13035** çıxarılacaqdır. Həmin ədəd ekrana daxil edildiyi kimi aşağıdakı operatorlar verildiyi halda çıxarılacaq:

```
long p;
cin>>p; cout<<"\np="<<p;
```

Tam ədədi daxil edərkən, onu klaviaturada səkkizlik və onaltılıq say sistemlərində yığmaq olar. Məsələn,

```
int N; cin>>N; cout<<"\nN="<<N;
```

operatorlarının yerinə yetirilməsi halında 077777 səkkizlik qiymətini daxil edib, cavabı onluq şəkildə N=32767 almaq olar. Eləcə də -0×7FFF onaltılıq qiymətini daxil edib, alarıq N=-32767 və s.

Həqiqi ədədlərin daxil edilməsi zamanı onların ixtiyari yazılış formalarından istifadə etmək olar, yəni onlar giriş axınında (klaviaturadan daxil edilməklə) qeyd olunmuş onluq nöqtə və ya sürüşən onluq nöqtə şəklində ifadə oluna bilər. Məsələn,

```
float pi;cin>>pi;cout<<"pi="<<pi;
```

operatorları 3.141593 və ya 3.141593e0 və ya +3.141593 və ya 0.3141593 e+1 ədədi qiymətinin daxil edilməsi halında həmişə ekrana pi=3.141593 qiymətini çıxaracaq. Əgər həqiqi qiymətin daxil edilməsi zamanı klaviaturada 3.14159265358979323**85<Enter>** qiyməti yığıldıqda, **cout** çıxış axınında yenə də **p=3.141593** çıxarılacaq. Əgər həqiqi qiymət kiçikdirsə və qeyd olunmuş onluq nöqtə ilə yazılış halında onun əhəmiyyətli hissəsi mərtəbə şəbəkəsindən kənara çıxırsa, onda qiymət sürüşən onluq nöqtə formasında çıxarılır. Məsələn,

```
float a; cin>>a;
cout<<"\na="<<a;
```

operatorları üçün klaviaturadan 0.00000000000000000000001 **<Enter>**daxil etdikdə ekrana a=1e-23 çıxarılacaq. Klaviaturada 1.0e-23 yığdıqda da eyni nəticə almaq olar.

Standart axından həqiqi qiymətin daxil edilməsi zamanı klaviaturada böyük tam ədəd yığmaq olar. Məsələn,

```
double d; cin>>d;
cout<<"d="<<d;
```

operatorları üçün klaviaturadan 112233445566778899 **<Enter>** daxil etməklə, **d**=1.122334e+17 yuvarlaqlaşdırılmış qiyməti almaq olar.

Standart **cout,cin,cerr,cloq** axınlarına << (axına daxil etmək) və >> (axından çıxarılma) əməliyyatlarını tətbiq etdikdə baza tipləri üçün "susmaqla" ifadə formatları müəyyən edilir. Məsələn, ədədlərin çıxarışı zamanı onların hər biri ifadə edilməsi üçün tələb olunan sayda mövqe tutur. Məsələn,

```
int i1=1,i2=2,i3=3,i4=4,i5=5;
cout<<"\n"<<i1<<i2<<i3<<i4<<i5;
```

operatorlarının yerinə yetirilməsi 12345 nəticəsini verir. Çıxarılan informasiyanın ifadə formatları, istifadəçi tərəfindən formatlaşdırma bayraqları vasitəsilə dəyişdirilə bilər. Formatlaşdırma bayraqlarını qeyd edək:

1) **skipws=0×0001** – bu halda axından çıxarmaq >> əməliyyatı probel simvolarını nəzərə almır;

2) **left=0×0002** – çıxarılan qiymətlər sahənin sol tərəfinə sıxılır;

3) **right=0×0004** – çıxarılan qiymətlər sahənin sağ tərəfinə sıxılır;

4) **internal=0×0008** – tamamlayıcı simvol (susmaqla probel) ədədlə, onun işarəsi arasında yerləşdirilir;

5) **dec=0×0010** – onluq say sistemi;

6) **oct=0×0020** – səkkizlik say sistemi;

7) **hex=0×0040** – onlatılıq say sistemi;

8) **showbase=0×0080** – çıxarış zamanı say sisteminin əlamətini göstərir (0× - onaltılıq say sisetmi, 0 – səkkizlik say sistemi);

9) **showpoint=0×0100** – həqiqi ədədlərin çıxarışı zamanı onluq nöqtənin göstərilməsini təmin edir;

10) **uppercase=0×0200** – ədədlərin çıxarılması zamanı yuxarı reqist hərflərindən istifadəni təmin edir: x simvolu və ABCDEF hərfləri onaltılıq say sistemi üçün, sürüşən onluq nöqtəli ədədlər üçün tərtib göstəricisi E;

11) **showpos=0×0400** – müsbət ədədlərin çıxarılışı zamanı ədədin işarəsinin ("+" simvolu) verilməsi;

12) **scientific=0×0800** – həqiqi ədədlərin (**float,double** tipli) sürüşən onluq nöqtə şəklində ifadəsi;

13) **fixed=0×1000** – həqiqi ədədlərin (**float,double** tipli) qeyd olunmuş onluq nöqtə şəklində ifadəsi;

14) **unitbuf=0×2000** – hər yeni çıxarış zamanı axınları təmizləyir;

15) **stdio=0×4000** – hər yeni çıxarış zamanı **stdout,stderr** axınlarını təmizləyir.

Bayraqlardan başqa formatın idarə edilməsi üçün **ios** sinfinin aşağıdakı dəyişənlərindən istifadə olunur:

1) **int x width** – çıxarış üçün minimal enli sahə verir;

2) **int x precision** – həqiqi ədədlərin çıxarışı zamanı kəsr hissəsinə maksimal sayda rəqəm yeri ayrılır;

3) **int x fill** – çıxarış sahəsinin **x width** ilə təyin olunan minimal eninə qədər hansı simvolla tamamlanacağını təyin edir (susmaqla bu simvol boş yer simvoludur).

Formatlaşdırma bayraqlarının və parametrlərinin dəyişdirilməsinin daha sadə üsulu manipulyatorlardan istifadədir. Manipulyatorlar – axının bayraq və vəziyyətlərinin dəyişdirilməsinə imkan verən xüsusi funksiyalardır. C++ dilinin giriş-çıxış sinifləri kitabxanasının manipulyatorları iki qrupa bölünür: parametrli manipulyatorlar və parametrsiz manipulyatorlar. Əvvəlcə parametrsiz manipulyatorları qeyd edək:

1) **dec** – giriş-çıxış zamanı onluq say sisteminin bayrağını təyin edir;

2) **hex** – giriş-çıxış zamanı onaltılıq say sisteminin bayrağını təyin edir;

3) **oct** – giriş-çıxış zamanı səkkizlik say sisteminin bayrağını təyin edir;

4) **ws** – daxil etmə zamanı tətbiq edilir və giriş axınından probellərin çıxarılmasını təmin edir;

5) **endl** – çıxış zamanı tətbiq edilir və çıxış axınına yeni sətir simvolunun daxil edilməsini təmin edir;

6) **ends** – çıxış zamanı tətbiq edilir və axına sətrin sonunu bildirən sıfır əlamətini daxil edir;

7) **flush** – yalnız çıxışda istifadə olunur və çıxış axınını təmizləyir.

Parametrli manipulyatorlar **iomanip.h** faylında təyin ediliblər:

1) **setbase(int n)** – say sisteminin əsasını (**n**) təyin edir, burada **n** parmetri 0, 8,10 və ya 16 ola bilər. Parametr 0 olduqda çıxışda say sistemi kimi onluq say sistemi seçilir;

2) **resetiosfags(long L)** – giriş-çıxış axınlarında **L** parametrinin qiymətlərinə görə ayrı-ayrı vəziyyət bayraqlarını atır;

3) **setiosflags(long L)** - giriş-çıxış axınlaında **L** parametrinin qiymətlərinə görə ayrı-ayrı vəziyyət bayraqları təyin edir;

4) **setfill(int n)-n** parametrinin qiyməti tamamlayıcı simvolun kodunu müəyyən edir;

5) **setprecision(int n)**- giriş-çıxış zamanı **n** parametrinin köməyi ilə həqiqi ədədlərin kəsr hissəsindəki rəqəmlərin sayını müəyyən edir;

6) **setw(int n)-n** parametrinin qiyməti çıxış sahəsinin minimal enini təyin edir.

Qeyd edək ki, **cin>>** operand və **cout<<** operand ifadələrində sağda duran operandın hər bir tipinə verilənlərin çevrilməsi üçün öz qaydası uyğun gəlir. Bu qaydalar əvvəlcədən operasiya-funksiya adlanan xüsusi prosedur (funksiya) şəklində təyin ediliblər. Bu funksiyaların təyini giriş-çıxış kitabxanalarında, prototipləri isə **iostream.h** faylında yerləşdirilib.

Misal.

```
#include<iostream.h>
struct address
{char*country; char*city; char*street;
 int number_of_house;
```

```
};
ostream &
operator<<(ostream & out,address ad)
{out<<"\n country:"<<ad.country;
 out<<"\n city:"<<ad.city;
 out<<"\n street:"<<ad.street;
 out<<"\n house:"<<ad.number_of_house;
 return out;
}
void main( )
{address ad={"Azerbaijan","Baku",
 "Xalilov",23};
 cout<<"\n Address:";
 cout<<ad<<"\n";
}
```

Nəticədə alarıq:

```
Address:
Country:Azerbaijan
city   :  Baku
street:  Xalilov
house  :  23
```

Misal. Aşağıdakı proqramda $\sin x$ və $\cos x$ funksiyalarının [0,p] parçasında n addım üçün qiymətləri cədvəli verilir. Cədvəlin qurulmasında manipulyatorlardan istifadə edilir.

```
#include<iostream.h>
#include<math.h>
#include<iomanip.h>
void main( )
{double a,b,x;int n=20;a=0;b=4*atan(1.);
cout<<"x sin(x)  cos(x) "<<endl;
cout<<"......"<<endl;
for(x=a; x<=b; x=x+(b-a)/n)
cout<<setprecision(4)<<setw(10)<<x<<" "
<<setprecision(4)<<setw(10)<<sin(x)<<" "
<<setprecision(4)<<setw(10)<<cos(x)<<endl;
}
```

6.15. C++ dilində fayllarla iş

C++ dilində giriş-çıxış kitabxanası ardıcıl fayllarla iş üçün vasitələrə malikdir. Fayllarla giriş-çıxış axınları arasında əlaqəyə baxarkən, aşağıdakı mövcud prosedurları qeyd etmək lazımdır:

1 – faylın yaradılması;
2 – axının yaradılması;
3 – faylın açılması;
4 – faylın axına qoşulması;
5 – axının köməyi ilə faylla mübadilə;
6 – axının fayldan ayrılması;
7 – faylın bağlanması;
8 – faylın ləğv edilməsi.

Bu sadalanan əməliyyatların hamısı C++ dilinin giriş-çıxış sinifləri kitabxanası vasitəsilə yerinə yetirilə bilər.

Fayl aşağıdakı kitabxana funksiyasının (ANSI C kitabxanası) köməyi ilə yaradıla bilər:

```
int creat(const char *path,int amode);
```

Bu funksiyanın prototipi **io.h** faylında yerləşir. Bu funksiya faylın verilmiş **path** adı ilə yeni fayl yaradır və ya artıq mövcud faylı təmizləyib iş üçün hazırlayır. Burada **amode** parametri yalnız yeni yaradılan fayl üçün lazım olur və yaradılan fayla keçid rejimini təyin edir. Bu parametr üçün **sys\stat.h** başlıq faylında aşağıdakı sabit-qiymətlər müəyyən olunub:

1) **S_IWRITE** – fayla yazılış etməyə icazə verilir;
2) **S_IREAD** – fayldan yalnız oxumağa icazə verilir;
3) **S_IREAD | S_IWRITE** – həm yazılış, həm də oxumağa icazə verilir.

Fayllarla iş üçün axınlar aşağıdakı siniflərin obyektləri kimi yaradılırlar:

1) **ofstream** – verilənlərin fayla daxil edilməsi üçün;
2) **ifstream** – verilənlərin fayldan çıxarılması üçün;
3) **fstream** – verilənlərin fayldan oxunması və fayla yazılması üçün.

Bu siniflərdən istifadə etmək üçün proqram mətninə əlavə **fstream.h** başlıq faylını daxil etmək lazımdır. Bundan sonra proqramda konkret fayl axınları təyin etmək olar (**ofstream**,

ifstream, fstream siniflərinin obyektləri üçün), məsələn aşağıdakı kimi:

ofstream outFile; – çıxış fayl axınını təyin edir,

ifstream inFile; – giriş fayl axınını təyin edir,

fstream ioFile; – giriş-çıxış fayl axınını təyin edir.

Fayl axını yaratdıqdan sonra onu konkret fayla **open( )** funksiyası ilə qoşmaq olar. Bu funksiyanın köməyi ilə faylı açmaqla yanaşı, onu müəyyən axınla əlaqələndirmək olar. Funksiyanın formatı aşağıdakı kimidir:

```
void open(const char *fileName,
          int mode=susmaqla qəbul olunan qiymət,
          int protection= susmaqla qəbul olunan qiymət);
```

Brinci parametr **fileName** – artıq mövcud olan və ya yeni yaradılan faylın adıdır. İkinci parametr – **mode** (rejim) açılan faylla işin rejimini (məsələn, yalnız yazılış və ya yalnız oxunuş) təyin edən bayraqlardır. Bayraqlar aşağıdakı kimi təyin edilib:

```
enum ios::open_mode {
in=0×01,// Yalnız oxuma üçün açılır
out=0×02,// Yalnız yazılış üçün açılır
ate=0×04,// Açılış zamanı faylın sonu axtarılır
app=0×08,// Verilənləri faylın sonuna yazmaq
trunc=0×10,// Mövcud fayl əvəzinə yenisini yaratmaq
nocreate=0×20,// Yeni fayl açmamaq
noreplace=0×40,// Mövcud faylı açmamaq
binary=0×80,// İkilik (mətni olmayan) mübadilə üçün açmaq
}
```

Üçüncü parametr – **protection** (müdafiə) – müdafiəni təyin edir və çox nadir hallarda istifadə olunur.

Adətən **open( )** funksiyasının çağırışı dəqiqləşdirilmiş ad üzrə aparılır:

sinfin obyektinin adı.sinfə aid funksiyanın çağırışı

Beləliklə, faylın açılması və konkret fayl axınına birləşdiril-

məsi **open()** funksiyasının aşağıdakı çağırışı ilə yerinə yetirilir:

axının adı.**open**(faylın adı, rejim, müdafiə);

Burada axının adı **ofstream, ifstream, fstream** siniflərinə aid obyektlərdən birinin adıdır. *Məsələn,*

```
outFile.open("C:\\USER\\R.DAT");
inFile.open("DATA.TXT");
ioFile.open("R1.DAT",ios::out);
```

6.16. C++ dilinin qrafik imkanları

C++ dilində qrafik qurmalar yaratmaq üçün qrafik kitabxanadan istifadə etmək tələb olunur. Qrafik kitabxanadan istifadə üçün proqram mətninə **graphics.h** başlıq faylını daxil etmək lazımdır. Qrafik sistemi idarə etmək üçün aşağıdakı funksiyalardan istifadə edilir:

1) **closegraph** – qrafik rejimdən mətn rejiminə keçidi təmin edir, prototipi: **void far closegraph(void)** ;

2) **graphdefaults** – susmaqla qrafik sistemin bütün parametrlərini təyin edir, prototipi: **void far graphdefaults(void)**;

3) **initgraph** – qrafik iş rejiminə keçidi təmin edir, prototipi: **void far initgraph(int far *graphdriver,int far *graphmode, char far *pathtodriver)**;

4) **restorecrtmode** – mətn rejiminə müvəqqəti qayıdışı təmin edir, prototipi: **void far restorecrtmode(void)**;

5) **setactivepage** – **page** səhifəsini qrafikanın çıxarılması üçün aktiv edir, prototipi: **void far setactivepage(int page)**;

6) **setallpalette** – palitranın bütün rənglərini təyin edir, prototipi: **void far setallpalette(struct palettetype far *palet-te)**;

7) **setaspectratio** – **x, y** koordinatları üzrə sıxılma əmsallarını təyin edir, prtotipi: **void far setaspectratio(int xasp,int yasp)**;

8) **setbkcolor** – fonun rəngini təyin edir, prototipi: **void far setbkcolor(int color);**

9) **setcolor** – təsvirin rəngni təyin edir, prototipi: **void far setcolor(int color);**

10) **setgraphmode** – **initgraph( )** funksiyasından fərqli olan qrafik rejimə keçidi təmin edir, prototipi: **void far setgraphmode(int mode);**

11) **setlinestyle** – xətlərin qalınlığını və tipini təyin edir, prototipi: **void far setlinestyle(int linestyle,unsigned upattern, int thickness);**

12) **setpalette** – palitranın rənglərindən birini təyin edir, prototipi: **void far setpalette(int colornum,int color);**

13) **settextstyle** – **outtext( )** funksiyası ilə çıxarılan mətnin stilini (şrift, simvolların ölçüləri) təyin edir, prototipi: **void far settextstyle(int font, int direction, int charsize);**

14) **setviewport** – təsvirlərin və mətnin çıxarılması üçün ekranın cari pəncərəsinin ölçülərini təyin edir, prototipi: **void far setviewport(int left, int top, int right, int bottom, int clip);**

15) **setvisualpage** – **page** qrafik sətrini əks etdirir, prototipi: **void far setvisualpage(int page);**

16) **setwritemode** – qrafik rejimdə xətlərin çıxarılması rejimini təyin edir, prtotipi: **void far setwritemode(int mode);**

17) **arc** – mərkəzi **(x,y)** nöqtəsində olan qövsün çəkilməsini təmin edir, prototipi: **void far arc(int x, int y, int stangle, int endangle, int radius);**

18) **bar** – rənglənmiş düzbucaqlının çəkilməsini təmin edir, protoitpi: **void far bar(int left, int top, int right, int bottom);**

19) **bar3d** – rənglənmiş paralelepipedin çəkilməsini təmin edir, prototipi: **void far bar3d(int left, int top, int right, int bottom, int depth, int topflag);**

20) **circle** – mərkəzi **(x,y)** nöqtəsində olan çevrənin çəkilməsini təmin edir, prototipi: **void far circle(int x, int y, int radius);**

21) **cleardevice** – ekranı fonun rəngi ilə təmizləyir, prototipi: **void far cleardevice(void)**;

22) **drawpoly** – **numpoints** təpə nöqtələri olan çoxbucaqlının konturunu çəkir, prototipi: **void far drawpoly(int numpoints, int far *polypoints)**;

23) **ellipse** – mərkəzi **(x,y)** nöqtəsində olan ellips çəkir, prototipi: **void far ellipse(int x, int y, int stangle,int endangle, int xradius, int yradius)**;

24) **fillellipse** – mərkəzi **(x,y)** nöqtəsində olan ellipsi qurur və onu əvvəlcədən təyin olunmuş rənglə doldurur, prototipi: **void far fillellipse(int x, int y, int xradius, int yradius)**;

25) **line** – **(x1,y1)**-dən **(x2,y2)**-yə qədər xətti çəkir, prototipi: **void far line(int x1, int y1, int x2, int y2)**;

26) **linerel** – cari nöqtədən, ondan **(dx,dy)** qədər aralı olan nöqtəyə qədər xətt çəkir, prototipi: **void far linerel(int dx, int dy)**;

27) **lineto** – cari nöqtədən **(x,y)** koordinatlı nöqtəyə qədər xətt çəkir, prototipi: **void far lineto(int x, int y)**;

28) **moverel** – mövqe göstəricisini cari nöqtədən, bu nöqtədən **(dx,dy)** qədər aralı olan nöqtəyə keçirir, prototipi: **void far moverel(int dx, int dy)**;

29) **moveto** – mövqe göstəricisini cari nöqtədən **(x,y)** koordinatlı nöqtəyə keçirir, prototipi: **void far moveto(int x, int y)**;

30) **outtext** – **textstring** mətn sətrini cari nöqtədən başlayaraq çıxarır, prototipi: **void far outtext(char far *textstring)**;

31) **outtextxy** – textstring mətn sətrini **(x,y)** koordinatlı nöqtədən başlayaraq çıxarır, prototipi: **void far outtextxy(int x, int y, char far *textstring)**;

32) **putpixel** – **(x,y)** koordinatlı nöqtəni çıxarır, prototipi: **void far putpixel(int x, int y, int color)**;

33) **rectangle** – verilmiş təpə nöqtəli düzbucaqlını çəkir, prototipi: **void far rectangle(int left, int top, int right, int bottom)**.

III BÖLMƏ

PROQRAMLAŞDIRMA ÜZRƏ MƏSƏLƏLƏR

BASIC/QBASIC dilində proqramlaşdırma üzrə məsələlər

Dilin sadə konstruksiyaları. Xətti strukturlu sadə proqramlar. Budaqlanma

1. Aşağıda gətirilmiş sabitlərin hansı tipə aid olduğunu təyin etməli:

1) `"4";`	2) `1998;`	3) `"BASF";`	4) `0.3798;`
5) `375;`	6) `12E-16;`	7) `0.25E 12;`	8) `777777777;`
9) `387.0;`	10) `123D-30;`	11) `123456.7;`	12) `"INTEGER";`
13) `-2778;`	14) `0.112442;`	15) `9877.1 D17;`	

2. Aşağıda göstərilmiş simvol ardıcıllıqlarından hansı identifikator (dəyişən adı) hesab edilə bilər:

1) `INTEGER;`	2) `2A;`	3) `ALFA;`	4) `ARRAY 1;`
5) `FORTRAN $;`	6) `ALT #;`	7) `.DATA;`	8) `RAT 100%;`
9) `PRINT;`	10) `A3$;`	11) `A[3];`	12) `A(3);`
13) `123 CENT;`	14) `A(3)%;`	15) `PI 3.14`	

3. Aşağıdakı hesabi ifadələrdə hesabi əməllərin yerinə yetirilmə ardıcıllığını təyin etməli:

1) `-a*b/a*a/c*b;` 2) `-3.4+a*sin(a+b*2)^3`
3) `a^(b+c)-(a-b);` 4) `x^(y^z)*a;`
5) `x+x/(x+y)*z/(z+x)+(x-y)/z;` 6) `s*(a-b)*m/n;`

4. Aşağıdakı cəbri ifadələri BASIC dilində yazmalı:

1) $1+x+x^2/2+x^3/3$; 2) `(a+b)sin(a)+(a-b)cos(b);`

3) $(a-b)/(c+\frac{b}{c+b/(c-d)})$; 4) $e^{x}-3\sin(x)+e^{(x-1)}$;

5) $3\sin(x)+4\cos^2(x^2)-1$; 6) $\sqrt{a^3+b+|y-x|^3}$;

7) $z^2-3+z^2/5$.

5. Tam `x` ədədi verilib. Hesabla:

1) $F=(x+2)^2+8(x+2)^2$; 2) $F=5x^3+2(x^3+2)^2$;
3) $F=(x+2)+3(x+2)^2$; 4) $F=x^2(x^2+1)$;
5) $F=4x^3+2(x^3+1)^3$; 6) $F=(x+1)/2+(x+1)^2$;
7) $F=x^2/2+(x^2/2)^2+5$; 8) $F=x/2+(x/2)^2$.

6. Aşağıda verilmiş proqramların nəticələrini təyin etməli:

1)
```
10 A=3
20 B=A+7
30 PRINT B
```

2)
```
10 C=0
20 D=2*C-3
30 PRINT C,D
```

3)
```
10 X=10
20 Y=53-X
30 PRINT X,Y
```

4)
```
10 L=5
20 L=L+1
30 PRINT L
```

5)
```
10 F=5
20 K=F^2+5
30 PRINT F,K
```

6)
```
10 A=10: B=5
20 B=B+A
30 PRINT A,B
```

7. `x,y,z` həqiqi ədədləri verilib. Onların ədədi ortasını tap.
8. `x,y` həqiqi ədədləri verilib. Onların cəmini, fərqini, hasilini və nisbətini tap.
9. $A(x_1,y_1)$ və $B(x_2,y_2)$ nöqtələri arasındakı məsafəni tap. $(D=\sqrt{(x_2-x_1)^2+(y_2-y_1)^2})$.
10. Üçbucaq `a,b,c` tərəfləri ilə verilib. Heron düsturundan istifadə etməklə üçbucağın sahəsini tap $(P=(a+b+c)/2,\ S=\sqrt{P\cdot(p-a)\cdot(p-b)\cdot(p-c)})$.
11. Aşağıda verilən proqramların nəticələrini təyin etməli.

1)
```
10 A=1
20 PRINT A,A^2
30 A=A+2
40 IF A<=10 THEN 20
50 END
```

2)
```
10 F=0
20 PRINT 2*F,F
30 F=F+3
40 IF F<=15 THEN 20
50 END
```

3)
```
10 P=16
20 PRINT P,P^2
30 P=P-2
40 IF P>=6 THEN 20
50 END
```

4)
```
10 T=12
20 C=36-T
30 PRINT T,C
40 T=T-2
50 IF T>=0 THEN 20
60 END
```

5)
```
10 H=20
20 C=H-8
30 PRINT H,C
40 H=H-6
50 IF H>=0 THEN 20
60 END
```

6)
```
10 I=1
20 P=7*I
30 PRINT I,P
40 I=I+1
50 IF I<=10 THEN 20
60 END
```

12. `x` və `y` həqiqi ədədləri verilib. Əgər `x>y` olarsa, `S`-ə `x-y`, əks halda `y-x+2` mənimsətməli.
13. `x` və `y` həqiqi ədədləri verilib. Əgər `x+y>0` olarsa, `S`-ə x^2+y^2, əks halda $(x+y)^2$ mənimsətməli.
14. `x` həqiqi ədədi verilib. Əgər `x<2` olarsa, onda `S`-ə x^2, əks halda `4` mənimsətməli.
15. `x` həqiqi ədədi verilib. Əgər `x<5` olarsa, onda `S`-ə x^2+4x+5, əks halda $1/(x^2+4x+5)$ mənimsətməli.
16. `x` və `y` tam ədədləri verilib. Əgər `x*y>0` olarsa, onda `S`-ə `x-sin(y)`, əks halda `1-x+cos(y)` mənimsətməli.
17. `x` və `y` həqiqi ədədləri verilib. Bu ədədlərdən ən böyüyünü təyin etməli.
18. `x,y,z` həqiqi ədədləri verilib. Bu ədədlərdən ən kiçiyini təyin etməli.
19. `a,b,c` həqiqi ədədləri verilib. `a<b<c` bərabərsizliyinin ödənib-ödənmədiyini yoxlamalı.

20. x həqiqi ədədi verilib. Əgər 0<x<2 şərti ödənirsə, S-ə x+2 qiymətini, əks halda x^2-2 qiymətini mənimsətməli.
21. x,y,z həqiqi ədədləri- verilib. Onlardan müsbət olanlarının kvadratını tapmalı.
22. a,b,c həqiqi ədədləri verilib. Tərəfləri a,b,c olan üçbucaq qurmaq olarmı?
23. x,y,z həqiqi ədədləri verilib. Əgər x+y>z olarsa, S-ə x+y+z qiymətini, əks halda x-y-z qiymətini mənimsətməli.
24. x,y,z həqiqi ədədləri verilib. Əgər x-y<z olarsa, S-ə $x^2-y+z/2$ qiymətini, əks halda $\sin(x)-y^2+z$ qiymətini mənimsətməli.
25. x həqiqi ədədi verilib. Əgər x>2 olarsa, S-ə 1-x qiymətini, -2<x<1 olarsa, x^2- nı, qalan hallarda isə 2*x qiymətini mənimsətməli.
26. x həqiqi ədədi verilib. Əgər x<0 olarsa, S-ə x^3 qiymətini, -3<x<1 olarsa, x/2-ni, qalan hallarda isə x+2 qiymətini mənimsətməli.
27. Verilmiş N tam ədədi üçün vurma cədvəlini qurmalı.
28. 10-dan 25-ə qədər natural ədədlərin cəmini tapmalı.
29. 12-dən 26-dək natural ədədlərin hasilini tapmalı.
30. 6-dan 22-dək natural ədədlərin cəmini 2 dəyişmə addımı ilə tapmalı.
31. 8-dən 27-dək natural ədədlərin hasilini 3 dəyişmə addımı ilə tapmalı.
32. 30-dan 1-dək natural ədədlərin cəmini -2 addımı ilə tapmalı.
33. Üç müsbət tam ədəd verilib. Onların ədədi və həndəsi ortalarını tapmalı.
34. Düzbucaqlı üçbucağın katetləri verilib. Onun perimetrini, sahəsini, daxilinə və xaricinə çəkilmiş çevrələrin radiuslarını hesablamalı.
35. Radiusu r olan çevrə xaricinə çəkilmiş düzgün n-bucaqlının perimetr və sahəsini tapmalı.
36. Radiusu r olan çevrə daxilinə çəkilmiş düzgün n-bucaqlının perimetr və sahəsini tapmalı.
37. Bərabərtərəfli üçbucağın tərəfi verilib. Onun perimetrini, sahəsini, daxilinə və xaricinə çəkilmiş çevrələrin radiuslarını hesablamalı.
38. Düzbucaqlı üçbucağın kateti və bu katetə söykənən iti bucağı verilib. Onun perimetrini, sahəsini, daxilinə və xaricinə çəkilmiş çevrələrin radiuslarını tapmalı.
39. Düzbucaqlı üçbucağın kateti və bu katetin əks tərəfinə söykənən iti bucaq verilib. Onun perimetrini, sahəsini, daxilinə və xaricinə çəkilmiş çevrələrin radiuslarını tapmalı.
40. Düzbucaqlı üçbucağın hipotenuzu və iti bucağı verilib. Onun perimetrini, sahəsini, daxilinə və xaricinə çəkilmiş çevrələrin radiuslarını tapmalı.
41. Çevrənin uzunluğu verilib. Bu çevrənin daxilinə və xaricinə çəkilmiş düzgün üçbucaqların sahələrini tapmalı.
42. Dairənin sahəsi verilib. Bu dairənin daxilinə və xaricinə çəkilmiş kvadratların sahələrini və perimetrlərini tapmalı.
43. Üçbucaq öz bucaqlarının qiymətləri və xaricinə çəkilmiş dairənin radiusu ilə verilib.Üçbucağın perimetrini və sahəsini tapmalı.
44. Üçbucaq tərəfi və bu tərəfə söykənən iki bucağı ilə verilib. Bu üçbucağın daxilinə və xaricinə çəkilmiş dairələrin sahələrini tapmalı.
45. Üçbucaq iki tərəfi və bu tərəflər arasında qalan bucaqla verilib. Üçbucağın

perimetrini və sahəsini tapmalı.

46. Üçbucaq tərəflərinin uzunluqları ilə verilib. Üçbucağın hündürlüyünü, medianını, tənbölənini və onun daxilinə, xaricinə çəkilmiş çevrələrin radiuslarını tapmalı.
47. Üçbucaq öz təpə nöqtələrinin koordinatları ilə verilib. Üçbucağın perimetrini və sahəsini tapmalı.
48. Üçbucaq tərəflərinin uzunluqları ilə verilib. Üçbucağın bucaqlarını tapmalı.
49. Rombun xaricinə çəkilmiş dairənin sahəsini, radiusunu, rombun verilmiş diaqonallarına əsasən tapmalı.
50. Dairə öz radiusu ilə verilib. Dairənin xaricinə və daxilinə çəkilmiş düzgün n-bucaqlıların (n=3,4,6) sahələrini tapmalı.
51. Verilmiş **x,y,z** qiymətlərinə əsasən aşağıdakıları hesablamalı:

1) $a=\dfrac{\sqrt{|x^2-y|}-\sqrt[4]{|y+x|}}{1+x^2+y^2+z^2}$; $b=x\operatorname{arctg}\left(\dfrac{x^2+y^2}{1+z^2}\right)$

2) $a=\sin(x^2+y^2-z)\cdot e^{-x}$; $b=\sqrt[3]{|x^2+\cos y|+1}$

3) $a=\cos(x^2+1)\sqrt{1+y^2}$; $b=\operatorname{tg}(xy)+y^2$

4) $a=(1+x^2y^2)\sin(x+y)$; $b=\sqrt{1+\operatorname{tg}^2(x^2+1)}$

5) $a=\cos(xe^{-y})+\sqrt{x^2+e^y}$; $b=\sin(x\cos y)$

6) $a=|y\sin x|+\sqrt{x^2\cos(y+x)}$; $b=\operatorname{tg}\left(\dfrac{x}{y^2+1}\right)$

7) $a=\left|y^2\operatorname{tg}x\right|+\sqrt{1+x^2+y^2}$; $b=|y|\cdot|x|$.

8) $a=(y+x)\sin(x\sqrt{y})$; $b=\cos(x^2+y+1)$.

9) $a=\sqrt{1+y^2}\cdot\operatorname{tg}\left(\dfrac{x^2}{1+|y|}\right)$; $b=\sqrt{1+y^2e^{-xy}}$

10) $a=\sqrt{1+x^2}\cdot\operatorname{ctg}\left(\dfrac{x\cos y}{1+x^2}\right)$; $b=\sqrt{1+xe^{-y^2}}$

Sadə dövri strukturlu və budaqlanan strukturlu proqramlar

52. **n** natural ədədi verilib. Aşağıdakıları hesablamalı:

1) `1+3+5+...+2n-1;` 2) `1*2*3*...*n;`
3) $1/1^2+1/2^2+...+1/n^2$; 4) `1+1/2+...+1/n;`
5) `(1+1/1^2)*(1+1/2^2)...(1+1/n^2);`
6) `1/sin(1)+1/(sin(1)+sin(2))+...+1/(sin(1)+...+sin(n));`

53. **a** həqiqi ədədi və **n** natural ədədi verilib. Aşağıdakıları hesablamalı:

1) `a*(a+1)*...*(a+n-1);` 2) $a*(a-n)*(a-2n)*...*(a-n^2)$
3) `1/a+1/(2*a)+...+1/(n*a);` 4) $1/a+1/a^2+...+1/a^n$
5) `1/a+1/(a*(a+1))+...+1/(a*(a+1)*...*(a+n))`.

54. a,b $(a<b)$ həqiqi ədədləri və n natural ədədi verilib. $y=f(x)$ funksiyası $[a,b]$ parçasında təyin edilib. Arqumentin $x_i=a+ih$ $(i=0,1,\ldots,n)$ $(h=(b-a)/n)$ qiymətləri üçün funksiyanın $y_i=f(x_i)$ qiymətlərini tapmalı:

1) $y=x^2+1$; 2) $y=x^2+2x$; 3) $y=\cos(x)$; 4) $y=(x-2)/(x^2+3)$
5) $y=x^2e^x$; 6) $y=1-x^2(2)$; 7) $y=x\sin(2x)$; 8) $y=x^2/(1+x)$

55. Funtlarla 1 funtdan 10 funta qədər (1 *funt* addımı ilə) verilmiş çəki vahidi ilə kiloqram çəki vahidi arasında uyğunluğu göstərən cədvəli qurmalı (1 *funt*=400 *qram*).

56. Dyumlarla 1 dyumdan 10 dyuma qədər (1 *dyum* addımı ilə) verilmiş məsafə vahidi ilə santimetr məsafə vahidi arasında uyğunluğu göstərən cədvəli qurmalı (1 *dyum*=2,54 *sm*).

57. Millərlə 5 mildən 75 miləddək (5 *mil* addımı ilə) verilmiş məsafə vahidi ilə kilometr məsafə vahidi arasında uyğunluğu göstərən cədvəli qurmalı (1 *mil*=1,609 *km*).

58. n natural ədədi verilib. Aşağıdakıları hesablamalı:

1) $((1+2)/(1+3))*((2+2)/(2+3))\ldots((n+2)/(n+3))$;
2) $1/(2*1+1)^2+1/(2*2+1)^2+\ldots+1/(2*n+1)^2$;
3) $(1/1^3)*(1/2^3)*\ldots*(1/n^3)$;
4) $(1^2/(1^2+2*1+3))*(2^2/(2^2+2*2+3))*\ldots*(n^2/(n^2+2n+3))$;
5) $((1+1)/(1+2))+((2+1)/(2+2))+\ldots+((n+1)/(n+2))$;
6) $(1-1/2^2)+(1-1/3^2)+\ldots+(1-1/n^2)$.

59. Aşağıdakı hasili hesablamalı:
$(1+\sin(0.1))*(1+\sin(0.2))*\ldots*(1+\sin(10))$

60. x həqiqi ədədi verilib. Hesablamalı:
$((x-2)*(x-4)*(x-6)\ldots(x-64))/((x-1)*(x-3)\ldots(x-63))$

61. A həqiqi ədədi verilib. Tapmalı:
1) $1;\ 1+1/2;\ 1+1/2+1/3;\ldots$ ədədləri arasında A-dan böyük birinci həddi;
2) $1+1/2+\ldots+1/n>A$ şərtini ödəyən birinci n qiymətini.

n natural ədədi verilib. Hesablamalı:

62. $1/(2*1)^2+1/(2*2)^2+\ldots+1/(2*n)^2$;
63. $1/1!+1/2!+\ldots+1/n!$;
64. $(2+1/1!)*(2+1/2!)*\ldots*(2+1/n!)$;
65. $(1+1)/1!+(2+1)/2!+\ldots+(n+1)/n!$;
66. $(1+1/1!)^2*(1+1/2!)^2*\ldots*(1+1/n!)^2$

n natural ədədi verilib. Birinci n vuruğun hasilini tapmalı:

67. $(1/2)*(3/4)*(5/6)*\ldots$;
68. $(1/1)*(3/2)*(5/3)*\ldots$.

n natural ədədi verilib. Hesablamalı:

69. $1/(1+1)+1/(3+1)+\ldots+1/((2n-1)+1)$;
70. $1/(2*1)^3+1/(2*4)^3+\ldots+1/(2*(3n-2))^3$;
71. $(1/(1*(1+1)))*(1/(2*(2+1)))*\ldots*(1/(n*(n+1)))$;
72. $(1^2/(1+1))*(3^2/(3+1))*\ldots*((2n-1)^2/((2n-1)+1))$.

n natural ədədi və x həqiqi ədədi verilib. Hesablamalı:

73. $x/1!+x^2/2!+\ldots+x^n/n!$;

74. $(1/1!+\sqrt{x})*(1/2!+\sqrt{x})*\ldots*(1/n!+\sqrt{x})$;

75. $(x+\cos(x))/2+(x+\cos(2x))/2^2+\ldots+(x+\cos(nx))/2^n$;

76. $(1+\sin(x)/1!)*(1+\sin(2x)/2!)*\ldots*(1+\sin(nx)/n!)$

n natural ədədi verilib. Hesablamalı:

77. $(-1/(2*1+1)*1)+(1/(2*2+1)*2)+\ldots+((-1)^n/(2*n+1)*n)$;

78. $(1/(1*(1+1)))-(1/(2*(2+1)))+\ldots+((-1)^{n+1}/(n*(n+1)))$;

79. $(-(1+1)/1!)+((2+1)/2!)+\ldots+((-1)^n(n+1)/n!)$;

80. $(1/(2*1+1)^2)+(1/(2*2+1)^2)+\ldots+(1/(2*n+1)^2)$

81. x həqiqi ədədi və n natural ədədi verilib. Hesablamalı:
$(1+(1-x)/1!)+(2+(1-x)^2/2!)+\ldots+(n+(1-x)^n/n!)$

82. n natural ədədi verilib. Hesablamalı:
$1/1+2/(1+1/2)+\ldots+n/(1+1/2+\ldots+1/n)$

83. n natural ədədi verilib. Hesablamalı:

1) $-1/3+1/5+\ldots+(-1)^n/(2n+1)$;

2) $1/(1*2)-1/(2*3)+\ldots+(-1)^{n+1}/(n*(n+1))$;

3) $-2/1!+3/2!+\ldots+(-1)^n*(n+1)/n!$

4) $-1/1!+2/2!+\ldots+(-1)^n*n/n!$

84. n natural ədədi və x həqiqi ədədi verilib. Hesablamalı:

1) $1!/1^2+2!/2^2+\ldots+n!/2^n$;

2) $1!/3-2!/5+\ldots+(-1)^{n+1}n!/(2n+1)$;

3) $(1/(x-1)-|x|)*(1/(x^2-1)-|x|^2)*\ldots*(1/(x^n-1)-|x|^n)$;

4) $(1/1!+\sin(x))*(1/2!+\sin(2x))*\ldots*(1/n!+\sin(nx))$

85. n natural ədədi verilib. Hesablamalı:

1) $1/2*n+1/2*(n-1)+\ldots+1/2$;

2) $(2n/(n+2))*((2n-1)/((n-1)+2))*\ldots*(2/(1+2))$;

3) $((n+1)/(n+2))+(((n-1)+1)/((n-1)+2))+\ldots+2/3$;

4) $(2n/(n^2-2))*((2(n-1))/((n^2-1)-2))*\ldots*2/(1^2-2)$

n natural ədədi verilib. Hesablamalı:

86. $(1/1)^1+(1/2)^2+\ldots+(1/n)^n$;

87. $(1/2)^2*(1/4)^4*\ldots*(1/2n)^{2n}$;

88. $(1/3)^1+(1/5)^2+\ldots+(1/(2n+1))^n$

89. x həqiqi ədədi və n natural ədədi verilib. Hesablamalı:
$x+x^3/3+x^5/5+\ldots+x^{2n-1}/(2n-1)$

90. n natural ədədi verilib. Hesablamalı:

1) $\sum_{i=1}^{n}\frac{2i^2+1}{2i^3}$; 2) $\sum_{i=1}^{n}\frac{(5i)^{2i}}{i^2+1}$; 3) $\sum_{i=1}^{n}\frac{\sqrt{i+2}}{3i^3-1}$;

4) $\sum_{i=1}^{n}\frac{\sin(i^2)}{i^2}$; 5) $\sum_{i=1}^{n}\frac{i^2}{i^4+1}$; 6) $\sum_{i=1}^{n}\frac{3^i}{i}$.

91. n natural ədədi və x həqiqi ədədi verilib. Hesablamalı:

1) $\sum_{i=1}^{n}\frac{x^i}{2i+2}$; 2) $\sum_{i=1}^{n}\frac{(i^2)^2}{2x^2+2}$; 3) $\sum_{i=1}^{n}\frac{(2i^2+2)^3}{x^i+2x}$;

4) $\sum_{i=1}^{n}\frac{i^3+3}{2x^i+3}$; 5) $\sum_{i=1}^{n}\frac{x^{2i}}{2i+3}$; 6) $\sum_{i=1}^{n}\frac{2i+1}{i\cdot x+2}$.

92. n natural ədədi verilib. Hesablamalı:

1) $\prod_{i=1}^{n}\frac{\cos(2i)}{2^i}$; 2) $\prod_{i=1}^{n}\frac{\ln(i)+2^i}{2i+5}$; 3) $\prod_{i=1}^{n}\frac{\sin(3i)}{2i^2+1}$;

4) $\prod_{i=1}^{n}\frac{i^{2i}+1}{i^3+2}$; 5) $\prod_{i=1}^{n}\frac{\sin(2i+1)}{2i^2-1}$; 6) $\prod_{i=1}^{n}\frac{5i^3-3}{(i^2)^i}$.

Massivlərin iştirakı ilə sadə proqramlar

93. n-ölçülü A vektoru verilib, harada n tam ədəddir. Hesablamalı:

1) $a_1+\ldots+a_n$; 2) $a_1*\ldots*a_n$; 3) $|a_1|+\ldots+|a_n|$;

4) $a^2_1+\ldots+a^2_n$; 5) $|a_1*\ldots*a_n|$; 6) $a_1-a_2+\ldots+(-1)^{n+1}a_n$

94. n-ölçülü A vektorundaki elementlərin ədədi ortasını tapmalı.

95. n-ölçülü A və B vektorları verilib. Hesablamalı: (harada C n-ölçülü vektordur)

1) A+B=C; 2) A-B=C; 3) A*B=C

96. n-ölçülü X və Y vektorları verilib. Onların skalyar hasilini $(x,y)=x_1*y_1+\ldots+x_n*y_n$ tapmalı.

97. n-ölçülü X vektoru verilib. Vektorun uzunluğunu $L=\sqrt{x_1^2+\ldots+x_n^2}$ tapmalı.

98. n-ölçülü A və B vektorları verilib. Bu vektorların nisbətini tapıb n-ölçülü X vektorunda verməli.

99. n-ölçülü A vektoru verilib. Bu massivin elementlərinin cəmini tapmalı və A massivinin hər bir elementini bu qiymətə bölməli, nəticələri hər hansı n-ölçülü B vektorunda verməli.

100. n-ölçülü A vektoru verilib. Bu massivin elementlərinin ədədi ortasını tapmalı və A massivinin hər bir elementini bu qiymətə vurmalı. Nəticələri n-ölçülü B vektorunda verməli.

101. n-ölçülü X vektoru verilib. $y=(x^2-2)/(2x)$ funksiyasının X massivində verilmiş arqumentin qiymətləri üçün uyğun qiymətlərini tapmalı və Y massivində verməli.

102. n-ölçülü A vektoru və k həqiqi ədədi verilib. A massivinin elementləri ilə k ədədinin hasillərini tapıb, elə həmin A massivində verməli.

103. n-ölçülü A vektoru verilib. Hesablamalı:

1) $a_n^3+a^3_{n-1}+\ldots+a_1^3$; 2) $(|a_n|/n)+(|a_{n-1}|/(n-1))+\ldots+|a_1|$;

3) $|a_n|*|a_{n-1}|*\ldots*|a_1|$; 4) $a_n-a_{n-1}+\ldots+(-1)^{n+1}a_1$

104. n-ölçülü A vektoru verilib. Aşağıdakıları hesablayıb, nəticələri B massivində yerləşdirməli.

1) $a_1, a_1+a_2, \ldots, a_1+\ldots+a_n$; 2) $a_1*a_1, a_1*a_2, \ldots, a_1*a_n$

105. Ardıcıllıq aşağıdakı qayda üzrə qurulur: $x_1=1, x_i=i*x_{i-1}+1/i$, $(i=2,3,\ldots,n)$. Ardıcıllığın $x_2,\ldots,x_n$ həddlərini tapmalı.

106. Ardıcıllıq aşağıdakı qayda üzrə qurulub: $x_1=0.1$; $x_2=5$; $x_i=x_{i-1}/2+3x_{i-2}$ $(i=3,4,\ldots,n)$. Ardıcıllığın $x_3,\ldots,x_n$ hədlərini tapmalı.

107. n natural ədədi və həqiqi a,b ədədləri verilib. $y_1,y_2,\ldots,y_n$ qiymətlərini tapmalı, harada $y_i=a+ih$, $h=(b-a)/n$.

108. n sayda sətri və m sayda sütunu olan (n,m - tam ədədlərdir) A matrisi verilib. Matris elementlərinin cəmini tapmalı.

109. n×m ölçülü A matrisi verilib. Matris elementlərinin hasilini tapmalı.

110. n×n ölçülü A və B matrisləri verilib. Bu matrislərin cəmini və fərqini tapmalı və nəticələri həmin ölçülü bir C matrisində yerləşdirməli.

111. n×n ölçülü A matrisi verilib. Matrisin izini, yəni baş diaqonal elementlərinin cəmini tapmalı.

112. n×n ölçülü A matrisi verilib. Matrisi transponirə edib, nəticəni həmin ölçülü B matrisində yerləşdirməli.

113. n×m ölçülü A matrisinin l nömrəli (1<l<n) sətrinin elementlərinin cəmini tapmalı.

114. n×m ölçülü A matrisinin k nömrəli (1<k<m) sütununun elementlərinin hasilini tapmalı.

115. n×n ölçülü A matrisi və k tam ədədi (1<k<n) verilib. k nömrəli sətrdə yerləşən elementləri, bu sətrdə olan baş diaqonal elementinə vurub, nəticələri n-ölçülü B vektorunda yerləşdirməli

116. n×n ölçülü A matrisi verilib. Matrisin k nömrəli sətr elementləri ilə l nömrəli sütun elementlərinin cəmini tapıb, nəticəni n-ölçülü B vektorunda yerləşdirməli.

117. n×n ölçülü A matrisi verilib. Matrisin l nömrəli sütun elementlərini bu sütundakı baş diaqonal elementinə bölüb, nəticələri n-ölçülü B vektorunda verməli.

118. n×n ölçülü A matrisi verilib. Matrisin k nömrəli sətr elementləri ilə l nömrəli sütun elementlərinin fərqini, hasilini və nisbətini tapıb, nəticələri uyğun olaraq n-ölçülü B,C,D vektorlarında yerləşdirməli.

119. n×n ölçülü A matrisi verilib. Matrisin l nömrəli sətrindən, A(l,k)/A(k,k) nisbətinə vurulmuş k-cı sətrini çıxmalı (1<k,l<n).

120. n×n ölçülü A matrisi verilib. Matrisin baş diaqonal elemenlərinin əks istiqamətdə, yəni n-ci həddən birinci hədd istiqamətində cəmləməli.

121. n×m ölçülü A matrisi verilib. Matrisin hər bir sətrindəki elementlərin cəmini tapıb, nəticələri n-ölçülü B vektorunda yerləşdirilməli, yəni

$$B_i=\sum_{j=1}^{m}A_{ij},\quad i=\overline{1,n}\ .$$

122. n×m ölçülü A matrisi və m-ölçülü B vektoru verilib. Matrisin vektora hasilini tapıb, nəticəni n-ölçülü C vektorunda verməli, yəni

$$C_i = \sum_{j=1}^{m} A_{ij} \cdot B_j, \quad i = \overline{1,n}$$

123. n-ölçülü A vektoru verilib. Vektorda elementlər artım ardıcıllığı ilə düzülüb. Bu ardıcıllığı pozmadan vektora hər hansı B elementini daxil etməli.
124. n-ölçülü A vektoru verilib. Vektorun k-cı və 1-ci mövqelərində yerləşən elementlərinin yerlərini dəyişmək lazımdır.
125. n×m ölçülü A matrisi verilib. Matrisin k-cı sətri ilə 1-ci sətrinin yerlərini dəyişdirməli.
126. n×m ölçülü A matrisi verilib. Matrisin modulca maksimal elementini tapıb, bu elementin olduğu sətri m-ölçülü B vektorunda yerləşdirib, çapa verməli.
127. n×m ölçülü A matrisi verilib. Matrisin minimal elementini tapıb, bu elementin olduğu sütunu n-ölçülü B vektorunda yerləşdirib, çapa verməli.
128. n-ölçülü A və m-ölçülü B vektorları verilib. B vektorunu A vektorunun k-cı və k+1-ci elementləri arasına yerləşdirməklə, n+m ölçülü C vektorunu qurmalı.
129. n×n ölçülü A matrisi verilib. Matrisin maksimal elementi yerləşən sətirlə, sütunun yerlərini dəyişdirib, yeni matrisi çapa verməli.
130. n×n ölçülü A matrisi verilib. Matrisin minimal elementinin yerləşdiyi sətirlə, sütunu matrisidən çıxarıb, alınan yeni matrisi çapa verməli.
131. n×m ölçülü A matrisi verilib. Matrisin hər bir sətrinin maksimal elementini tapıb, nəticəni, n-ölçülü B vektorunda çıxarmalı.
132. n×m ölçülü A matrisi verilib.Matrisin hər bir sütuunun minimal elementini tapıb, nəticəni m-ölçülü B vektorunda çıxarmalı.

Mürəkkəb dövr strukturlu proqramlar. Müəyyən dəqiqliklə hesablamalar

133. Ardıcıllıq $a_1=0$; $a_2=1$; $a_i=a_{i-2}+a_{i-1}/2^{i-1}$; (i=3,4,...) qaydası üzrə qurulur. $a_1*a_2*...*a_n$ hasilini tapmalı (n-natural ədəddir).
134. Ardıcıllıqlar $a_1=1$; $b_1=1$; $a_i=1/2*(\sqrt{b_{i-1}+1}/(2*\sqrt{a_{i-1}}))$, $b_i=2a^2_{i-1}+b_{i-1}$, (i=2,3,...) qaydası üzrə qurulur. $a_1*b_1+...+a_n*b_n$ cəmini tapmalı (n – natural ədəddir).
135. Ardıcıllıqlar $x_1=1$; $y_1=1$; $x_i=0.3x_{i-1}$; $y_i=x_{i-1}+y_{i-1}$; (i=2,3,...) qaydası ilə qurulur. $(x_1/(1+|y_1|))+...+(x_n/(1+|y_n|))$ cəmini tapmalı. (n-natural ədəddir).
136. Ardıcıllıqlar $a_1=1$, $b_1=1$, $a_i=3b_{i-1}+2a_{i-1}$; $b_i=2a_{i-1}+b_{i-1}$ (i=2,3,...) qaydası ilə qurulur. $(1/(1+a_1^2+b^2_1)*1!)+...+(1/(1+a^2_n+b_n^2)*n!)$ cəmini tapmalı (n – natural ədəddir).
137. Həqiqi x və y ədədləri və n natural ədədi verilib. Ardıcıllıqlar $a_1=x$; $b_1=y$; $a_i=2b_i-1+a_i-1$; $b_i=2a^2_{i-1}+b_{i-1}$; (i=2,3,...) qaydası üzrə qurulub. $(a_1*b_1)/1!+...+(a_n*b_n)/n!$ cəmini tapmalı.
138. n natural ədədi verilib. $b_1,...,b_n$ ardıcıllığını almalı, harada i=1,...,n qiymətləri üçün b_i aşağıdakı qiymətləri alır: 1) i; 2) i^2; 3) i!; 4) 2^{i+1}

139. $(3*a+4)/a^2-5a-9$ ifadəsinin qiymətlərini $a=1,2,\ldots,100$ qiymətləri üçün hesablamalı.

140. $x^5-9x^4+1.7x^2-9$ çoxhədlisinin qiymətlərini $x=1,2,\ldots,10$ qiymətləri üçün hesablamalı.

141. a_1, a_2, a_3, a_4, $x_1,\ldots,x_{50}$ həqiqi ədədləri verilib. $b_1,\ldots,b_{50}$ ardıcıllığını almalı, harada

$$b_i=((x_i^2-x_i-a_1)/(x_i-a_1))*((x_i^3-x_i-a_2)/(x_i-a_2))*(x_i-a_3)-$$
$$-((x_i^4-x_i+a_4)/x_i)+x_i*(x_i-a_3);\quad (i=1,2,\ldots,50).$$

142. Fibonaççi ədədləri ardıcıllığı $u_1=0;\ u_2=1;\ u_i=u_{i-1}+u_{i-2}\ (i=3,4,\ldots)$ qanunu üzrə qurulur. $n>1$ natural ədədi verilib. $u_1,\ldots,u_n$ ardıcıllığını almalı.

143. $P_1(x)=x;\ P_2(x)=(3x^2-1)/2;\ P_3(x)=(5x^2-3x)/2$ funksiyalarının, $x=1,2,\ldots,20$ arqumentləri üçün qiymətlərini hesablamalı.

144. n natural ədədi verilib. $y=(x^2-3x^2+2)/(2x^3-1)$ funksiyasının $x=1,1.1,\ldots,1+0,1*n$ qiymətləri üçün qiymətlərini hesablamalı.

145. n natural ədədi və $a,h,b,d_1,\ldots,d_n$ həqiqi ədədləri verilib. İfadənin qiymətini hesablayın:

$$d_1(b-a)+d_2(b-a)*(b-(a-h))+\ldots+d_n(b-a)*(b-(a-h))*\ldots*$$
$$*(b-a-((n-1)h))$$

146. n natural ədədi və $x_1,\ldots,x_n$ $(n>2)$ həqiqi ədədləri verilib. Hesablamalı:

$$(1/(|x_1|+1)+x_2)*(1/(|x_2|+1)+x_3)*\ldots*(1/(|x_{n-1}|+1)+x_n)$$

147. n natural ədədi və $x_1,\ldots,x_n$ həqiqi ədədləri verilib. Hesablamalı:

$$(x_1+2x_2+x_3)*(x_2+2x_3+x_4)*\ldots*(x_{n-2}+2x_{n-1}+x_n)$$

148. n natural ədədi və $x_1,\ldots,x_n$ həqiqi ədədləri verilib. Hesablamalı:

$$(x_1+x_2+x_3)x_2*(x_2+x_3+x_4)x_3+\ldots+(x_{n-2}+x_{n-1}+x_n)x_{n-1}$$

149. n natural ədədi və a,b $(b>a>0)$ həqiqi ədədləri verilib. $y_1,\ldots,y_n$ həqiqi ədədlər ardıcıllığını almalı, harada $y_i=x_i^2;\ x_i=a+ih,\ h=(b-a)/n$

150. n natural ədədi və $a_1,\ldots,a_n$ tam ədədləri verilib. $a_1,\ldots,a_n$ ardıcıllığının hədlərini, onların kvadratlarının n ədədinə bölünməsindən sonra qalan qalıq hədləri ilə əvəz etməli.

151. Müsbət, həqiqi a,x,eps ədədləri verilib. $y_1,y_2,y_3,\ldots$ ardıcıllığı $y_1=a$, $y_i=1/2(y_{i-1}+x/y_{i-1})$, $i=2,3,\ldots$ qanunu üzrə qurulub. Bu ardıcıllıqda $|y^2_n-y^2_{n-1}|<eps$ şərtini ödəyən ilk y_n həddini tapmalı.

152. Ardıcıllıq $x_1=1;\ x_i=(2-x^3_{i-1})/5;\ i=2,3,\ldots$ qanunu üzrə qurulub. $|x_n-x_{n-1}|<0.001$ şərtini ödəyən ilk x_n həddini tapmalı.

153. Ardıcıllıq $y_1=0;\ y_i=(y_{i-1}+1)/(y_{i-1}+2);\ i=2,3,\ldots$ qanunu üzrə qurulub. $|y_n-y_{n-1}|<0.0001$ şərtini ödəyən birinci y_n həddini tapmalı.

154. Həqiqi x, eps ədədləri verilib. Aşağıdakı sonsuz cəmi eps dəqiqliyi ilə hesablamalı: $1/x^3+1/(x^3*2^2)+1/(x^3*3^2)+\ldots$

155. Həqiqi x, eps ədədləri verilib. Sonsuz cəmi eps dəqiqliyi ilə hesablamalı: $x^2+x^2/2+x^2/3+\ldots$

Həqiqi x, eps ədədləri verilib. Aşağıdakı sonsuz cəmləri eps dəqiqliyi ilə hesablamalı:

156. $-x/1+x^2/2-x^3/3+\ldots$;

157. $x/1!+x/2!+x/3!+\ldots$;

158. $1/(x^2+1)+1/(x^2+2^3)+1/(x^2+3^3)+\ldots$;

159. **eps** həqiqi ədədi verilib. Aşağıdakı sonsuz cəmləri **eps** dəqiqliyi ilə hesablamalı:
1) $1+1/2^2+1/3^2+\ldots$; 2) $1/2+1/(2*(2+1))+1/(3(3+1))+\ldots$;

160. **eps** həqiqi ədədi verilib. Aşağıdakı sonsuz cəmləri **eps** dəqiqliyi ilə hesablamalı:
1) $-1/1!+1/2!-\ldots$; 2) $-2/1!+2/2!-2/3!+\ldots$;

161. **x** həqiqi ədədi verilib. $a_1, a_2, \ldots$ ardıcıllığı $a_n=x^n/n!$ qanunu üzrə qurulur $a_1+\ldots+a_k$ cəmini tapmalı, harada **k**, $k>10$ və $|a_{k+1}|<0.0001$ şərtlərini ödəyən tam ədəddir.

162. **x** həqiqi ədədi verilib $a_1, a_2, \ldots$ ardıcıllığı $a_n=x^{2n}/(n*(n+1)*(n+2))$ qanunu üzrə qurulur. $a_1+\ldots+a_k$ cəmini tapmalı, harada **k**, $k>10$ və $|a_{k+1}|<0.0001$ şərtini ödəyən tam ədəddir.

163. **eps** həqiqi ədədi verilib. $a_1, a_2, \ldots$ ardıcıllığı $a_n=n/(2n^2+1)$ qanunu üzrə qurulur $|a_n-a_{n-1}|<eps$ şərtini ödəyən birinci a_n həddini ($n>2$) tapmalı.

164. **eps** həqiqi ədədi verilib. $a_1, a_2, \ldots$ ardıcıllığı
$$a_n=(1-1/2!)*(1+1/3!)\ldots(1+(-1)^n/(n+1)!)$$
qanunu üzrə qurulur. $|a_n-a_{n-1}|<eps$ şərtini ödəyən birinci a_n həddini ($n>2$) tapmalı.
Hesablamalı:

165. $\sum_{i=1}^{10}\prod_{k=1}^{12}\frac{k}{k^2+i^2}$;

166. $\sum_{i=1}^{10}\prod_{k=1}^{10}\frac{i^2}{k+i^2}$;

167. $\sum_{k=1}^{10}\prod_{m=1}^{10}\frac{k+m}{k+m+2}$;

168. $\prod_{i=1}^{10}\sum_{k=1}^{10}\frac{k+1}{k^2+i}$;

169. $\prod_{i=1}^{10}\sum_{k=1}^{10}\frac{i+k}{i^2+k}$;

170. $\prod_{i=1}^{10}\sum_{k=1}^{20}\frac{k+i}{k^2+i^2}$;

171. $\sum_{i=1}^{10}\sum_{j=1}^{15}\frac{i}{i^2+j^2}$;

172. $\sum_{i=1}^{10}\sum_{j=1}^{15}\frac{i^2}{j+i^2}$;

173. $\sum_{i=1}^{10}\sum_{j=1}^{15}\frac{i+j}{i+j+2}$;

174. $\sum_{i=1}^{10}\sum_{j=1}^{15}\frac{j^2}{j^2+1}$;

175. $\sum_{i=1}^{10}\sum_{j=1}^{20}\frac{j^2}{i^2+1}$;

176. $\sum_{i=1}^{15}\sum_{j=1}^{10}\frac{j+i}{j^2+1}$;

177. $\prod_{i=1}^{10}\prod_{j=1}^{10}\frac{i}{i+j}$;

178. $\prod_{i=1}^{10}\prod_{j=1}^{10}\frac{j+i}{i+j+1}$;

179. $\prod_{i=1}^{10}\prod_{j=1}^{10}\frac{j+i^2}{j^2+i+1}$;

180. $\sum_{i=1}^{10}\sum_{k=1}^{i}\frac{k}{k+i}$;

181. $\sum_{i=1}^{10}\sum_{k=1}^{i}\frac{k^2}{k^2+i}$;

182. $\sum_{i=1}^{10}\sum_{k=i+1}^{i+5}\frac{k}{k^2+i^2}$;

183. $\sum_{i=1}^{10}\prod_{k=1}^{i}\frac{k}{k+i}$;

184. $\sum_{i=1}^{10}\prod_{k=1}^{i}\frac{k^2}{k^2+i}$;

185. $\sum_{i=1}^{10}\prod_{k=i+1}^{i+5}\frac{k}{k^2+i^2}$;

186. $\prod_{i=1}^{10}\sum_{k=1}^{i}\frac{k}{k+i}$;

187. $\prod_{i=1}^{10}\sum_{k=1}^{i}\frac{k^2}{k^2+i}$;

188. $\prod_{i=1}^{10}\prod_{k=1}^{i}\frac{k}{k+i}$;

189. $\prod_{i=1}^{10}\prod_{k=1}^{i}\frac{k^2}{k^2+i}$.

190. $\sum_{i=1}^{\infty}\frac{i^2}{i^4+1}$;

191. $\sum_{i=1}^{\infty}\frac{(-1)^i}{i^2+1}$;

`eps=0.001` dəqiqliyi ilə hesablamalı:

192. $\sum_{i=1}^{\infty}\frac{2^i}{i!}$;

193. $\sum_{i=1}^{\infty}\frac{2^i}{i^i}$;

194. $\sum_{i=1}^{\infty}\frac{i}{i^3+1}$;

195. $\sum_{i=1}^{\infty}\frac{(-1)^i}{i!}$;

196. $\sum_{i=1}^{\infty}\frac{i^2+1}{i^4+i^2+1}$;

197. $\sum_{i=1}^{\infty}\frac{2i-1}{i^2+1}$;

198. $\sum_{i=1}^{\infty}\frac{i^3}{e^i}$;

199. $\sum_{i=1}^{\infty}\frac{\ln(i+1)}{i^2+1}$;

200. $\sum_{i=1}^{\infty}e^{-i}\ln(i+1)$;

201. $\sum_{i=1}^{\infty}ie^{-i}$;

202. $\sum_{i=1}^{\infty}\frac{i}{\ln^3(i+1)}$;

203. $\sum_{i=1}^{\infty}i^{-2}e^{-i}$;

204. $\sum_{i=1}^{\infty}\frac{i}{e^{i^2}}$;

205. $\sum_{i=1}^{\infty}\frac{i^2+2i+1}{i^4+2i^2+1}$;

206. $\sum_{i=1}^{\infty}\frac{\sin i}{i}$;

207. $\sum_{i=1}^{\infty}\sin ie^{-i}$;

208. $\sum_{i=1}^{\infty}\cos ie^{-i}$;

209. $\sum_{i=1}^{\infty}\frac{i\sin i}{i^2+1}$;

210. $\sum_{i=1}^{\infty}ie^{-i^2}$;

211. $\sum_{i=1}^{\infty}\frac{\cos i}{i^2}$;

212. $\sum_{i=1}^{\infty}\frac{i\cos i}{i^2+1}$;

213. $\sum_{i=1}^{\infty}\frac{1}{i^3}$;

214. $\sum_{i=1}^{\infty}\frac{i\sin i+i^2\cos i}{i^3+1}$;

Dövr və budaqlanmanın uzlaşması. Tam ədədlər

215. **n** natural ədədi və $a_1,\dots,a_n$ həqiqi ədədləri verilib. Bu həqiqi ədədlər daxilində ən böyüyünü tapmalı.

216. **n** natural ədədi və $a_1,\dots,a_n$ həqiqi ədədləri verilib. Tək indeksli ədədlərin ən kiçiyini tapmalı.

217. **n** natural ədədi və $a_1,\dots,a_n$ həqiqi ədədləri verilib. Cüt indeksli ədədlərin ən

böyüyünü tapmalı.

218. n natural ədədi və $a_1, \ldots, a_n$ həqiqi ədədləri verilib. Müsbət ədədlərin cəmini və sayını tapmalı.

219. n natural ədədi və $a_1, \ldots, a_n$ həqiqi ədədləri verilib. Mənfi ədədlərin kvadratları cəmini tapmalı.

220. n natural ədədi və $a_1, \ldots, a_n$ həqiqi ədədləri verilib. $a_1, \ldots, a_n$ ardıcıllığında müsbət ədədləri bir vahid artırmalı, mənfi ədədləri isə 0.1 ədədi ilə əvəz etməli.

221. n natural ədədi və $a_1, \ldots, a_n$ həqiqi ədədləri verilib. $a_1, \ldots, a_n$ ardıcıllığında ikidən kiçik bütün ədədləri sıfırla əvəz etməli.

222. n natural ədədi və $a_1, \ldots, a_n$ həqiqi ədədləri verilib. Ardıcıllıqdakı mənfi ədədlərin sayını və müsbət ədədlərin hasilini tapmalı.

223. n natural ədədi və $a, x_1, \ldots, x_n$ tam ədədləri verilib. Əgər $x_1, \ldots, x_n$ ardıcıllığında a-ya bərabər heç olmasa bir element varsa, onda ardıcıllığın bu cür həddindən sonra gələn elementlərinin cəmini tapmalı.

224. Tam $a, n, x_1, \ldots, x_n$ ədədləri verilib. $x_1, \ldots, x_n$ ardıcıllığında a-ya bərabər olan həddin sıra nömrəsini təyin etməli, belə bir element yoxdursa, sıfır çap olunmalı.

225. n natural ədədi və $x_1, \ldots, x_n$ tam ədədləri verilib. $x_1, \ldots, x_n$ ardıcıllığında müsbət və ya mənfi ədədlərin çoxluq təşkil etdiyini təyin etməli.

226. n natural ədədi və $x_1, \ldots, x_n$ tam ədədləri verilib. Ardıcıllığın ən böyük həddinin mütləq qiymətcə vahiddən böyük olub-olmadığını təyin etməli.

227. n tam ədədi verilib. Təyin etməli, bu ədəd cüt və ya tək ədəddir.

228. $n, a_1, \ldots, a_n$ tam ədədləri verilib. $a_1, \ldots, a_n$ ardıcıllığında neçə cüt ədəd olduğunu təyin etməli.

229. $n, a_1, \ldots, a_n$ tam ədədləri verilib. Ardıcıllıqdakı tək ədədlərin cəmini tapmalı.

230. $n, a_1, \ldots, a_n$ tam ədədləri verilib. Ardıcıllığın cüt ədədlərinin ən böyüyünü tapmalı.

231. $n, a_1, \ldots, a_n$ tam ədədləri verilib. Ardıcıllığın tək ədədlərinin ən kiçiyini tapmalı.

232. n tam ədədi verilib. Ədəddəki rəqəmlərin sayını və cəmini tapmalı.

233. n tam ədədi verilib. Təyin etməli n ədədi 3-ə tam bölünürmü (ədəddəki rəqəmlərin cəmi 3-ə bölünürsə, ədəd də 3-ə bölünür).

234. n tam ədədi verilib. Təyin etməli n ədədi 9-a tam bölünürmü (ədəddəki rəqəmlərin cəmi 9-a bölünürsə, ədəd də 9-a bölünür).

235. n tam ədədi verilib. Ədədin bir də daxil olmaqla bütün bölənlərini tapmalı.

236. m tam ədədi verilib. m ədədindən kiçik bütün mükəmməl ədədləri, yəni öz bölənlərinin cəminə bərabər olan ədədləri tapmalı.

237. m tam ədədi verilib. Birdən m-ə qədər bütün sadə ədədləri tapmalı.

238. n tam ədədi verilib. Bu ədədin bütün sadə bölənlərini tapmalı.

239. k və l tam ədədləri verilib. Əgər bu ədədlər bir-birindən fərqlidirsə, onların hər ikisini, bu ədədlərdən ən böyüyü ilə əvəz etməli, bu ədədlər bərabərdirsə, onları sıfırla əvəz etməli.

240. **n** və **m** tam ədədləri verilib. Hansı ədəddə rəqəmlərin sayının çox olduğunu təyin etməli.
241. **n** və **m** tam ədədləri verilib. Hansı ədədin rəqəmlərinin cəminin böyük olduğunu təyin etməli.
242. $n, a_1, \ldots, a_n$ tam ədədləri verilib. Ardıcıllığın tək ədədlərindən ibarət ardıcıllıq qurmalı.
243. $n, a_1, \ldots, a_n$ tam ədədləri verilib. Ardıcıllıqda tək və ya cüt ədədlərin çox olduğunu təyin etməli.
244. **n** tam ədədi verilib. **A** və **B** massivləri aşağıdakı qayda üzrə qurulur. Əgər **i** tək ədəddirsə $a_i=i$, əks halda $a_i=i/2$ və əgər **i**- tək ədəddirsə, $b_i=i^2$, əks halda $b_i=i^2+2$. Hesablamalı: $(a_1-b_1)^2+\ldots+(a_n-b_n)^2$.
245. $n, a_1, \ldots, a_n$ tam ədədləri verilib. **B** massivinin elementləri ardıcıllığın elementlərindən aşağıdakı kimi qurulur: Əgər a_i-nin 3 qiymətinə nisbəti 1 qalığını verirsə, $b_i=a_i^2$, əks halda $b_i=1/a_i^2$. Hesablamalı: alınan **B** massivinin elementlərinin cəmini.
246. $n, a_1, \ldots, a_n$ tam ədədləri verilib. Bu ədədlərdən 3-ə qalıqsız bölünənlərin sayını və cəmini tapmalı.

Massivlərin iştirakı ilə məsələlərin həlli

247. **n** natural ədədi və **n** sayda həqiqi ədədlərdən ibarət **A** massivi verilib. **n** ölçülü **B** vektorunu qurmalı, harada **B** massivi **A** massivindən aşağıdakı qayda üzrə asılıdır: $b_i=a_1$, $b_n=a_n$, $b_i=(a_{i+1}-a_i)/3$, $i=2,\ldots,n-1$.
248. **n** natural ədədi və **n** ölçülü **A**, **B** vektorları verilib. Hesablamalı: $(a_1+b_n)*(a_2+b_{n-1})*\ldots*(a_n+b_1)$.
249. **n** natural ədədi və **n** ölçülü **A** vektoru verilib. Həmin ölçülü **B** massivini aşağıdakı qayda ilə qurmalı: Əgər $a_i>0$ olarsa, $b_i=1/a_i$ əks halda $b_i=2*a_i$ qəbul etməli ($i=1,\ldots,n$). Sonra aldığımız **B** massivi elementlərinin cəmini tapmalı.
250. **n** natural ədədi və **n** ölçülü **A** vektoru verilib. **N** ölçülü **B** massivi aşağıdakı qayda üzrə qurulur: əgər $a_i>0$ olarsa $b_i=a_i+1$, əks halda $b_i=a_i^2$ ($i=1,\ldots,n$) qəbul etməli. Alınan **B** massivi elementlərinin hasilini tapmalı.
251. **n** natural ədədi və **n** ölçülü **A** vektoru verilib. Bu massivin elementlərindən $B(i,j)=A(i)-3*A(j)$ qaydası üzrə ($i,j=1,\ldots,n$) $n\times n$ ölçülü **B** matrisini almalı.
252. **n** natural ədədi və **n** ölçülü **A** vektoru verilib. **A** massivinin elemenlərindən aşağıdakı qayda üzrə həmin ölçülü **B** massivini qurmalı: əgər $0<a_i<10$ olarsa $b_i=a_i$ əks halda $b_i=-a_i$ ($i=1,\ldots,n$) qəbul etməli.
253. **n** natural ədədi və **n** ölçülü **A**,**B** vektorları verilib. Bu massiv elementlərindən bir element **A**-dan, bir element **B**-dən götürmək şərti ilə **2n** ölçülü **C** vektorunu qurmalı.
254. **n** natural ədədi və **n** ölçülü **A**, **B** vektorları verilib. Bu massiv elemenlərindən $C(i,j)=A(j)/(|B(i)|)$ qaydası üzrə ($i,j=1,\ldots,n$) $n\times n$ ölçülü **C** matrisini qurmalı.

255. n natural ədədi verilib. A(i,j)=i+2*j qaydası üzrə n×n ölçülü A matrisini qurmalı (i,j=1,…,n).

256. n natural ədədi verilib. A(i,j)=1/(i+j) qaydası üzrə n×n ölçülü A matrisini qurmalı (i,j=1,…,n).

257. n×n ölçülü A matrisi verilib. Matrisin maksimal elementini və bu elementin durduğu sətr və sütunun nömrəsini təyin etməli.

258. n×k ölçülü A və k×m ölçülü B matrisləri verilib (n,k,m-tam ədədlərdir). Matrislərin hasilini tapıb, nəticəni n×m ölçülü C matrisində yerləşdirməli.

259. n×n ölçülü A matrisi verilib. Matrisin müsbət elementlərin sayını və cəmini tapmalı.

260. n×m ölçülü A matrisi verilib. Matrisin 2-dən böyük və 5-dən kiçik bütün elementlərinin sayını və hasilini tapmalı.

261. n×m ölçülü A matrisi verilib. Matrisin ən kiçik elementinin yerləşdiyi sətr elementlərinin cəmini tapmalı.

262. n×m ölçülü A matrisi verilib. Matrisin ən böyük elementinin yerləşdiyi sütun elementlərinin hasilini tapmalı.

263. n ölçülü A vektoru verilib. Bu massiv elementlərindən n×n ölçülü B matrisini aşağıdakı qayda üzrə qurmalı: B(i,j)=A(i)/2*A(j) (i,j=1,…,n).

264. n×n ölçülü A matrisini aşağıdakı qayda üzrə qurmalı: əgər i<j olarsa $A(i,j)=i^2+j^2$, əks halda A(i,j)=1/i+j, (i,j=1,…,n).

265. n×n ölçülü A matrisini aşağıdakı qayda üzrə qurmalı: əgər i>j olarsa A(i,j)=sin(i+j), əks halda A(i,j)=cos(i+j), (i,j=1,…,n).

266. n×n ölçülü A matrisi verilib. Matrisin elementlərindən aşağıdakı qayda üzrə həmin ölçülü B matrisini qurmalı: əgər i>j olarsa B(i,j)=A(i,j), əks halda B(i,j)=1/A(i,j), (i,j=1,…,n).

267. n×n ölçülü A matrisi verilib. Matrisin elementlərindən aşağıdakı qayda üzrə həmin ölçülü B matrisini qurmalı: əgər i<j olarsa, $B(i,j)=A(i,j)^2$, əks halda B(i,j)=1-A(i,j) (i,j=1,…,n).

268. n×n ölçülü A matrisi verilib. Matrisin elementlərindən aşağıdakı qayda üzrə həqiqi ölçülü B və C matrislərini qurmalı: əgər j>i olarsa B(i,j)=A(i,j), əks halda B(i,j)=A(j,i) və əgər j<i olarsa C(i,j)=A(i,j), əks halda C(i,j)=-A(i,j), (i,j=1,…,n).

269. n×n ölçülü A matrisi verilib. Matrisin sətr elementlərinin cəmlərindən ibarət n ölçülü B vektorunu qurmalı.

270. n×n ölçülü A matrisi verilib. Matrisin sətr elementlərinin hasillərindən ibarət n ölçülü B vektorunu qurmalı.

271. n ölçülü X vektoru verilib. Bu massiv elementlərindən aşağıdakı qayda üzrə qurulan (əgər X(i)<2 olarsa Y(i)=X(i), əks halda Y(i)=0,5) Y massiv elementlərinin ən böyüyünü tapmalı.

272. n ölçülü X vektoru verilib. Massiv elementlərindən (əgər X(i)<1 olarsa, Y(i)=X(i), əks halda Y(i)=2) qaydası üzrə qurulan Y massivi elementlərinin ən kiçiyini tapmalı.

273. n ölçülü X vektoru verilib. Massivin elementlərindən (əgər 0<X(i)<10 olarsa, Y(i)=X(i), əks halda Y(i)=1) qaydası üzrə qurulan Y massivi elementlərinin cəmini tapmalı.
274. n ölçülü X vektoru verilib. Massivin elementlərindən (əgər X(i)<1 olarsa, Y(i)=X(i), əks halda Y(i)=1/X(i)) qaydası ilə qurulan Y massivi elementlərinin mütləq qiymətcə hasilini tapmalı.
275. n×n ölçülü A matrisi verilib. Matrisin baş diaqonal elementlərinə soldan paralel olan əlavə baş diaqonal elementlərinin cəmini tapmalı.
276. n×n ölçülü A matrisi verilib. Matrisin baş diaqonal elementlərinə sağdan paralel olan əlavə baş diaqonal elementlərinin hasilini tapmalı.
277. n ölçülü A vektoru verilib. Massivin müsbət elementlərindən ibarət B massivini qurmalı.
278. n ölçülü A vektoru və P həqiqi ədədi verilib. Təyin etməli, A massivində P ədədi varmı?
279. n ölçülü A vektoru verilib. Massivin k indeksli (1<k<n) elementini çıxarmalı.
280. n ölçülü A vektoru və B tam ədədi verilib. Massivin k indeksli mövqeyinə (1<k<n) B ədədini daxil etməli.
281. n×m ölçülü A matrisi verilib. Matrisin k nömrəli sətrini çıxarmalı. (1<k<n).
282. n×m ölçülü A matrisi verilib. Matrisin l nömrəli (1<l<m) sütununu çıxarmalı.
283. n×m ölçülü A matrisi və m ölçülü B vektoru verilib. B massivini A matrisinin k nömrəli (1<k<n) sətri kimi matrisə daxil etməli.
284. n×m ölçülü A matrisi və n ölçülü C vektoru verilib. C massivini A matrisinin l nömrəli (1<k<n) sütunu kimi matrisə daxil etməli.
285. n×m ölçülü A matrisi verilib. Matrisin k və l nömrəli (1<k,l<n) sətrlərinin yerlərini bir-birilə əvəz etməli.
286. n×m ölçülü A matrisi verilib. Matrisin k və l nömrəli (1<k,l<n) sütunlarının yerlərini bir-birilə əvəz etməli.
287. n ölçülü A vektoru verilib. Massivin elementlərini onların artımı sırası boyunca düzməli.
288. n ölçülü A massivi verilib. Təyin etməli, massivin elementləri artım sırası boyunca düzülmüşdürmü?
289. n×n ölçülü A matrisi verilib. Matrisin ən kiçik və ən böyük elementlərindən hansı mütləq qiymətcə böyükdür?
290. n×n ölçülü A matrisi verilib. Matrisin ən kiçik elementinin mütləq qiyməti, ya matrisin mənfi elementlərinin cəmi böyükdür?
291. n ölçülü A massivi və n×n ölçülü B matrisi verilib. Matrisin cüt indeksli elementləri arasında A massivinin elementləri ilə üst-üstə düşən elementləri sıfırla əvəz etməli.
292. n ölçülü A massivi və n×n ölçülü B matrisi verilib. Matrisin 5-ci və 6-cı sütunları arasına A massivini sütun kimi əlavə edib alınan n×(n+1) ölçülü matrisi çap etməli.
293. n×n ölçülü A matrisi verilib. Matrisin birinci və sonuncu, ikinci və

sonuncudan əvvəlki və s. sutunlarını bir-biri ilə əvəz edib, yeni matrisi çapa verməli.

294. n×n ölçülü A matrisi verilib. Matrisin birinci və sonuncu, ikinci və sonuncudan əvvəlki və s. sətirlərini bir-biri ilə əvəz edib, yeni matrisi çapa verməli.

295. n×n ölçülü A matrisi verilib. Matrisin n nömrəli sətri ilə n nömrəli sütununun yerlərini dəyişdirib, yeni matrisi çapa verməli.

296. n×n ölçülü A matrisi verilib. Matrisin baş diaqonal elementlərindən sağda qalan elementlərindən ən böyüyünü tapmalı.

297. n×n ölçülü A matrisi verilib. Matrisin baş diaqonal elementlərindən solda qalan elementlərindən ən kiçiyini tapmalı.

Alt proqramlar. Fayllar

298. n və m tam ədədləri verilib. Hesablamalı: `(n!+m!)/(n-m)!`

299. n və m tam ədədləri verilib. Hesablamalı: `(n!*m!)/(n+m)!`

300. Hesablamalı: (n -tam ədəddir):
1) `3!; 4!;…; n!;` 2) `1/5!; 1/6!;…;1/n!`
3) `6!+6; 7!+7;…; n!+n.`

301. İki üçbucaq tərəflərinin qiymətləri ilə verilib. Onların sahələrini Heron düsturu ilə tapmalı.

302. İki kvadrat tənlik verilib. Onların köklərini tapmalı.

303. x,y həqiqi ədədləri verilib. Hesablamalı: $F(x,2y,3)+F(5,x^2,y-x)$, harada $F(a,b,c)=(2*a-b-\sin(c))/(1+a*b*c)$.

304. x,y,z həqiqi ədədləri verilib. Hesablamalı:
$F(x+1,x*y,z)-F(x/y,y-2,z+3)$,
harada $F(a,b,c)=(a^2+b^2)/(2a-c^3)$.

305. x,y həqiqi ədədləri verilib. Hesablamalı:
$F(x,y)+F(1/x,2y)+F(1-x^2,xy)$,
harada əgər $a+b<0$ olarsa $F(a,b)=2a-b$, əks halda a^2+b^2.

306. x,y həqiqi ədədləri verilib. Hesablamalı:
$F(\sin(x),y)+F(\cos(x),1/x+y)-F(x^2,y^2-1)$, harada əgər $a>0$ və $b<0$ olarsa $F(a,b)=a/b$, əks halda $F(a,b)=\sin(a*b)$.

307. x,y,z həqiqi ədədləri verilib. Hesablamalı:
$(\max(x,x+y)+\max(x,y+z))/(1+\max(x*y,z))$.

308. x,y,z həqiqi ədədləri verilib. Hesablamalı:
$(\min(x^2,y)-\min(y,z))/(1-\min(x+y,y+z))$.

309. n və m natural ədədləri verilib. Birdən n-ə və birdən m-ə qədər natural ədədlərin cəmini tapmalı.

310. n və m natural ədədləri verilib. Birdən n-ə qədər və n-dən m-ə qədər ($n<m$) natural ədədlərin kvadratları cəmini tapmalı.

311. n və m natural ədədləri verilib. n-dən birə qədər və m-dən birə qədər natural ədədlərin hasilini tapmalı.

312. n ölçülü A və m ölçülü B vektorları verilib. Onların elementlərinin cəmini tapmalı.

313. n ölçülü A və B vektorları verilib. Onların elementlərinin ədədi ortasını tapmalı.
314. n ölçülü A və B vektorları verilib. Onların müsbət elementlərinin cəmini və sayını tapmalı.
315. n ölçülü A və B vektorları verilib. Onların cüt indeksli elementlərinin hasilini tapmalı.
316. n ölçülü A və m ölçülü B vektorları verilib. Onların ən böyük elementlərini təyin etməli.
317. n ölçülü A və B vektorları verilib. Onların mənfi elementlərinin mütləq qiymətlərinin cəmini tapmalı.
318. Aşağıdakı y=f(x) funksiyalarının verilmiş [1,2] parçasında n=24 üçün (h=(b-a)/n) düzbucaqlılar düsturu ilə müəyyən inteqrallarını hesablamalı:
1) $y=1/(1+x)$; 2) $y=1/(1+x^2)$; 3) $y=1/(e^x+1)$;
4) $y=x/(1+x)$; 5) $y=\sin(x)/x$; 6) $y=1/x^2$;
7) $y=\sin(x^2)$; 8) $y=2x/(1+x^2)$; 9) $y=x(x+1)$.
319. Aşağıdakı funksiyaların, verilmiş [1,2] parçasında eps=0.001 dəqiqliyi ilə trapeslər düsturu ilə müəyyən inteqrallarını hesablamalı:
1) $y=1/\sqrt{x}$; 2) $y=e^x$; 3) $y=\cos(x^2)$;
4) $y=\cos(x)/x$; 5) $y=1/(1+x^3)$; 6) $y=x*\sin(x)$;
7) $y=1/(1-x+x^2)$; 8) $y=\sin(x)/(x^2+1)$; 9) $y=1/(x^2+1)$.
320. Aşağıdakı funksiyaların, verilmiş [1,2] parçasında eps=0.001 dəqiqliyi ilə Simpson düsturu ilə müəyyən inteqrallarını hesablamalı:
1) $y=1/\sqrt{1+x^4}$; 2) $y=1/(1+\sin(x))$; 3) $y=\cos(x)/(1+x)$;
4) $y=(x^4+1)/(x^6+1)$; 5) $y=(x+1)*\sin(x)$; 6) $y=\ln(x)$;
7) $y=x/(1+\sin(x))$; 8) $y=x/\sqrt{1+x}$; 9) $y=2x^2+4$.
321. Aşağıdakı qeyri-xətti tənlikləri parçanı yarıya bölmə üsulu ilə eps=0.001 dəqiqliyi ilə həll etməli:
1) $x^3+x^2-3=0$; 2) $x^3-3x^2+9x-8=0$; 3) $x^3-6x-8=0$;
4) $x^3-3x^2+6x+3=0$; 5) $x^3+3x+1=0$; 6) $x^3+4x-6=0$
7) $x^3-3x^2+9x+2=0$; 8) $x^3+x-5=0$; 9) $x^3-0,1x^2+0,4x-1,5=0$;
322. Aşağıdakı qeyri-xətti tənlikləri iterasiya üsulu ilə eps=0,001 dəqiqliyi ilə həll etməli:
1) $2x^2+4x-1=0$; 2) $2x^3-3x^2-12x-5=0$; 3) $x^3-3x^2+3=0$;
4) $2x^3+9x^2-21=0$; 5) $x^3+3x^2-2=0$; 6) $2x^3+9x^2-10=0$;
7) $x^3-3x^2-3=0$; 8) $x^3-12x-5=0$; 9) $x^3+3x^2-1=0$
323. Aşağıdakı qeyri-xətti tənlikləri kəsənlər üsulu ilə eps=0.001 dəqiqliyi ilə həll etməli:
1) $2x^3+9x^2-6=0$; 2) $x^3-3x^2-24x-3=0$; 3) $x^3-12x+6=0$;
4) $2x^3-3x^2-12x+10=0$; 5) $x^3-12x+10=0$; 6) $2x^3-3x^2-12x+1=0$;
7) $x^3-4x^2+2=0$; 8) $2x^3-3x^2-12x+8$; 9) $x^3-12x-10=0$.
324. Aşağıdakı qeyri-xətti tənlikləri toxunanlar üsulu ilə eps=0.001 dəqiqliyi ilə həll etməli:
1) $x^3+3x^2-1=0$; 2) $x^3+2x^2+2=0$; 3) $x^3-2x+2=0$;
4) $x^3-3x^2+9x-10=0$; 5) $x^3+3x-1=0$; 6) $x^3+x-3=0$;
7) $x^3+4x-6=0$; 8) $x^3+3x+1=0$.

325. Eyler üsulunu tətbiq etməklə, aşağıdakı diferensial tənlikləri `[a,b]` parçasında `h` addımı ilə həll etməli.

1) $y'=y+3x^2*e^x$, `y(0)=0, a=0, b=1, h=0.1;`
2) $y'=1/2*(x*y)$, `y(0)=1, a=0, b=1, h=0.1;`
3) $y'=x^2+y^2$, `y(0)=0, a=0, b=1, h=0.1;`
4) $y'=1+xy^2$, `y(0)=0, b=1, h=0.1;`
5) $y'=y/(x+1)-y^2$, `y(0)=1, a=0, b=1, h=0.1;`
6) $y'=(x+y)/(y-x)$, `y(0)=1, a=0, b=1, h=0.1;`
7) $y'=y+(1+x)y^2$, `y(1)=-1, a=1, b=1.5, h=0.1;`
8) $y'=y-(2x)/y$, `y(0)=1, a=0, b=0, h=0.1.`

326. Runqe-Kutta üsulunu tətbiq etməklə, aşağıdakı diferensial tənlikləri `[a,b]` parçasında `h` addımı ilə həll etməli;

1) $y'=2x-y$, `y(0)=1, a=0, b=1, h=0.1;`
2) $y'=x^3+y^3$, `y(0)=0, a=0, b=1, h=0.1;`
3) $y'=2x-3y$, `y(0)=0, a=0, b=1, h=0.1;`
4) $y'=2x*y$, `y(0)=1, a=0, b=1, h=0.1;`
5) $y'=(x*y)/(1-x^2)$, `y(0)=1, a=0, b=1, h=0.1;`
6) $y'=x-y$, `y(0)=0, a=0, b=1, h=0.1;`
7) $y'=2x-\sin(y/x)$, `y(0)=1, a=0, b=1, h=0.1;`
8) $y'=(x*y)/12$, `y(0)=2, a=0, b=1, h=0.1`

327. Həqiqi ədədlərdən ibarət `F` faylı verilib. Bu fayl elementlərinin cəmini tapmalı.

328. Həqiqi ədədlərdən ibarət `F` faylı verilib. Bu fayl elementlərinin hasilini tapmalı.

329. Həqiqi ədədlərdən ibarət `F` faylı verilib. Bu fayl elementlərindən ən böyüyünü tapmalı.

330. Həqiqi ədədlərdən ibarət `F` faylı verilib. Bu fayl elementlərinin ədədi ortasını tapmalı.

331. Tam ədədlərdən ibarət `F` faylı verilib. Bu fayldakı cüt ədədlərin sayını tapmalı.

332. Tam ədədlərdən ibarət `F` faylı verilib. Bu fayldakı tək ədədlərin cəmini tapmalı.

333. `n` natural ədədi verilib. `G` faylına $b_i=i$ `(i=1,...,n)` qaydası üzrə tapılan $b_1,\dots,b_n$ tam ədədlərini yazmalı.

334. `n` natural ədədi verilib. `G` faylına, $b_i=i!$ `(i=1,...,n)` qaydası üzrə tapılan $b_1,\dots,b_n$ tam ədədlərini yazmalı.

335. `n` natural ədədi verilib. `G` faylına $b_i=1/i!$ `(i=1,...,n)` qaydası üzrə tapılan $b_1,\dots,b_n$ həqiqi ədələrini yazmalı.

336. Qrupdakı tələbələrin soyadlarından ibarət `F` faylı qurmalı.

337. Azərbaycan şairlərinin soyadlarından ibarət `F` faylı qurmalı.

338. Fakultədə tədris olunan fənn adlarından ibarət `F` faylı qurmalı.

339. Müasir alqoritmik dillərin adlarından ibarət `F` faylı qurmalı.

340. Tam ədədlərdən ibarət `F` faylı verilib. Faylın cüt ədədlərini `G` faylına, tək ədədlərini isə `H` faylına yazmalı.

341. Tam ədədlərdən ibarət **F** və **G** faylları verilib. **F** və **G** fayllarının elementlərini ardıcıl olaraq **H** faylına yazmalı.

342. Tarixi şəxsiyyətlərin adlarından ibarət **F** faylı qurmalı.

343. **F** simvol faylı verilib. Faylın təkrarını **G** faylında almalı.

344. **F1** və **F2** simvol faylları verilib. **F1** faylını **F2**-yə, **F2** faylını isə **F1**-ə köçürməli (**G** - köməkçi faylından istifadə etməklə).

345. Həqiqi **a** və **b** ədədləri verilib. Tapmalı: $f(a,5b,3.14)+f(2,a,a+b)$, burada $f(x,y,z)=(3x-2y-\cos z)/(1+|z|)$.

346. Həqiqi **a** və **b** ədədləri verilib. Tapmalı: $f(a,2b)+f(2a,b)-f(a+b,a-b)$, burada $f(x,y)=(x^2+y^2+1)/(x^2+2xy+2y^2+4)$.

347. Həqiqi **a** ədədi verilib. Tapmalı: $(3\cdot f(1/2)+2f(1+a))/(5+f(1-a^2))$, burada $f(x)=\left(\sum_{k=0}^{5}\frac{x^{2k+1}}{(2k+1)!}\right)\Big/\left(\sum_{k=0}^{5}\frac{x^{2k}}{(2k)!}\right)$.

348. Həqiqi **a**, **b**, **c** ədədləri verilib. Tapmalı:

$$\frac{\max(a+b,b+c,a+c)+\max(abc,a^2b,ac^2)}{1+\max(a^2+b^2,1+c^2,1+a^2)}.$$

349. Həqiqi **a**, **b**, **c** ədədləri verilib. Tapmalı:

$$\frac{\min(a+b+c,b+2c,a+2b)+\min(2a+b,2b+c)}{1+\min^2(ab,ac,ac)}.$$

350. Həqiqi **a**, **b**, **c** ədədləri verilib. Tapmalı: $f(a+b,b+c)\cdot f(a+c,b-c)$, burada $f(x,y)=x/(1+y^2)+y/(1+x^2)-(x-y)^2$.

351. Həqiqi **a**, **b**, **c** ədədləri verilib. Tapmalı: $f(1,a,b)-f(a,b,c)\cdot f(c,1,b)$, burada $f(x,y,z)=\dfrac{\max(x,y,z)\cdot\min(x,y,z)}{\max^2(x,y,z)+\min^2(x,y,z)}$.

352. **n** natural ədədi verilib. **n**-dən kiçik bütün ədədlər içərisində, iki ədədin kvadratları cəmi şəklində göstərilə bilənləri ayırmalı. Tam kvadratların qurulması üçün proseduru təyin etməli.

353. Həqiqi $x_1,y_1,x_2,y_2,\ldots,x_{10},y_{10}$ ədədləri verilib. Təpə nöqtələri (x_1,y_1), $(x_2,y_2),\ldots,(x_{10},y_{10})$ olan onbucaqlının perimetrini tapmalı. Koordinatları ilə verilmiş nöqtələr arasındakı məsafəni təyin etmək üçün prosedur təyin etməli.

354. $(x_1,y_1),(x_2,y_2),\ldots,(x_{10},y_{10})$ nöqtələrinin müstəvidə koordinatları olan $x_1,y_1,x_2,y_2,\ldots,x_{10},y_{10}$ həqiqi ədədləri verilib. Bu nöqtələr içərisində ən kiçik perimetrli üçbucağı təyin edən nöqtələrin koordinatlarını tapmalı. Üçbucağın perimetrini hesablayan proseduru qurmalı.

355. $(x_1,y_1),(x_2,y_2),...,(x_{10},y_{10})$ nöqtələrinin müstəvidə koordinatları olan $x_1,y_1,x_2,y_2,...,x_{10},y_{10}$ həqiqi ədədləri verilib. Bu nöqtələr içərisində ən böyük sahəli üçbucağı təyin edən nöqtələrin koordinatlarını tapmalı. Üçbucağın sahəsini hesablayan prosedur qurmalı.

356. n natural ədədi və $x_1,y_1,x_2,y_2,...,x_n,y_n$ həqiqi ədədləri verilib. Təpə nöqtələri $(x_1,y_1),(x_2,y_2),...,(x_n,y_n)$ olan n-bucaqlının sahəsini tapmalı.

357. n natural ədədi verilib. n,n+1,2n ədədləri arasında əkiz ədədlərin, yəni aralarındakı fərq ikiyə bərabər sadə ədədlər olub-olmadığını təyin etməli.

358. Sadə ədədləri təyin edən proseduru qurmalı.

359. Ədədin onluq say sistemindəki yazılışı olan simvollar sətri ilə verilmiş tam ədədin qiymətini hesablayan prosedur qurmalı.

360. Verilən simvol hərf olduqda doğru qiyməti, əks halda isə yalan qiyməti alan prosedur qurmalı.

361. Verilən simvol hərf olmadıqda, verilmiş uyğun hərfi alan prosedur qurmalı.

362. Verilmiş simvolun simvollar sətrində sağdan ən birinci iştirak mövqeyini təyin edən prosedur qurmalı. Əgər sətr bu simvola malik deyilsə, prosedur -1 qiymətini verməli.

363. Verilmiş simvollar ardıcıllığında sıfırları vahidlərlə, vahidləri isə sıfırlarla əvəz edən prosedur qurmalı.

364. Verilmiş sətrdən, verilmiş ikinci sətrə də aid olan simvolları çıxaran prosedur qurmalı.

365. Verilmiş simvollar ardıcıllığında olan saitlərin sayını hesablayan prosedur qurmalı.

366. Verilmiş simvollar ardıcıllığında olan (.,:;) işarələrinin sayını hesablayan prosedur qurmalı.

367. Verilmiş simvollar ardıcıllığında olan boş yerlərin sayını hesablayan prosedur qurmalı.

368. Elementləri həqiqi ədədlər olan fayl verilib. Fayl elementlərinin kvadratları cəmini tapmalı.

369. Elementləri həqiqi ədədlər olan fayl verilib. Fayl elementlərinin cəminin modulunu tapmalı.

370. Elementləri tam ədədlər olan fayl verilib. Fayl elementlərindən müsbət olanlarının sayını tapmalı.

371. Elementləri tam ədədlər olan fayl verilib. Faylın mənfi elementlərinin sayını tapmalı.

372. Elementləri tam ədədlər olan fayl verilib. Faylın sıfır elementlərinin sayını tapmalı.

373. Elementləri tam ədədlər olan fayl verilib. Faylın ən kiçik elementini tapmalı.

374. Elementləri tam ədədlər olan fayl verilib. Faylın mənfi elementləri içərisində ən böyüyünü tapmalı.

375. Elementləri tam ədədlər olan fayl verilib. Faylın cüt nömrəli elementlərinin ən kiçiyini tapmalı.

376. Elementləri tam ədədlər olan fayl verilib. Faylın üçə tam bölünən nömrəli elementlərinin ən böyüyünü tapmalı.

377. Elementləri tam ədədlər olan fayl verilib. Faylın cüt nömrəli elementlərindən modulca ən böyüyünü tapmalı.

378. Elementləri tam ədədlər olan fayl verilib. Faylın ən böyük və ən kiçik elementlərinin cəmini tapmalı.
379. Elementləri tam ədədlər olan fayl verilib. Faylın birinci və sonuncu elementlərinin cəmini tapmalı.
380. Elementləri tam ədədlər olan fayl verilib. Faylın tək elementlərinin sayını tapmalı.
381. Elementləri tam ədədlər olan fayl verilib. Faylın dördə tam bölünən elementlərinin sayını tapmalı.
382. Elementləri tam ədədlər olan fayl verilib. Faylın beşə tam bölünən elementlərinin sayını tapmalı.
383. Elementləri tam ədədlər olan fayl verilib. Faylın tək elementlərinin sayını tapmalı.
384. Elementləri tam ədədlər olan **F** faylı verilib. **F** faylının tək elementlərini **G** faylında almalı.

Qrafik imkanlar

385. Təpə nöqtələri (100,100), (150,100), (80,170) olan üçbucaq qurmalı.
386. Təpə nöqtələri (80,80), (170,80), (170,150), (80,150) olan düzbucaqlı qurmalı.
387. Təpə nöqtələri (100,100), (150,100), (170,120), (150,140), (100,140) olan beşbucaqlı qurmalı.
388. Təpə nöqtələri (120,100), (140,120), (140,140), (120,160), (100,140), (100,120) olan altıbucaqlı qurmalı.
389. Mərkəzi ekranın mərkəzi ilə üst-üstə düşən, tərəfləri ekranın koordinat oxlarına paralel və 30-a bərabər olan kvadrat qurmalı.
390. Mərkəzi ekranın mərkəzi ilə üst-üstə düşən, tərəfləri ekranın koordinat oxlarına paralel, 30 və 50-yə bərabər olan düzbucaqlı qurmalı.
391. Mərkəzi ekranın mərkəzi ilə üst-üstə düşən və radiusu 40-a bərabər dairə qurmalı.
392. Mərkəzi ekranın mərkəzi ilə üst-üstə düşən, radiusu 30-a, hündürlüyünün eninə olan nisbəti 1/3 olan ellips qurmalı.
393. Ekranın mərkəzində, tərəfləri uyğun olaraq 10,20,...,100 olan bir-birinin daxilində verilmiş 10 kvadrat qurmalı.
394. Ekranın mərkəzində, radiusları uyğun olaraq 5,10,15,.....,50 olan bir-birinin daxilində verilmiş 10 çevrə qurmalı.
395. Üçbucaq və düzbucaqlılardan istifadə etməklə ev şəkli qurmalı.
396. Üçbucaqlılardan və düzbucaqlıdan istifadə etməklə ağac şəkli qurmalı.
397. Çevrə və düz xəttlərdən istifadə etməklə saat şəkli qurmalı.
398. Ellips, çevrə, düz xətt və düzbucaqlılardan istifadə etməklə balıq şəkli qurmalı.
399. İki üçbucağın kəsişməsi ilə alınan altıgüşəli ulduz qurmalı.
400. İki kvadratın kəsişməsi ilə alınan səkkizgüşəli ulduz qurmalı.
401. Aşağıdakı funksiyaların qrafikini qurmalı;

```
1) y=3x^2;    2) y=-6x^2+3x;    3) y=sin x;
4) y=1/x^2;   5) y=2x+2;        6) y=(x+3)/(x-2).
```

402. Üçbucaq və xətlərin vasitəsilə gəmi şəkli qurmalı və onun müxtəlif hissələrini ayrı-ayrı rənglərlə rəngləməli.

403. Düz xətlərdən istifadə etməklə, tovuz quşuna bənzəyən şəkil qurmalı.

404. Düz xətlərdən istifadə etməklə, şəbəkəyə bənzəyən şəkil qurmalı.

405. Koordinatları təsadüfi seçilən nöqtələrdən ibarət fiqurlar qurmalı.

406. Düz xətt və çevrələrdən istifadə etməklə avtomobil şəkli qurmalı.

407. Düz xətt və çevrələrdən istifadə etməklə mümkün musiqi alətlərinin şəklini qurmalı.

408. Tərəfi 50 olan kvadrat qurmalı. Kvadratın mərkəzi ekranın mərkəzi ilə üst-üstə düşməli, tərəfləri isə ekranın tərəflərinə paralel olmalıdır.

409. Tərəfləri 40 və 60 olan düzbucaqlı qurmalı, Düzbucaqlının mərkəzi ekranın mərkəzi ilə üst-üstə düşməli, tərəfləri isə ekranın tərəflərinə paralel olmalıdır.

410. Diaqonalları 60 və 80 olan romb qurmalı. Rombun mərkəzi ekranın mərkəzi ilə üst-üstə düşməli, tərəfləri isə ekranın tərəflərinə paralel olmalıdır.

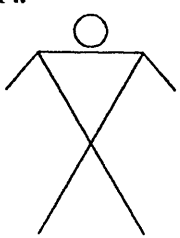
411. Ekranda çevrənin mərkəzini və radiusunu təyin edən üç tam ədəd verilib. Əgər çevrə ekranın mərkəzindən keçən üfüqi düz xətlə kəsişmirsə, onda bu çevrəni və çevrəyə həmin düz xəttə nəzərən simmetrik olan çevrəni qurmalı.

412. Ekranda çevrənin mərkəzini və radiusunu təyin edən üç tam ədəd verilib. Əgər çevrə ekranın mərkəzindən keçən şaquli düz xətlə kəsişmirsə, onda bu çevrəni və çevrəyə həmin düz xəttə nəzərən simmetrik olan çevrəni qurmalı.

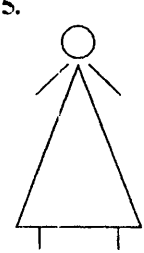
413. Dörd tam ədəd parçanın ekrandakı vəziyyətini təyin edir. Bu parçanı və bu parçaya ekranın mərkəzi nöqtəsinə nəzərən simmetrik olan parçanı qurmalı.

Aşağıdakı şəkilləri nöqtə, parça və çevrələrdən istifadə etməklə qurmalı.

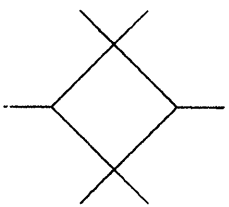
414.



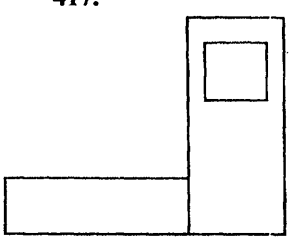
415.



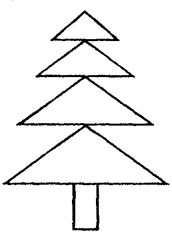
416.



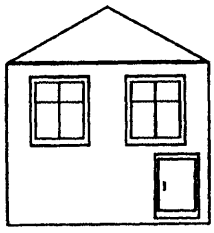
417.



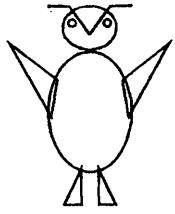
418.



419.



420.



421. Ekranda iki düzbucaqlı qurmalı. Onlardan biri üfüqi xətlərlə, digəri isə şaquli xətlərlə ştrixlənməlidir.
422. Doqquz konsentrik çevrə qurub, onları növbə ilə ardıcıl olaraq yaşıl və qırmızı rənglərlə boyamalı.
423. Bir-birinin daxilində verilən 10 kvadrat qurub, onları növbə ilə ardıcıl olaraq yaşıl və qırmızı rənglərlə boyamalı.
424. Ekranın sol kənarından başlayaraq, ekranın sağ kənarınadək hərəkət edən qırmızı rəngli düzbucaqlı quran proqram tərtib etməli.
425. Qırmızı rəngli ayparanı quran proqram tərtib etməli.
426. Ekranın mərkəzində bir-birinin daxilində verilmiş doqquz kvadrat qurub, onları üç müxtəlif rənglə boyamalı.
427. Ekranın mərkəzində yaranıb, tədricən ölçülərini üç dəfə artırıb, sonra isə əvvəlki ölçülərinə qayıdan yaşıl rəngli dairə quran proqram tərtib etməli.
428. Müxtəlif rənglərlə boyanmış iki dördbucaqlının kəsişməsini qurub, onların fırlanmasını təmin edən proqram tərtib etməli.
429. Ekranın yuxarı sol kənarından başlayaraq, ekranın sağ aşağı kənarınadək hərəkət edən qırmızı rəngli kvadrat quran proqram tərtib etməli.
430. Svetoforun işinin modelini quran proqram tərtib etməli.
431. Yanıb-sönən şüaları olan günəş şəklini quran proqram tərtib etməli.

Dilin sadə konstruksiyaları. Sadə proqramlar

432. Aşağıdakı düsturları Pascal dilində yazmalı.

1) $(1+x)^2+\sqrt{y}$; 2) $|ax+by+c|$; 3) $a*b/c+a*b$;
4) $\cos(x)/(1+x^3)$; 5) $|x|-y/1+|x*y|$; 7) $1+x+x^2/2$;
6) $x^2+y^2/(1-x^2+y^2/2)$; 8) $e^x-1/\cos(x)-\sqrt{1+x^3}$

433. Aşağıdakıları Pascal dilində yazmalı:

1) $(b+\sqrt{b^2+4ac})/2a-a^3c+b^{-2}$;
2) $(\sin(x)+\cos(y))/(\cos(x)-\sin(y))$;
3) $(x+y)/(y+1)-(xy-12)/(34+x)$;
4) $(3+e^{y-1})/(1+x^2|y-tg(x)|)$;
5) $x-x^3/3+x^5/5$;
6) $|x^2x^3|-7x/(x^3-15x)$;
7) $e^x-x-2+(1+x)^2$;
8) $x-10\sin(x)+|x^4-x^5|$.

434. Aşağıdakı ifadələrin qiymətini tapmalı:

1) `trunc(6.9);` 2) `round(6.9);` 3) `trunc(3.2);`
4) `round (3.2);` 5) `trunc(-1.8);` 6) `round(-1.8);`
7) `20 div 6;` 8) `20 mod 6;` 9) `23 div 4;`
10) `17 mod 3;` 11) `52 div 7;` 12) `36 mod 12.`

435. Aşağıdakı ifadələrin hansı tipə tam və ya həqiqi tipə aid olduğunu təyin etməli:

1) `1.5;` 2) `20/4;` 3) `sqr(4);`
4) `sqrt(5);` 5) `round(7.5);` 6) `trunc(-3.14).`

436. Əgər `y` həqiqi, `n` isə tam tipli dəyişəndirsə, onda aşağıdakı mənimsətmə operatorlarından hansı doğru, hansı isə düzgün deyildir.

1) `y:=n+1;`
2) `n:=y-1;`
3) `n:=4.0;`
4) `y:=trunc(y);`
5) `n:=n div 2;`
6) `y:=y div 2;`
7) `n:=n/2;`
8) `n:=sqr(sqrt(n)).`

437. Aşağıdakı mənimsətmə operatorlarında hesabi əməllər hansı ardıcıllıqla yerinə yetiriləcək və əgər `A =1,2; B=10; C=4` olarsa, `z` hansı qiyməti alar;

```
1) z:=-A*B/C*A/B*C;
2) z:=A*A/(A+B)*C/(C+A)+(A-B)/C;
3) z:=A*SQR(B)*C+A/B-A;
4) z:=A*EXP(3*SQRT(C))-B/(B-1);
5) z:=(((SQRT(C)*A)*B+1)+0.3)-C*A/(A*B).
```

438. `x` həqiqi ədədi verilib. Aşağıdakı düsturların hesablanması üçün proqramlar qurmalı:

1) $F=1-2x+3x^2-4x^3$; 2) $F=1+2x+3x^2+4x^3$;
3) $F=2x^4-3x^3+4x^2-5x+6$; 4) $F=(x-1)^2+3x^3-8x^2$.

439. Aşağıdakı proqramların nəticələrini təyin etməli.

```
1) program k1 ;
   var a,b:integer;
   begin a:=10; b:=a+5;
     write(a,b)
   end.
```

```
2)program k2 ;
  var a,b,c:real;
  begin
    a:=3.5; b:=sqrt(a);
    c:=a+b; write(c)
  end.
```

```
3)program k3 ;
  var x,y:integer;
  begin
    x:=5; y:=sqr(x);
    write (x,y)
  end.
```

440. Aşağıdakı proqramların nəticələri təyin etməli.

```
1) program k4 ;
   var x,y:real;
    begin
      x:=-12.5; y:=7.5;
      if abs(x)>abs(y) then x:=x/5;
      write (x,y)
    end.
```

```
2)program k5 ;
var x:integer; s:real;
  begin
     x:=5;
     s:=1+x+sqr (x)/2;
     write (s)
   end.
```

```
3)program k6 ;
var x,y:real;
   begin x:=3.2;
     if x>=0 then y:=0 else y:=sqr(x);
     write(y)
   end.
```

441. Aşağıdakı proqramların nəticələrini təyin etməli:

```
1)program k7 ;
var x,y:real;
   begin x:=2.5;
     if x<2 then y:= x else
      if x<3 then y:=2 else y:=-x+5;
     write (y)
   end.
```

```
2)program k8 ;
var x,y:real;
    begin x:=1.5;
```

```
if x>1 and x<2 then y:=sqr(x)-1
  else y:=-x- 4;
write (y)
end.
```

442. x həqiqi ədədi verilib. Əgər x>0 olarsa, S-ə 2*x qiymətini, əks halda x^3 mənimsətməli.
443. x tam ədədi verilib. Əgər x>2 olarsa, S-ə x^2 qiymətini, əks halda 4*x mənimsətməli.
444. x və y həqiqi ədədləri verilib. Əgər x>y olarsa, S-ə x-y qiymətini, əks halda $y-x^3$ mənimsətməli.
445. x və y həqiqi ədədləri verilib. Əgər x+y>5 olarsa, S-ə x+y qiymətini, əks halda x^2+y^2 mənimsətməli.
446. a,b,c həqiqi ədədləri verilib. a,b,c əmsallı kvadrat tənliyin kökləri varsa, onları təyin etməli.
447. x həqiqi ədədi verilib. Əgər 0<x<2 olarsa, S-ə x+2 qiymətini, əks halda x-2 mənimsətməli.
448. x,y,z həqiqi ədədləri verilib. Əgər x+y>z olarsa, S-ə x+y+z qiymətini, əks halda x-y-z mənimsətməli.
449. x tam ədədi verilib. Əgər x<0 olarsa, S-ə x-i, x>3 olarsa 2, qalan hallarda x+5 mənimsətməli.
450. Birinci n natural ədədin cəmini tapmalı.
451. Birinci n natural ədədin hasilini tapmalı.
452. x və y koordinatlı nöqtənin, mərkəzi koordinat mərkəzi ilə üst-üstə düşən r radiuslu dairənin daxilinə düşüb, düşmədiyini tə'yin etməli.
453. a,b,c həqiqi ədədləri verilib. Onlardan neçəsinin mənfi ədəd olduğunu təyin etməli.
454. a,b,c həqiqi ədədləri verilib. Onlardan ən böyük ikisinin hasilini tapmalı.
455. x həqiqi ədədi və a,b tam ədədləri verilib. x ədədinin (a,b) intervalına aid olub-olmadığını tə'yin etməli.
456. Sahəsi S1 olan dairə və sahəsi S2 olan kvadrat verilib. Təyin etməli: kvadrat dairənin daxilində yerləşərmi.
457. a,b (a<b) həqiqi ədədləri və n natural ədədi verilib. y=f(x) funksiyası [a,b] parçasında təyin edilib. Arqumentin x_i=a+ih (i=0,n) (h=(b-a)/n) qiymətləri üçün funksiyanın x_i=f(x_i) qiymətlərini tapmalı:
1) $y=2x^2+1$; 2) $y=x|x+1|$; 3) y=sin(x); 4) $y=e^{-x}$;
5) $y=3x^3+5$; 6) y=1-cos(x^2); 7) $y=2x^2/5$;
8) y=|sin(x)|+|cos(x)|.
458. x həqiqi ədədi verilib. Əgər 0<x<2 olarsa, S-ə x+2, əks halda x^2-2 mənimsətməli.
459. x tam ədədi verilib. Əgər x<0 olarsa, S-ə -1, x=0 olarsa 0 və x>0 olarsa 1 qiymətini mənimsətməli.
460. x həqiqi ədədi verilib. Əgər x>2 olarsa, S-ə 1-x qiymətini, -2<x<1 olarsa x^2, qalan hallarda isə 2x mənimsətməli.

461. n tam ədədi və n sayda həqiqi ədədlər verilir, onlardan ən böyüyünü tapmalı.

462. n tam ədədi və n sayda həqiqi ədədlər verilir, müsbət ədədlərin sayını tapmalı.

Məntiqi ifadələr

463. Aşağıdakı ifadələrin qiymətlərini hesablamalı.
1) `sqr(x)+sqr(y)<=4; x=0,3` və `y=-1,6` üçün;
2) `K mod 7=k div 5-1; k=15` üçün;
3) `odd(trunk(10*p)); p=0,2` üçün.

464. Aşağıdakı ifadələrin qiymətlərini hesablamalı.
1) `not odd(n); n=0` üçün;
2) `t and(p mod 3=0); t=true` və `p=10` üçün;
3) `(x*y<>0)and(y>x); x=2` və `y=1` üçün;
4) `(x*y<>0)or(y>x); x=2` və `y=1` üçün;
5) `a or(not b); a=false` və `b=true` üçün.

465. `a=true` və `b=false` olduğu halda aşağıdakı ifadələri hesablamalı.
1) `a or b and not a;` 2) `(a or b)and not a;`
3) `not a and b;` 4) `not(a and b).`

466. Aşağıdakı şərtlər ödəndiyi halda `true` ödənmədiyi halda isə `false` qiymətləri alan ifadələr qurmalı.
1) $x \in [0,1]$; 2) $x \notin [0,1]$;
3) x, [2,5] və ya [-1,1] parçalarına aiddir;
4) x, [2,5] və ya [-1,1] parçalarına aid deyil;
5) x,y,z ədədlərindən hər biri müsbətdir;
6) x,y,z ədədlərindən heç olmasa biri müsbətdir;
7) x,y,z ədədlərindən heç biri müsbət deyil.

467. Aşağıdakı ifadələrin qiymətlərini hesablamalı.
1) `ord(false)=1;` 2) `pred(true);` 3) `ord(succ(false))>0`

468. Aşağıdakı ifadələrin qiymətlərini hesablamalı.
1) `not(pred(c)or(ord(c)=1)); c=true` üçün;
2) `(p<true)=(q=false); p=q=true` üçün;
3) `a and b>a or b; a=false, b=true` üçün.

469. a,b,c natural ədədləri Pifaqor üçlüyünü təşkil edirsə, yəni $c^2=a^2+b^2$ şərtini ödəyirsə `true`, ödəmirsə `false` qiymətini almalı.

470. Verilmiş x,y ədədləri birinci koordinat rübündə yerləşən nöqtənin koordinatlarıdırsa `true`, əks halda `false` qiymətini almalı.

471. (x_1,y_1) və (x_2,y_2) düzbucaqlının uyğun olaraq yuxarı sol və aşağı sağ təpə nöqtələrinin koordinatlarıdır. Əgər A(x,y) nöqtəsi düzbucaqlıya aiddirsə `true`, əks halda `false` qiymətlərini almalı.

472. C ədədi a və b ədədlərinin ədədi ortasıdırsa, `true`, əks halda `false` qiymətini almalı.

473. N natural ədədi dəqiq kvadratdırsa, `true`, əks halda `false` qiymətini almalı.

474. C və d ədədləri uyğun olaraq a ədədinin kvadratı və kubudursa, true, əks halda false qiymətini almalı.

475. X – mənfi tam ədəd olub, K ədədinə tam bölünürsə, true, əks halda false qiymətini almalı.

476. x,y,z ədədləri öz aralarında bərabərdirsə true, əks halda false qiymətini almalı.

477. x,y,z ədədlərindən yalnız ikisi öz aralarında bərabərdirsə true, əks halda false qiymətini almalı.

478. a=true, x=1 olduqda d məntiqi dəyişəni hansı qiyməti alar:
1) d:=x<2; 2) d:=not a or odd(x); 3) d:=ord(a)<>x.

479. n və k natural ədədlərinin hər ikisi cütdürsə true, əks halda false qiymətini almalı.

480. a və b məntiqi dəyişənlərindən yalnız biri true qiymətini alırsa true, əks halda false qiymətini almalı.

481. a,b,c məntiqi dəyişənlərindən yalnız biri false qiymətini alırsa true, əks halda false qiymətini almalı.

482. Aşağıdakı ifadələrin doğruluğunu isbat etməli:
1) a and(not a)≡false; 2) a or(not a)≡true;
3) not(not a)≡a; 4) true or a≡true;
5) false and a≡false; 6) a or a≡a.

483. Aşağıdakı ifadələrin doğruluğunu isbat etməli:
1) not(a or b)≡(not a)and(not b);
2) a and(b or c)≡(a and b)or(a and c);
3) a<=b≡not a or b;
4) a and b≡(a<true)<b;
5) not a≡a<true.

Sadə dövrlər. Budaqlanan strukturlu proqramlar

484. N tam ədədi verilib. Hesablamalı:
1) 1+2+...+n; 3) 1+1/2+....+1/n;
2) 1*3*5*...*(2n-1); 4) $(1/2^2)*(1/3^2)*...*(1/(n+1)^2)$.

485. a həqiqi ədədi və n tam ədədi verilib. Hesablamalı:
1) a^n; 2) $(a^2+1)*(a^2+2)*...*(a^2+n)$;
3) $\sqrt{a}+2\sqrt{a}+...+n\sqrt{a}$;
4) 1/a+1/(a(a-1))+...+1/(a(a-1)*...*(a-n)).

486. n tam ədədi verilib. Hesablamalı:
1) 2^n; 2) $(1+1/1^2)*(1+1/2^2)*...*(1+1/n^2)$;
3) (1/sin(1))+(1/(sin(1)+sin(2)))+...+(1/(sin(1)+...+sin(n)))
4) (2/(1+3))*(4/(2+3))*...*(2n/(n+3)).

487. N tam ədədi verilib. Hesablamalı: 1*2+2*3+...+n*(n+1)

488. x həqiqi ədədi verilib. Hesablamalı:
((x-1)+(x-3)+... +(x-59))/((x-2)*(x-4)*...*(x-60)).

489. x həqiqi ədədi verilib. 1; 1/(1+2); 1/(1+2+3);... ədədləri arasında x ədədindən kiçik birinci həddi tapmalı.

490. A həqiqi ədədi verilib. (1*2*...*n)>A şərtini ödəyən birinci n qiymətini tapmalı.

491. n tam ədədi verilib. Hesablamalı:

1) (1/1!)*(2/2!)*...*(n/n!);
2) (1+1/1!)+(1+1/2!)+...+(1+1/n!);
3) (1/1!)*(1/2!)*...*(1/n!);
4) 1!+(3*2!)+...+(2n-1)*n!.

492. n tam ədədi verilib. Birinci n toplananın cəmini tap:

1) (1/2)+(1/4)+...; 2) (2*3)+(4*5)+... .

493. n tam ədədi verilib. Hesablamalı:

1) 1/3+1/5+...+1/(2n-1);
2) $(1/2^2)*(1/4^2)*...*(1/2n^2)$;
3) $(\sqrt{1}/2)+(\sqrt{3}/4)+...+(\sqrt{2n-1}/2n)$.

494. x həqiqi və n tam ədədi verilib. Hesablamalı:

1) (x/1!)*(2x/2!)*...*(nx/n!);
2) $(\sqrt{x})+(2\sqrt{x})+...+(n\sqrt{x})$;
3) $(2x-\cos(3x))^2*(4x-\cos(5x))^2*...*((2n)*x-\cos((2n-1)x))^2$;
4) (sin(3x)+cos(2x))/1!+...+(sin((2n-1)x)+cos((2n)x))/n!

495. n tam ədədi və x həqiqi ədədi verilib. Hesablamalı:

1) 1/n+1/(n-1)+...+1/1;
2) $(2n^2-1)*(2(n-1)^2-1)*...*(2^2-1)$;
3) (sin(n*x)/n)+(sin((n-1)*x)/(n-1))+...+(sin(x)/1).

496. Hesablamalı:

1) $1/1^2+1/2^2+...+1/100^2$;
2) $1/1^3+1/2^3+...+1/50^3$;
3) $1/2^2+1/4^2+...+1/128^2$;
4) $(1^2/(1^2+3))*(2^2/(2^2+3))*...*(52^2/(52^2+3))$.

497. n tam ədədi və x-həqiqi ədədi verilib. Hesablamalı:

1) $x/1!+x^2/2!+...+x^n/n!$;
2) (1+sin(x)/1!)*(1+sin(2x)/2!)*...*(1+sin(nx)/n!);
3) $(1/2-\cos|x|)*(2/3-\cos^2|x|)*...*(n/(n+1)-\cos^n|x|)$;
4) $(1/1!+x)+(1/2!+x^2)+...+(1/n!+x^n)$.

498. n natural ədədi verilib. Hesablamalı:

1) $1/3^2+1/5^2+...+1/(2n+1)^2$;
2) (1/(1*2))*(1/(2*3))*...*(1/(n*(n+1)));
3) 2/1!+3/2!+...+(n+1)/n!;
4) 1!/1+2!/(1+1/2)+...+n!/(1+1/2+...+1/n).

499. Natural n ədədi verilib. Hesablamalı:

1) $(1/1)^n+(1/2)^n+...+(1/n)^n$;
2) $(1/1)^1+(1/2)^2+...+(1/n)^n$;
3) $(1+1/1^n)+(1+1/2^n)+...+(1+1/n^n)$.

500. Natural n ədədi verilib. Hesablamalı:

1) $(1+1/1^1)*(1+1/2^2)*...*(1+1/n^n)$;

2) $(1+1/1)^n*(1+1/2)^n*...*(1+1/n)^n$;
3) $(1+1/1)^1*(1+1/2)^2*...*(1+1/n)^n$.

501. N natural ədədi verilib. Hesablamalı:
1) $(1/3)^1*(1/5)^2....(1/(2n-1))^n$;
2) $(1/2)^n+(1/4)^n+...+(1/2n)^n$;
3) $(1+1/3^n)*(1+1/5^n)*...*(1+1/(2n-1)^n)$;
4) $(1+1/2)^1+(1+1/4)^2+...+(1+1/2n)^n$

502. Hesablamalı: $1*2+2*3*4+...+50*51*...*100$

503. Tam n və k ədədləri verilib $(n>k>0)$.
Hesablamalı: $n*(n-1)*...*(n-k+1)/k!$

504. Verilmiş üç ədəddən müsbətlərinin kvadratını, mənfilərinin isə kubunu tapmalı.

505. $A(x_1,y_1)$ və $B(x_2,y_2)$ nöqtələri verilib. Onlardan koordinat mərkəzinə daha yaxın olanını tapmalı.

506. Bir-birindən fərqli x və y ədədləri verilib. Onlardan kiçiyini, onların cəminin yarısı ilə, böyüyünü isə onların ikiqat hasili ilə əvəz etməli.

507. Verilmiş n və m ədədləri bərabərdirsə, bu ədədlərin hər ikisini onlar arasında ən böyük olanı ilə, əks halda isə ən kiçik olanı ilə əvəz etməli.

508. Verilmiş üç ədəddən ən böyüyünün və ən kiçiyinin cəmini tapmalı.

509. $max(min(a,b),min(c,d))$ tapmalı.

510. Verilmiş a,b,c ədədlərindən hansının d-yə bərabər olduğunu tapmalı. Onlardan heç biri d-yə bərabər deyilsə, $max(d-a,d-b,d-c)$ -ni tapmalı.

511. a,b,c ədədləri verilib, $a \geq b \geq c$ şərti ödənərsə, bu ədədləri 2-yə vurmalı, əks halda onların mütləq qiymətlərini tapmalı.

512. Hesablamalı: $f(x)=\begin{cases} x^2-3x+9, & x \leq 3 \\ 1/(x^3+6), & x>3 \end{cases}$

513. Hesablamalı: $f(x)=\begin{cases} -x^2+3x+9, & x \geq 3 \\ 1/(x^3-6), & x<3 \end{cases}$

514. Hesablamalı: $f(x)=\begin{cases} 9, & x \leq -3 \\ 1/(x^2+1), & x>-3 \end{cases}$

515. Hesablamalı: $f(x)=\begin{cases} 0, & x \leq 1 \\ 1/(x+6), & x>1 \end{cases}$

516. Hesablamalı: $f(x)=\begin{cases} -3x+9, & x \leq 7 \\ 1/x-7, & x>7 \end{cases}$

517. Hesablamalı. $f(x)=\begin{cases} x^2, & 0 \leq x \leq 3 \\ 4, & \text{əks halda} \end{cases}$

518. Hesablamalı: $f(x)=\begin{cases} x^2-x, & 0 \leq x \leq 1 \\ x^2-\sin(x^2), & \text{əks halda} \end{cases}$

519. Hesablamalı: $f(x)=\begin{cases}-x^2+x-9, & x\ge 8\\ 1/(x^4-6), & x<8\end{cases}$

520. Hesablamalı: $f(x)=\begin{cases}4x^2+2x-19, & x\ge -3.5\\ 2x/(4x+1), & x<3.5\end{cases}$

521. Hesablamalı: $f(x)=\begin{cases}-x^2+3x+9, & x\le 3\\ x/(x^2+1), & x>3\end{cases}$

522. Hesablamalı: $f(x)=\begin{cases}-3x+9, & x>3\\ x^3/(x^2+8), & x\le 3\end{cases}$

523. Hesablamalı: $f(x)=\begin{cases}-x^3+9, & x\le 13\\ -3/(x+1), & x>13\end{cases}$

524. Hesablamalı: $f(x)=\begin{cases}45x^2+5, & x>3.6\\ 5x/(10x^2+1), & x\le 3.6\end{cases}$

525. Hesablamalı: $f(x)=\begin{cases}x^4+9, & x<3.2\\ 54x^4/(-5x^2+7), & x\ge 3.2\end{cases}$

526. Hesablamalı: $f(x)=\begin{cases}x^2+3x+9, & x\le 3\\ \sin(x)/(x^2-9), & x>3\end{cases}$

527. Hesablamalı: $f(x)=\begin{cases}\cos(2x)+9, & x>-4\\ \cos(x)/(x+9), & x\le -4\end{cases}$

528. n tam ədədi verilib. Birinci n vuruğu tapmalı:
`(2/3)*(4/5)*(6/7)*...`

529. n tam ədədi verilib. Hesablamalı:
`cos(1)/sin(1)*(cos(1)+cos(2))/(sin(1)+sin(2))*...*`
`*(cos(1)+cos(2)+...+cos(n))/(sin(1)+sin(2)+...+sin(n))`.

530. x həqiqi ədədi verilib. Hesablamalı:
$$x-\frac{x^3}{3!}+\frac{x^5}{5!}+\frac{x^7}{7!}+\frac{x^9}{9!}+\frac{x^{11}}{11!}+\frac{x^{13}}{13!}.$$

531. n natural ədədi və həqiqi x ədədi verilib. Hesablamalı:
$$\sin(x)+(\sin(x)\cdot\sin(x))+\ldots+\underbrace{(\sin(x)\cdot\ldots\cdot\sin(x))}_{n\ \text{sayda}}.$$

532. n natural ədədi və həqiqi x ədədi verilib. Hesablamalı:
$a(a-n)(a-2n)\ldots(a-n^2)$.

533. n natural ədədi və həqiqi və həqiqi a ədədi verilib. Hesablamalı:
$$\frac{1}{a}+\frac{1}{a^2}+\frac{1}{a^4}+\ldots+\frac{1}{a^{2n-2}}.$$

534. n natural ədədi və həqiqi x ədədi verilib. Hesablamalı:
$\sin(x)+\sin(x^2)+\ldots+\sin(x^n)$.

535. n natural ədədi verilib. Hesablamalı:
$1\cdot 2+2\cdot 3\cdot 4+3\cdot 4\cdot 5\cdot 6+\ldots+n(n+1)\ldots 2n$.

536. n natural ədədi və həqiqi x ədədi verilib. Hesablamalı:
$\cos(x)+\cos(x^2)+\ldots+\cos(x^n)$.

537. n natural ədədi verilib. Hesablamalı:
$\frac{2}{1}+\frac{3}{2}+\frac{4}{3}+\ldots+\frac{n+1}{n}$.

Massivlər

538. n elementi olan birölçülü A massivi verilib. Hesablamalı:
1) massivin bütün elementlərinin cəmini;
2) massiv elementlərinin ədədi ortasını;
3) massiv elementlərinin kvadratlarının hasilini;
4) massiv elementlərinin mütləq qiymətlərinin hasilini.

539. n elementi olan A massivi verilib. Massivin elementlərinin hasilini tapıb, bu massiv elementlərinin hər birindən bu qiyməti çıxıb, nəticələri bir n-ölçülü B massivinə yerləşdirməli.

540. n sayda müsbət elementi olan A massivi verilib. Hesablamalı:
1) $a_1+\ldots+a_n$ və $a_1*\ldots*a_n$; 2) $\sqrt{a_1}+\ldots+\sqrt{a_n}$;
3) $2(a_1+\ldots+a_n)^2$; 4) $\sin|a_1+\ldots+a_n|$.

541. n-ölçülü A və B vektorları verilib. Hesablamalı:
1) verilmiş massivlərin cəmini; 2) verilmiş massivlərin fərqini.

542. n elementi olan A massivi verilib. Hesablamalı:
1) $\sqrt{|a_1*\ldots*a_n|}$; 2) $(\sqrt{|a_1|-a_1})^2+\ldots+(\sqrt{|a_n|-a_n})^2)$;
3) $\sqrt{10+a_1^2}+\ldots+\sqrt{10+a_n^2}$; 4) $\cos(|a_1|)*\ldots*\cos(|a_n|)$.

543. n elementi olan X massivi verilib. Aşağıdakı funksiyaların qiymətlərini X massivində verilmiş arqumentin qiymətləri üçün hesablayıb, Y massivində verməli:
1) $y=\cos(x)/x$; 2) $y=x^2-1$; 3) $y=\sin(x)/(1-x)^2$;
4) $y=1/\sin(x)$; 5) $y=|\cos(x)|$; 6) $y=x^2/e^x$.

544. n-ölçülü A və B vektorları verilib. Hesablamalı:
1) verilmiş massivlərin ünsürlərinin hasilini;
2) verilmiş massivlərin ünsürlərinin nisbətini.

545. N ölçülü A massivi verilib. Hesablamalı:
1) $|a_n*a_{n-1}*\ldots*a_1|$; 2) $(a_n+a_{n-1}.+\ldots+a_1)^3$;
3) $na_n*(n-1)a_{n-1}*\ldots*a_1$.

546. n ölçülü A vektoru verilib. Aşağıdakıları hesablayıb, nəticələri B massivində verməli:
1) $|a_1|,|a_1+a_2|,\ldots,|a_1+\ldots+a_n|$;
2) $a_1,a_1*a_2,\ldots,a_1*a_n$;
3) $a_1+1!,a_2+2!,\ldots,a_n+n!$.

547. n ölçülü A vektoru verilib. Hesablamalı:
1) $a_1/1!+\ldots+a_n/n!$; 2) $(1+1!/a_1)\ldots(1+n!/a_n)$;
3) $(a_1^2+1/1!)+\ldots+(a_n^2+1/n!)$.

548. n elementi olan A vektoru verilib. Massivin tək indeksli elementlərini B massivində cüt indeksli elementlərini isə hər hansı C massivində verməli.

549. Ardıcıllıq aşağıdakı qayda üzrə qurulub: $x_1=1$; $x_i=((i+1)/2)x_{i-1}$, $(i=2,3,\ldots,n)$. Ardıcıllığın $x_2,\ldots,x_n$ həddlərini tapmalı.

550. Ardıcıllıq aşağıdakı qayda üzrə qurulub: $x_1=1$; $x_2=0,3$; $x_i=(i+1)x_{i-2}$ $(i=3,4,\ldots,n)$. Ardıcıllığın $x_3,\ldots,x_n$ həddlərini tapmalı.

551. n sayda sətri, m sayda sütunu olan A matrisi verilib. Matrisin mütləq qiymətcə elementlərinin cəmini tapmalı.

552. c n×m ölçülü A matrisi verilib. Matris elementlərinin kvadratları hasilini tapmalı.

553. n×n ölçülü A matrisi verilib. Matrisin baş diaqonal elementlərinin mütləq qiymətcə cəmini tapmalı.

554. n×n ölçülü A və B matrisləri verilib. Bu matrislərin elementlərinin kvadratları cəmini və fərqini tapıb, nəticəni həmin ölçülü bir C matrisində yerləşdirməli.

555. n×m ölçülü A matrisinin l nömrəli (1<l<n) sətrinin elementlərinin hasilini tapmalı.

556. n×m ölçülü A matrisinin k nömrəli (1<k<m) sütununun elementlərinin cəmini tapmalı.

557. n×n ölçülü A matrisinin l nömrəli sətr elementləri ilə k nömrəli sətir elementlərinin hasilini tapıb, nəticəni n ölçülü B vektoruna verməli.

558. n×n ölçülü A matrisinin k nömrəli sətri ilə l nömrəli sütununun elementlərinin kvadratları cəmini tapıb, B massivində yerləşdirməli.

559. n×n ölçülü A və B matrisləri verilib. Bu matrisləri birləşdirib, nəticəni n×2n ölçülü bir C matrisində yerləşdirməli.

560. n ölçülü A vektoru verilib. Bu massivin elementlərindən B(I,J)=(A(I)*A(I))/(2*A(J)) düsturu ilə n*n ölçülü B matrisini qurmalı.

561. n×n ölçülü A matrisinin k nömrəli sətrindəki elementləri, bu sətirdəki baş diaqonal elementinə bölüb, nəticələri n ölçülü B massivində yerləşdirməli.

562. n ölçülü A vektoru verilib. Bu massivin elementlərindən B(I,J)=A(I)-1/A(J) düsturu ilə n×n ölçülü B matrisini qurmalı.

563. Natural ədədlərdən ibarət n ölçülü A massivi verilib. Massivin verilmiş k natural ədədinə qalıqsız bölünən elementlərinin cəmini tapmalı.

564. Tam ədədlərdən ibarət n ölçülü A massivi verilib. Massivdə rast gəlinən sıfır elementlərinin sıra nömrələrindən ibarət massiv qurmalı.

565. Tam ədədlərdən ibarət n ölçülü A massivi verilib. Massivdə müsbət və ya mənfi elementin birinci gəldiyini təyin etməli.

566. Natural ədədlərdən ibarət n ölçülü A massivi verilib. Massivdəki cüt ədədlərdən ibarət massiv qurmalı, belə ədədlər yoxdursa, bu faktı elan etməli.

567. Həqiqi ədədlərdən ibarət n ölçülü A massivi verilib. Massivin verilmiş Z ədədindən böyük olan bütün elementlərini bu ədədlə əvəz etməli və əvəzləmələrin sayını təyin etməli.

568. Həqiqi ədədlərdən ibarət **n** ölçülü **A** massivi verilib. Massivdəki mənfi, müsbət və sıfır elementlərinin sayını təyin etməli.

569. Tam ədədlərdən ibarət **n** ölçülü **A** massivi verilib. Massivin $a_i \geq i$ şərtini ödəyən elementlərini çap etməli.

570. Natural ədədlərdən ibarət **n** ölçülü **A** massivi verilib. Massivin **m** ədədinə bölünməsi nəticəsində qalıq həddi **1** olan **(0≤1≤m-1)** elementlərini çap etməli.

571. **n** ölçülü **A** massivi verilib. Massivin tək indeksli elementlərini cüt indeksli elementləri ilə əvəz etməli.

572. Tam ədədlərdən ibarət **n** ölçülü **A** massivi verilib. Massivdə bir sıfır ədədi var. Bu ədəd də daxil olmaqla, sıfır ədədinə qədər olan elementləri çap etməli.

573. Tam ədədlərdən ibarət **n** ölçülü **A** massivi verilib. Massivdə imtahan nəticələri göstərilib. İmtahandan 2 və 5 qiymət almış tələbələrin sayını təyin etməli.

574. **n** ölçülü **A** massivi verilib. Massivdə əvvəlcə arqumentin qiyməti, onun ardınca isə uyğun funksiya qiyməti verilib. Arqument və funksiya qiymətlərini ayrı-ayrı massivlərdə verməli.

575. **n** ölçülü **A** massivi verilib. Massivdə hazırlanan detalların ölçüləri verilib. Detalın ölçüsü (c,d) intervalında olmalıdır. Zay detalların sayını, belə detallar yoxdursa, sıfır qiymətini çapa verməli.

576. **n** ölçülü **A** massivi verilib. Massivdə şəhərdəki **n** bankda dolların manata dəyişdirmə məzənnəsi verilib. Ən əlverişli dəyişmə kursu olan bankı təyin etməli.

577. Tam ədədlərdən ibarət **n** ölçülü **A** massivi verilib. Massivin ikinin tərtiblərinə bərabər (yəni 2,4,8,16,...) indeksli elementlərini çapa verməli.

578. Həqiqi ədədlərdən ibarət **n**-ölçülü **A** massivi verilib. Massiv elementləri arasında verilmiş **k** ədədindən kiçik, böyük və **k**-ya bərabər elementlərin sayını tapmalı.

579. Həqiqi ədədlərdən ibarət **n**-ölçülü **X** massivi verilib. Hesablamalı:

$$y_i = \sqrt{(x_i - M)^2/(n-1)},$$

burada **M** – **X** massivindəki elementlərin ədədi ortasıdır.

580. Həqiqi ədədlərdən ibarət **n**-ölçülü **A** massivi verilib. Massivin, indeksləri sadə ədədlərlə üst-üstə düşən elementlərinin cəmini tapmalı.

581. Həqiqi ədədlərdən ibarət **n**-ölçülü **A** massivi verilib. Massivin indeksləri Fibonaççi ədədləri olan elementlərinin cəmini tapmalı.

582. Həqiqi ədədlərdən ibarət **n**-ölçülü **A** massivi verilib. Hesablamalı:

$$\sqrt[n]{|a_1 a_2 \dots a_n|}.$$

583. **n**-ölçülü **A** massivi verilib. Tapmalı:

$$\max(a_2, a_4, \dots, a_{2k}) + \min(a_1, a_3, \dots, a_{2k+1}),\ (k<n).$$

584. Həqiqi ədədlərdən ibarət **n**-ölçülü **A** massivi verilib. Massivin verilmiş **[c,d]** parçasına aid olan elementlərini çapa verməli.

585. Müsbət tam ədədlərdən ibarət **n**-ölçülü **A** massivi verilib. Massivin verilmiş

M ədədindən böyük olan elementlərinin hasilini tapmalı, belə elementlər yoxdursa, bu barədə məlumat verməli.

586. n-ölçülü A massivi verilib. Massiv sıfır və vahidlərdən təşkil olunub. Sıfırları massivin əvvəlinə, vahidləri isə sonuna yerləşdirməli.

587. Həqiqi ədədlərdən ibarət n ölçülü A massivi verilib. Massivin $|a_i|>\max(a_1,\dots,a_n)$ şərtini ödəyən elementlərini sıfırla əvəz etməli.

588. Həqiqi ədədlərdən ibarət n-ölçülü A massivi verilib. Hesablamalı: $\max(a_1+a_{2k}, a_2+a_{2k-1}, a_n+a_{k+1})$, $(k<n)$.

589. Həqiqi ədədlərdən ibarət n-ölçülü A massivi verilib. Massiv yalnız müsbət və mənfi ədədlərdən ibarətdir. Massivdəki müsbət və mənfi ədədlərin hasilini tapıb, onları mütləq qiymətcə müqayisə etməli və hansı hasilin daha böyük olduğunu təyin etməli.

590. Tam ədədlərdən ibarət n-ölçülü A massivi verilib. Massivin verilmiş m ədədinin kvadratı olan elementlərini tapmalı.

591. Tam ədədlərdən ibarət n-ölçülü A massivi verilib. Massivdən $\min(a_1,\dots,a_n)$ qiymətinə bərabər elementləri çıxarıb, yeni massiv qurmalı.

592. Tam ədədlərdən ibarət n-ölçülü A massivi növbədə duran n sayda alıcının hər birinə mağazada ona sərf olunmuş zaman (dəqiqə ilə) ardıcıllığından ibarətdir. Hər hansı k-cı $(1\le k\le n)$ alıcının növbədə durduğu zamanı hesablamalı.

593. Tam ədədlərdən ibarət n-ölçülü A massivi verilib. Massivin minimal elementini, massiv elementlərinin ədədi ortasının tam hissəsi ilə əvəz etməli.

594. Tam ədədlərdən ibarət n-ölçülü A və B massivləri verilib. B massivini aşağıdakı qaydada dəyişməli:

$$b_i=\begin{cases}10b_i, & a_i \le 0\\ 0, & \text{əks halda}\end{cases}$$

595. Həqiqi ədədlərdən ibarət n-ölçülü A massivi verilib. Massivi aşağıdakı qaydada dəyişməli:

$$a_i=\begin{cases}a_i\ \min^2(a_i), & a_k \ge 0\\ a_i\ \max^2(a_i), & a_k < 0\end{cases} \quad (1\le k\le n).$$

596. Tam ədədlərdən ibarət n-ölçülü A massivi verilib. Massivin maksimal və minimal elementləri arasındakı elementlərinin cəmini tapmalı.

597. Tam ədədlərdən ibarət n-ölçülü A massivi verilib. Massivin minimal m və maksimal M elementlərini tapıb, (m,M) intervalına daxil olan və A massivinə aid olmayan tam ədədləri artım sırası ilə çapa verməli.

598. Tam ədədlərdən ibarət n-ölçülü A massivi verilib. Massivdə n sayda insanın yaşı gətirilib. Yaşları 10 il intervalında (yəni 0 – 9 il, 10 – 19 il, 20 – 29 il və s.) olan insanların sayını tapmalı.

Mürəkkəb dövr və budaqlanan strukturlu proqramlar

599. Ardıcıllıq $a_1=1$, $a_i=a_{i-1}/2+1/i$ $(i=2,3,\dots)$ qaydası ilə qurulub. $a_1+\dots+a_n$ cəmini tapmalı.

600. Ardıcıllıq $a_1=1$, $a_2=1$, $a_i=(a^2_{i-1}+1)+a^2_{i-2}$ $(i=3,4,...)$ qaydası ilə qurulub.
$a_1/1!+...+a_n/n!$ cəmini tapmalı.

601. Ardıcıllıqlar $x_1=1$; $y_1=1$; $x_i=x^2_{i-1}+y_{i-1}$, $y_i=1/y_{i-1}$ $(i=2,3,...)$ qaydası ilə qurulub. Hesablamalı: $(x_1/2*y_1).....(x_n/2*y_n)$.

602. N natural ədədi verilib. $i=1,...,n$ qiymətləri üçün aşağıdakı qiymətləri alan $B(i)$ ardıcıllığını almalı: 1) $1/i!$; 2) $1+1/2+.....+1/i$; 3) $i(1/1!+....+1/i!)$.

603. $(2i^2+1)/(i^3-7i^2-12)$ ifadəsinin qiymətlərini $i=1,2,...,30$ üçün hesablamalı.

604. x^3-2x^2+3x+5 çoxhədlisinin qiymətlərini $x=1,\ 1.5,\ 2,...,10$ üçün hesablamalı.

605. $F1(x)=1+\sqrt{1-\cos(x)}$, $F2(x)=e^x\sin(x)$, $F3=\sin(x)*\cos(x)$ funksiyalarının, arqumentin $[0.2,\ 2]$ parçasında $h=0.2$ addımı ilə qiymətlərini hesablayın.

606. $F1(x)=x^2*\sin(x)$, $F2(x)=1/\sin(x)$, funksiyalarının, arqumentin $[1,2]$ parçasında $h=0.1$ addımı ilə qiymətlərini hesablamalı.

607. n natural ədədi və $a_1,...,a_n$ həqiqi ədədləri verilib. $b_1,...,b_{n-2}$ ardıcıllığını almalı, harada $b_i=a_{i+2}+a_{i+1}$, $i=1,...,n-2$.

608. n natural ədədi və $a_1,...,a_n$ həqiqi ədədləri verilib. $b_1,...,b_n$ ardıcıllığını almalı, harada $b_i=a_i/(1+(a_1+...+a_i)^2)$ $i=1,...,n$.

609. Tutaq ki, $a_1,a_2....$ həqiqi ədədləri verilib. Məlumdur ki, $a_1>0$, və a_2, a_3,... ədədləri arasında heç olmasa bir mənfi ədədi var. Bu ardıcıllığın birinci mənfi ədədə rast gələnə qədər olan $a_1,a_2,...$ ədədlərinin 1) cəmini; 2) hasilini; 3) ədədi ortasını hesablamalı.

610. n natural ədədi və $a_1,...,a_n$ həqiqi ədədləri verilib. $b_1,...,b_n$ ardıcıllığını almalı, harada $b_1=a_1$, $b_n=a_n$, $b_i=(a_{i+1}-a_i)/3$, $i=2,...,n-1$.

611. $b>0$ həqiqi ədədi verilib və $a_1,a_2,...$ ardıcıllığı $a_1=1$, $a_i=a^2_{i-1}+1$ $(i=2,3,...)$ qaydası ilə qurulub. b-dən kiçik və ya ona bərabər $a_1,a_2,...$ qiymətlərini almalı.

612. $b>0$ həqiqi ədədi verilib və $a_1,a_2,...$ ardıcıllığı $a_1=b$, $a_i=a_{i-1}+1/\sqrt{i}$ $(i=2,3...)$ qaydası ilə qurulub. Ardıcıllığın birinci mənfi ünsürünü tapmalı.

613. $b>0$ həqiqi ədədi verilib və $a_1,a_2,...$ ardıcıllığı $a_1=b$, $a_i=(a_{i-1}+1)/(i-\sin(i))$ $(i=2,3...)$ qaydası ilə qurulub. Ardıcıllığın birinci mənfi ünsürünü tapmalı.

614. a və b həqiqi ədədləri verilib $(a>1)$. b ədədindən kiçik olan $a,a^2,a^3,...$ ardıcıllığının bütün hədlərini almalı.

615. $a>0$ həqiqi ədədi verilib və $x_1,x_2,...$ ardıcıllığı $x_1=(a+1)/2$, $x_i=1/2(x_{i-1}+(a/x_{i-1}))$ $i=2,3,...$ qaydası ilə qurulur. Ardıcıllığın $|x^2_i-a|<0.0001$ şərtini ödəyən bütün həddlərini almalı.

616. x və eps həqiqi ədədləri verilib. $1+x/1!+x^2/2!+...$ sırasının eps dəqiqliyi ilə cəmini tapmalı.

617. **x** və **eps** həqiqi ədədləri verilib. $x+x^2/2+x^3/3+\ldots$ sırasının **eps** dəqiqliyi ilə cəmini tapmalı.

618. **x** və **eps** həqiqi ədədləri verilib. $x+x^3/3+x^5/5+\ldots$ sırasının **eps** dəqiqliyi ilə cəmini tapmalı.

619. **x** və **eps** həqiqi ədədləri verilib. $(x-1)/x+(x-1)^2/2x^2+(x-1)^3/3x^3+\ldots$ sırasının **eps** dəqiqliyi ilə cəmini tapmalı.

620. **eps** həqiqi ədədi verilib. Aşağıdakı sonsuz sıraların cəmini **eps** dəqiqliyi ilə hesablamalı.

1) $1+1/2+1/3+\ldots$; 2) $1+1/3+1/5+1/7+\ldots$;

3) $1/(1*2)+1/(2*3)+1/(3*4)+\ldots$; 4) $1/(1*2*3)+1/(2*3*4)+\ldots$.

621. **eps** həqiqi ədədi verilib. $1/(1^2+2)+1/(2^2+4)+\ldots$ sırasının cəmini eps dəqiqliyi ilə hesablamalı.

622. **eps** həqiqi ədədi verilib. $a_1, a_2, \ldots$ ardıcıllığı $a_n=(1-1/2)*(1-1/3)*\ldots*(1-1/n+1)$ qaydası üzrə qurulur. $|a_n-a_{n-1}|<$**eps** şərtini ödəyən birinci a_n həddini tapmalı.

623. **x** və **eps** həqiqi ədədləri verilib. $a_1, a_2, \ldots$ ardıcıllığı $a_1=x$, $a_i=\sqrt{|4a_{i-1}-2x|}$ $(i=2,3\ldots)$ qaydası ilə qurulur. $|a_n-a_{n-1}|<$**eps** şərtini ödəyən birinci a_n həddini tapmalı.

624. **x** və **eps** həqiqi ədədləri verilib. $a_1, a_2, \ldots$ ardıcıllığı $a_1=x$, $a_i=(16+x)/(1+|a^3_{i-1}|)+3a_{i-1}$ $(i=2,3\ldots)$ qaydası ilə qurulur. $|a_n-a_{n-1}|<$**eps** şərtini ödəyən birinci a_n həddini tapmalı.

625. **x** və **eps** həqiqi ədədləri verilib. $a_1, a_2, \ldots$ ardıcıllığı $a_1=x$, $a_i=2a_{i-1}+x/(4a^2_{i-1})$ $(i=2,3\ldots)$ qaydası ilə qurulur. $|a_n-a_{n-1}|<$**eps** şərtini ödəyən birinci a_n həddini tapmalı.

626. **eps** həqiqi ədədləri verilib. Ardıcıllıqlar $x_1=10, y_1=10, x_i=0.1y_{i-1}$, $y_i=0.1x_i-0.12y_i$ $(i=2,3\ldots)$ qaydası ilə qurulub. $|x_i|<$**eps**, $|y_i|<$**eps** şərtlərini ödəyən ən kiçik i qiymətinii tapmalı.

Variant operatoru. Tam ədədlər

627. Ümumi həddi $a_n=\dfrac{(-1)^{n-1}}{n^n}$ kimi təyin edilən ədədi sıranın modulca verilmiş **eps** ədədindən kiçik bərabər olan hədlərinin cəmini tapmalı.

628. Ümumi həddi $a_n=\dfrac{1}{2^n}+\dfrac{1}{3^n}$ kimi təyin edilən ədədi sıranın modulca verilmiş **eps** ədədindən kiçik bərabər olan hədlərinin cəmini tapmalı.

629. Ümumi həddi $a_n=\dfrac{2n-1}{2^n}$ kimi təyin edilən ədədi sıranın modulca verilmiş **eps** ədədindən kiçik bərabər olan hədlərinin cəmini tapmalı.

630. Ümumi həddi $a_n=\dfrac{1}{(3n-2)(3n+1)}$ kimi təyin edilən ədədi sıranın modulca verilmiş **eps** ədədindən kiçik bərabər olan hədlərinin cəmini tapmalı.

631. Ümumi həddi $a_n=\frac{10^n}{n!}$ kimi təyin edilən ədədi sıranın modulca verilmiş **eps** ədədindən kiçik bərabər olan hədlərinin cəmini tapmalı.

632. Ümumi həddi $a_n=\frac{n!}{2(n)!}$ kimi təyin edilən ədədi sıranın modulca verilmiş **eps** ədədindən kiçik bərabər olan hədlərinin cəmini tapmalı.

633. Ümumi həddi $a_n=\frac{n!}{n^n}$ kimi təyin edilən ədədi sıranın modulca verilmiş **eps** ədədindən kiçik bərabər olan hədlərinin cəmini tapmalı.

634. Ümumi həddi $a_n=\frac{2^n\cdot n!}{n^n}$ kimi təyin edilən ədədi sıranın modulca verilmiş **eps** ədədindən kiçik bərabər olan hədlərinin cəmini tapmalı.

635. Ümumi həddi $a_n=\frac{3^n\cdot n!}{(2n)!}$ kimi təyin edilən ədədi sıranın modulca verilmiş **eps** ədədindən kiçik bərabər olan hədlərinin cəmini tapmalı.

636. Ümumi həddi $a_n=\frac{n!}{3n^n}$ kimi təyin edilən ədədi sıranın modulca verilmiş **eps** ədədindən kiçik bərabər olan hədlərinin cəmini tapmalı.

637. Ümumi həddi $a_n=\frac{n!}{(2^n)!}$ kimi təyin edilən ədədi sıranın modulca verilmiş **eps** ədədindən kiçik bərabər olan hədlərinin cəmini tapmalı.

638. Ümumi həddi $a_n=\frac{2^n}{(n-1)!}$ kimi təyin edilən ədədi sıranın modulca verilmiş **eps** ədədindən kiçik bərabər olan hədlərinin cəmini tapmalı.

639. Hədləri $a_n=arctg(a_{n-1})+1$; $a_1=0$ kimi təyin edilən ardıcıllığın $|a_n-a_{n-1}|<eps$ şərtini ödəyən $(eps=10^{-3})$ həddini tapmalı.

640. Hədləri $a_n=2+\frac{1}{a_{n-1}}$; $a_1=2$ kimi təyin edilən ardıcıllığın $|a_n-a_{n-1}|<eps$ şərtini ödəyən $(eps=10^{-3})$ həddini tapmalı.

641. Hədləri $a_n=\frac{1}{2}tg(a_{n-1})$; $a_1=0,5$ kimi təyin edilən ardıcıllığın $|a_n-a_{n-1}|<eps$ şərtini ödəyən $(eps=10^{-3})$ həddini tapmalı.

642. Hədləri $a_n=\frac{1}{(2n)^2}$; $a_1=0,25$ kimi təyin edilən ardıcıllığın $|a_n-a_{n-1}|<eps$ şərtini ödəyən $(eps=10^{-3})$ həddini tapmalı.

643. Hədləri $a_n=\frac{1}{2}\cos(a_{n\ 1})$; $a_1=0,5$ kimi təyin edilən ardıcıllığın $|a_n-a_{n-1}|<eps$ şərtini ödəyən $(eps=10^{-3})$ həddini tapmalı.

644. Hədləri $a_n=\frac{2+a^2_{n-1}}{2a_{n-1}}$; $a_1=2$ kimi təyin edilən ardıcıllığın $|a_n-a_{n-1}|<eps$ şərtini ödəyən $(eps=10^{-3})$ həddini tapmalı.

645. Hədləri $a_n=\frac{a_{n-1}+a_{n-2}}{2}$; $a_1=1$; $a_2=2$ kimi təyin edilən ardıcıllığın $|a_n-a_{n-1}|<eps$ şərtini ödəyən $(eps=10^{-3})$ həddini tapmalı.

646. Hədləri $a_n=e^{-a_{n-1}}$; $a_1=0$ kimi təyin edilən ardıcıllığın $|a_n-a_{n-1}|<eps$ şərtini ödəyən $(eps=10^{-3})$ həddini tapmalı.

647. Fəsillərin adları (qış, yaz, yay, payız) sadalanır, qış fəsilindən sonra gələn fəsilin sıra nömrəsini təyin etməli.

648. Yay aylarının adları (iyun, iyul, avqust) sadalanır, avqust ayından əvvəl gələn ayın adını təyin etməli.

649. Həftənin günləri sadalanır. Cümə günündən sonra gələn günün adını təyin etməli.

650. Proqramlaşdırma dillərinin (Beyzik, Paskal, Fortran, Ada, Lisp, Pll) siyahısı verilib. Paskal dilinin sıra nömrəsini və Lisp dilindən sonra gələn dilin adını təyin etməli.

651. Həftənin günləri sadalanır. İş günlərini ayırmalı (Bazar ertəsi, ... , Cümə).

652. İlin ayları sadalanır və M tam ədədi verilib. M nömrəli ayın adını təyin etməli $(0<M<11)$.

653. İlin ayları sadalanır və M tam ədədi verilib. M nömrəli aydan sonra gələn ayın adını təyin etməli.

654. Notlar sadalanır (do, re, mi, fa, sol, lya, si). Mi notundan sonra gələn notun adını və nömrəsini təyin etməli.

655. Rəqəmlərin adları sadalanır (sıfır, bir, iki, üç, dörd, beş) və onların uyğun rəqəm simvolları («0», ..., «5») verilib. Uyğun rəqəm simvola görə, dəyişənə onun adını mənimsəməli.

656. Ölkələrin və onların paytaxtlarının adları sadalanır. Ölkənin adını bildirən dəyişənə əsaslanaraq, ona uyğun paytaxt adını başqa bir dəyişənə mənimsətməli.

657. Fəsillərin və ilin aylarının adları sadalanır. Verilmiş m nömrəli ayın hansı fəsilə aid olduğunu təyin etməli.

658. Ölkələrin və qitələrin adları sadalanır. Hansı ölkənin hansı qitəyə aid olduğunu təyin etməli.

659. Həftənin günlərinin nömrələrini və uyğun günlərin azərbaycan və ingilis dillərində adını verməli, hər hansı k nömrəli günü seçməli.

660. Aparılmış imtahanın nəticələri verilib. Qrupda imtahandan əla, yaxşı, kafi və qeyri-kafi qiymətlər almış tələbələrin sayını təyin etməli.

661. n tam ədədi və x həqiqi ədədi verilib. $n=1$ olduqda $S=x^2$, $n=2$ üçün $S=x^2-1$ və $n=3$ üçün $S=\cos(x)$ olur. Verilmiş k tam ədədi üçün $(1\le k\le 3)$ S-in uyğun qiymətini tapmalı.

662. n tam ədədi və x həqiqi ədədi verilib. $n=1$ üçün $S=x-1$, $n=2$ üçün $S=(2x-1)/2$, $n=3$ üçün $S=\sin(x)/x$; $n=4$ üçün $S=e^x$ və $n=5$ üçün $S=\sin(x)$ olur. Verilmiş k tam ədədi üçün $(1\le k\le 5)$ S-in uyğun qiymətini tapmalı.

663. İlin aylarının nömrələrini və uyğun ayların adlarını verməli, hər hansı K nömrəli ayı seçməli.
664. Planetlərin nömrələrini və uyğun planet adlarını verməli, hər hansı K planeti seçməli.
665. Düzbucaqlı koordinat sisteminin rüblərinin nömrələrini və bu rüblərdə x, y-in aldığı işarələri verməli, hər hansı K nömrəli rübü seçməli.
666. n tam ədədi və n sayda sadə ədədlər verilib. K-.cı yerdə duran sadə ədədi (1≤K≤N) göstərin.
667. n tam ədədi verilib. Əgər bu ədəd cüt ədəddirsə, onu olduğu kimi, tək ədəddirsə, onun kvadratını çap etməli.
668. n tam ədədi və n ölçülü tam ədədlərdən ibarət A vektoru verilib. Massivdəki cüt ədədlərin və tək ədədlərin sayını tapmalı.
669. n tam ədədi və n ölçülü tam ədədlərdən ibarət A vektoru verilib. Massivdəki cüt ədədlərin sayını və cəmini tapmalı.
670. n tam ədədi və n ölçülü tam ədədlərdən ibarət A vektoru verilib. Massivin tək ədədlərinin ən böyüyünü tapmalı.
671. n tam ədədi verilib. Ədədin bütün bölənlərini azalma sırası ilə çap etməli.
672. n tam ədədi verilib. Bu ədədin mükəmməl, yəni öz bölənlərinin cəminə bərabər olub-olmadığını təyin etməli.
673. n tam ədədi və n ölçülü tam ədədlərdən ibarət A massivi verilib. Massivin cüt ədədlərindən ibarət digər bir B massivini qurmalı.
674. n və m - tam ədədləri verilib. Bu ədədlərin ən böyük ortaq bölənini tapmalı.
675. Həftənin günlərinin (1-5 tam ədədləri) nömrəsinə görə bu gün keçiriləcək dərslərin sayını təyin edən proqram qurmalı.
676. Hər bir (0-9) rəqəm üçün, onun ingilis dilindəki yazılışını (0-zero,1-one,2-two,...) verən proqram qurmalı.
677. Hər bir (1-12) rəqəm üçün, uyğun ay adlarını (yanvar, fevral,...) verən proqram qurmalı.
678. Hər bir (2-5) rəqəm üçün, uyğun qiyməti (2-qeyri-kafi, 3-kafi, 4-yaxşı, 5-əla) verən proqram qurmalı.
679. Hər bir (1-5) rəqəm üçün L parçasının uyğun ölçü vahidindəki uzunluğunu (1-desimetr, 2-kilometr, 3-metr, 4-millimetr, 5-santimetr) metrlə ifadə edən proqram qurmalı.
680. Müsbət x və y həqiqi ədədləri verilib. Bu ədədlər üzrə aparıla bilən hesabi əməliyyatlar (1-toplama, 2-çıxma, 3-vurma, 4-bölmə) (1-4) rəqəmlərlə verilib. Daxil edilən qiymətə görə uyğun hesab əməlinə aparan proqram qurmalı.
681. Hər bir (1-5) rəqəm üçün m kütləsinin uyğun ölçü vahidindəki çəkisini (1-kiloqram, 2-milliqram, 3-qram, 4-ton, 5-sentner) kiloqramla ifadə edən proqram qurmalı.

Massivlərə aid mürəkkəb məsələlər

682. n natural ədədi və n-ölçülü həqiqi ədədlərdən ibarət A vektoru verilib. Massivin ən kiçik elementini tapmalı.

683. n natural ədədi və n-ölçülü həqiqi ədədlərdən ibarət A vektoru verilib. Massivin ünsürlərinin mütləq qiymətcə ən böyüyünü tapmalı.

684. n natural ədədi və n-ölçülü həqiqi ədədlərdən ibarət A vektoru verilib. Massivin cüt indeksli ünsürlərinin ən kiçiyi ilə tək indeksli elementlərinin ən böyüyünün cəmini tapmalı.

685. n natural ədədi və n-ölçülü həqiqi ədədlərdən ibarət A vektoru verilib. Massivin mənfi ünsürlərindən ibarət B massivini qurmalı.

686. n natural ədədi və n-ölçülü həqiqi ədədlərdən ibarət A vektoru verilib. Massivin mənfi və müsbət ünsürlərinin sayını tapmalı.

687. n natural ədədi və n-ölçülü həqiqi ədədlərdən ibarət A vektoru verilib. Hesablamalı: $S=a_n(a_n+a_{n-1})\ldots(a_n+\ldots+a_1)$

688. n natural ədədi və n- ölçülü həqiqi ədədlərdən ibarət A vektoru verilib. Massivin mütləq qiymətcə i^2-dan (i=1,2,...,n) kiçik elementlərinin cəmini tapmalı

689. n natural ədədi və n-ölçülü həqiqi ədədlərdən ibarət A vektoru verilib. Massivin 5-ə qalıqsız bölünən elementlərinin cəmini tapmalı.

690. n,p natural ədədləri və n-ölçülü, tam ədədlərdən ibarət A vektoru verilib. Massivin P-yə qalıqsız bölünən elementlərinin hasilini və sayını tapmalı.

691. n natural ədədi və n-ölçülü, həqiqi ədədlərdən ibarət A vektoru verilib. Massivin elementlərini onların artımı boyunca düzməli.

692. n,m natural ədədləri və həqiqi ədədlərdən ibarət n×m ölçülü A matrisi verilib. Matrisin ən kiçik elementini və bu elementin durduğu sətr və sütunun nömrəsini tapmalı.

693. n natural ədədi və n×n ölçülü A matrisi verilib. Matrisin mənfi ünsürlərinin sayını və hasilini tapmalı.

694. n natural ədədi və n-ölçülü A vektoru verilib. Bu massivin elementlərindən B(i,j)=A(i)/(1-A(j)) qaydası ilə (i,j=1,...,n) B matrisini qurmalı.

695. n natural ədədi və n ölçülü A vektoru verilib. Əgər A(I)>0 olarsa B(i)=A(I), əks halda B(i)=A(i)*A(i) qəbul etməklə, B massivini qurmalı.

696. n natural ədədi və n×n ölçülü A matrisi verilib. Matrisin baş diaqonalında və ondan yuxarıda duran bütün elementləri sıfırla əvəz etməli.

697. n natural ədədi və n×n ölçülü A matrisi verilib. Matrisin sütün elementlərinin cəmindən ibarət B massivini qurmalı.

698. n natural ədədi və n×n ölçülü A matrisi verilib. Matrisin baş diaqonal elementlərindən ibarət B massivini qurmalı.

699. n natural ədədi və n×n ölçülü A matrisi verilib. Matrisin baş diaqonal elementlərinə soldan paralel olan əlavə baş diaqonal elementlərinin ən kiçiyini tapmalı.

700. n natural ədədi və n×n ölçülü A matrisi verilib. Matrisin baş diaqonal elementlərinə sağdan paralel olan əlavə baş diaqonal elementlərinin müsbət olanlarının cəmini və sayını tapmalı.

701. n natural ədədi verilib. Baş diaqonal elementləri vahid, yerdə qalan elementləri sıfır olan n×n ölçülü A matrisini qurmalı.

702. n natural ədədi verilib. Baş diaqonal elementləri n,n-1,...,1 olan, qalan elementləri sıfır olan n×n ölçülü matris qurmalı.

703. n natural ədədi verilib. Baş diaqonal elementləri 1*2,2*3,...,n*(n+1) olan, qalan elementləri isə sıfır olan n×n ölçülü matris qurmalı.

704. n natural ədədi verilib. Elementləri A(i,j)=1/(i+j-1) qaydası ilə (i,j=1,...,n) tapılan A matrisini qurmalı.

705. n natural ədədi verilib. Əgər i<j olarsa, B(i,j)=1/(i+j-1), əks halda, B(i,j)=1/(i+j+1) qaydası ilə tapılan B matrisini qurmalı (i,j=1,...,n).

706. n natural ədədi və n- ölçülü A vektoru verilib. Massivin ən böyük elementi ilə onun birinci elementinin yerini dəyişməli.

707. n natural ədədi və n- ölçülü A vektoru verilib. Massivin ən kiçik elementi ilə onun axırıncı elementinin yerini dəyişməli.

708. n natural ədədi və n- ölçülü A vektoru verilib. Bu A massivinə əvvəlcə onun mənfi və sıfır elementlərini, sonra isə müsbət elementini yerləşdirməli.

709. n natural ədədi və n- ölçülü A vektoru verilib. Bu A massivinə əvvəlcə onun müsbət elementlərini, sonra isə sıfır elementlərini yerləşdirməli.

710. n natural ədədi və n×n ölçülü A massivi verilib. Matrisin ən böyük elementi olan sətri ilə, ən kiçik elementi olan sətrlərin yerlərini dəyişməli.

711. n×n ölçülü A matrisi verilib. Matrisin baş diaqonalından yuxarıda qalan hissəsindəki müsbət elementlərin sayını və cəmini tapmalı.

712. n×n ölçülü A Matrisi verilib. matrisin baş diaqonalındakı maksimal qiymətin yerləşdiyi sətirlə, verilmiş m nömrəli sətrin yerlərini dəyişməli.

713. n×m ölçülü A matrisi verilib. Matrisin hər bir sətirində maksimal və minimal elementləri tapıb, onları uyğun olaraq sətrin birinci və sonuncu elementləri ilə əvəz etməli.

714. n×n ölçülü A matrisi verilib. Bu matrisin sehirli kvadrat olduğunu, yəni bütün sətir və sütunlardakı elementlərinin cəminin bir-birinə bərabər olduğunu təyin etməli.

715. n×n ölçülü A matrisi verilib. Matrisin baş diaqonalına nəzərən simmetrik matris olduğunu təyin etməli.

716. n×n ölçülü A matrisi verilib. Matrisin hər bir sətirindəki maksimal elementləri tapıb, onları baş diaqonal elementləri ilə əvəz etməli.

717. n×m ölçülü A matrisi verilib. Matrisin hər bir sətirindəki elementləri onların artımı boyunca düzməli.

718. n×m ölçülü A matrisi verilib. Matrisin hər bir sətrindəki minimal qiymətləri tapıb, onlar arasında maksimalını təyin etməli.

719. n×n ölçülü A matrisi verilib. Matrisin baş diaqonal elementlərinin cəmini tapıb, matrisin cüt nömrəli sətirlərini bu cəmə bölməli, tək nömrəli sətirləri isə olduğu kimi saxlayıb, alınan matrisi çapa verməli.

720. n×n ölçülü A matrisi verilib. Matrisin maksimal və minimal elementlərinin yerləşdiyi sətirləri və bu sətirdəki elementlərin cəmini çapa verməli.

721. n×m ölçülü A matrisi verilib. Matrisin verilmiş m ölçülü B vektorunun elementləri ilə üst-üstə düşən sətirlərinin nömrələrini təyin etməli, belə sətir yoxdursa, bu barədə məlumat verməli.

722. m×n ölçülü A matrisi verilib. Matrisin cüt nömrəli sətirlərinin minimal elementlərini tapmalı.
723. m×n ölçülü A matrisi verilib. Matrisin elementlərindən heç olmasa biri verilmiş C ədədinə bərabər olan sətirlərin nömrələrini təyin etməli.
724. m×n ölçülü A matrisi verilib. Matrisin sütunlarını k-cı sətir (1≤k≤m) elementlərinin artımı boyunca düzməli.

Çoxluqlar və yazılışlar

725. Aşağıdakı münasibətlərin qiymətlərini hesablamalı:
   1) [2]<>[2,2,2];
   2) ['a','b']=['b','a'];
   3) [2,3,5,7]<=[1,2,3,4,5,6,7];
   4) [7,1,3]<>[2,4,6,8];
   5) [BAKI]<=[BAKI,QUBA];
   6) [7,1,2,3,4,5,6]=[1,2,3,4,5,6,7].
726. Aşağıdakı ifadələrin qiymətlərini hesablamalı:
   1) TRUNC(3,9) IN [1,3,5];
   2) SUCC('C') IN ['B','C','D'];
   3) 16 IN [15,16];
   4) 'A' IN ['A','B','C'];
   5) ROUND(4,7) IN [3,4,5];
   6) PRED('B') IN ['A','B','C'].
727. Aşağıdakı ifadələrin qiymətlərini hesablamalı:
   1) [1,3,5]+[2,4];
   2) [2,4,6,8]*[3,5,7];
   3) [1,2,3,4,5]-[2,4];
   4) [ ]+[4];
   5) [1,2,3,4,5,6]*[3,4,5,6,7,8];
   6) [2,3,4]-[1,2,3,4,5].
728. Aşağıdakı ifadələrin qiymətlərini hesablamalı:
   1) ['A','B','C']+['B','E'];
   2) ['B','C','D']*['B','C'];
   3) ['L','K','M']-['A','L','B','M'];
   4) ['B','F']+['B','C','D'].
729. Aşağıdakı ifadələrin qiymətlərini hesablamalı:
   1) [2,4]+[1,2,3,4,5]*[1,3,5];
   2) [4,5,6,7]-[1,4,6,7]+[1,3,7];
   3) ['A','B','C']+['D','E']*['A','C'].
730. N simvoldan ibarət mətn verilib. Bu mətnə verilmiş 'A','B','C' simvollardan hansılarının daxil olduğunu təyin etməli.
731. N simvoldan ibarət mətn verilib. Bu mətndə hansı rəqəmlərin olduğunu təyin etməli. Mətnin sonu nöqtə işarəsi ilə tamamlanır və bu işarə ilə təyin olunur.

732. N - tam ədədi verilib. 1,...,N - tam ədədlər çoxluğundan tək və cüt ədədlərdən ibarət çoxluqlar ayırmalı.

733. N simvoldan ibarət mətn verilib. Bu mətndə rast gəlinən «0»-dan «9»-a qədər rəqəmlərdən ibarət çoxluq qurmalı.

734. N simvoldan ibarət mətn verilib. Bu mətndə rast gəlinən «A»-dan «F»-ə qədər hərflərdən ibarət çoxluq yaratmalı.

735. N - tam ədədi verilib. 1,...,N tam ədədləri çoxluğundan 3-ə və 5-ə qalıqsız bölünən ədədlərdən ibarət çoxluqlar ayırmalı.

736. N tam ədədi verilib. 1,...,N tam ədədləri çoxluğundan 2-yə və 3-ə qalıqsız bölünən ədədlərdən ibarət çoxluqlar ayırmalı.

737. N simvollardan ibarət mətn verilib. Bu mətndə rast gəlinən «A»-dan «Z»-ə qədər hərf və «0»-dan «9»-a qədər rəqəmlərdən ibarət çoxluq yaratmalı.

738. N simvollardan ibarət mətn verilib. Mətndə rast gəlinən rəqəm və cəbri əməl işarələrindən ibarət çoxluq yaratmalı.

739. n tam ədədi verilib. 2,...,n tam ədədləri çoxluğundan 2-yə bölmə nəticəsində alınan qalıq həddlərindən ibarət çoxluq ayırmalı.

740. n tam ədədi verilib. 3,...,n tam ədədləri çoxluğundan 3-ə bölmə nəticəsində alınan qalıq hədlərindən ibarət çoxluq ayırmalı.

741. Aşağıdakı anlayışların ifadə edilməsi üçün uyğun yazılış tiplərini təsvir etməli:

1) zaman - saat, dəqiqə və saniyə;
2) vaxt - gün, ay və il;
3) ünvan - şəhər, küçə, ev, mənzil;
4) seminar - fənn, müəllim, qrupun nömrəsi, həftənin günü, məşğələnin saatı, auditoriya.

742. n sayda tələbənin soyadı, anadan olduğu il, ali məktəbə daxil olduğu il və sessiyada üç fəndən aldığı qiymətlər verilir. Tələbələrin siyahısını çıxarmalı və əlaçıların siyahısını ayırmalı.

743. n sayda tələbə haqqında anket məlumatlar (mis. 742) və sessiyada üç fəndən aldığı qiymətlər verilir. Yalnız yaxşı qiymətlər almış tələbələrin siyahısını çıxarmalı.

744. n sayda tələbə haqqında anket məlumatlar (mis. 742) və sessiyada üç fəndən aldığı qiymətlər verilir. Soyadı A hərfi ilə başlayan tələbələr haqqında məlumatları çıxarmalı.

745. n sayda tələbə haqqında anket məlumatlar (mis. 742) və sessiyada üç fəndən aldığı qiymətlər verilir. Tələbələrin sessiya üzrə orta qiymətini tapmalı və onlar haqqında məlumatla birgə çıxarmalı.

746. n sayda tələbənin soyadı və sessiyada üç fəndən aldığı qiymətlər verilir. Tələbələrin sessiya üzrə orta qiymətini tapmalı və onların siyahısını orta qiymətin azalması boyunca düzməli.

747. n sayda tələbnin soyadı və sessiyada üç fəndən aldığı qiymətlər verilir. Tələbələrin anadan olduğu illərin artımı boyunca siyahısını çıxarmalı.

748. Çoxhədli verilib. X dəyişənin oxşar həddlərini tapmalı və onların əmsallarının cəmini hesablamalı.

749. Tələbələrin soyadı və anadan olduqdarı illər verilmiş siyahıdan, soyadları

"B" hərfi ilə başlayan tələbələrin soyadlarını və təvəllüdlərini çapa verməli.

750. Tələbələrin soyadı və fərdi işdən aldıqları ballar verilmiş siyahıdan, soyadları "K" ilə başlayan tələbələrin soyadlarını və orta ballarını çapa verməli.

751. İşçilərin soyadı və aylıq əmək haqları verilmiş siyahıdan soyadı "M" hərfi ilə başlayan işçilərin soyadlarını və aylıq məvaciblərini çapa verməli.

752. Konfetlərin adları, qiyməti və son istifadə tarixləri verilmiş siyahıdan, adları "A" hərfi liə başlayan konfetlərin adlarını, qiymətini və son istifadə tarixlərini çapa verməli.

753. Tələbələrin soyadı və semestr ərzində aldıqları qiymətlər verilmiş siyahıdan, yalnız 4 və 5 qiymətlər almış tələbələrin soyadlarını çapa verməli.

754. İnformatika üzrə olimpiadada iştirak etmiş tələbələrin soyadları və yığdıqları ballar verilmiş siyahıdan, 30 baldan yüksək bal yığmış tələbələrin soyadlarını çapa verməli.

755. Əhalinin siyahıya alma nəticələri verilmiş siyahıdan 1990-cı ildə anadan olmuş şəxslərin soyadlarını və onların ümumi sayını çapa verməli.

756. Tələbələrin soyadları və imtahan sessiyasında aldığı qiymətlər verilmiş siyahısından yüksək təqaüd alacaq (yalnız 5 qiymətləri olanlar), adi təqaüd alacaq (yalnız 4 və 5 qiymətləriolanlar) və təqaüd almayacaq (3 qiyməti olanlar) tələbələrin soyadlarını çapa verməli.

757. Aptekdə olan dərmanların adları, qiymətləri, son istifadə tarixləri, sayı verilən siyahıdan, ən bahalı və ən ucuz dərmanların adlarını, aptekdəki dərmanların ümumi sayını, ümumi dəyərini çapa verməli.

758. Kursda tədris olunan fənlərin adları, bu fənləri keçən müəllimlərin soyadları, bu dərslərin keçirildiyi auditoriyaların nömrələri verilmiş siyahıdan, "EHM və proqramlaşdırma" fənnini tədris edən müəllimin soyadını və bu fənnin keçildiyi auditoriyanın nömrəsini çapa verməli.

Funksiya və prosedurlar. Fayllar

759. `n,m` - tam ədədləri verilib. Hesablamalı: `(n!+1)+(n!/m!)-(m-n)!`

760. `n,m` - tam ədədləri verilib. Hesablamalı: `((n+m)!/(1-n!))+m!`

761. Üçbucaq təpə nöqtələrinin müstəvi üzərindəki koordinatorları ilə verilib. Onların perimetrlərini hesablamalı.

762. `y=sin(x)+2x` funksiyasının, x arqumentinin [a,b] parçasında `h` addımı ilə aldığı qiymətlər üçün qiymətlər cədvəlini qurmalı.

763. $y=x^2+5$ funksiyasının `x`-in `[a,b]` parçasında `h` addımı ilə aldığı qiymətlər üçün qiymətlər cədvəlini qurmalı.

764. $f(x)=(x+y)^{-5}-(x-2*y)^3$ funksiyasının `x,y` arqumentlərinin həqiqi qiymətləri üçün qiymətini tapmalı.

765. $f(x)=x^3+2*y^2-(x*y)^{-3}$ funksiyasının `x,y` arqumentlərinin həqiqi qiymətləri üçün qiymətini tapmalı.

766. $f(x)=(x-y)^{-3}+(2*x+1)^6-y^2$ funksiyasının x,y arqumentlərinin həqiqi qiymətləri üçün qiymətini tapmalı.

767. Müxtəlif simvollardan ibarət sətir verilib. Sətrdəki nöqtələrin sayını tapmalı.

768. Müxtəlif simvollardan ibarət sətir verilib. Sətrdəki «A» simvollarının sayını tapmalı.

769. `x,y` həqiqi ədədləri verilib. Hesablamalı:
`(max(2x, (x+y)/2)+max(x-y,y))/max(x,y)`.

770. x, y həqiqi ədədləri verilib. Hesablamalı:
$\min(x^2, y) + (1 - \min(2xy, x+y))$.

771. `x,y,z` tam ədədləri verilib. Hesablamalı:
`max(x,2y,z)+max(x-y,1-z,x*z)`.

772. `x,y,z` tam ədədləri verilib. Hesablamalı:
`min(x-1,y/z,2z+2)/min(x+y,y-1,z)`.

773. Üç natural ədəd verilib. Onların qarşılıqlı sadə ədəd, yəni vahiddən başqa ortaq bölənə malik olmayan ədədlər olub-olmadığını təyin etməli.

774. `n` natural ədədi verilib. Onun ikilik say sistemində ifadəsini verməli.

775. `n` tam ədədi verilib. n-ci Fibonaççi ədədini tapmalı. Bu ədədlər aşağıdakı qayda ilə tapılır: `f(1)=1, f(2)=1, f(n)=f(n-1)+f(n-2)`.

776. `n`-ölçülü `A` və `m` ölçülü `B` vektorları verilib. Bu massivlərin elementlərinin həm cəmi, həm də hasillərini tapmalı.

777. `n`-ölçülü A vektoru verilib. Massivin birinci elementindən `m`-ci (`m<n`) elementlərinə qədər cəmini və `m`-ci elementindən `n`-ci elementinə qədər cəmini tapmalı.

778. `n`-ölçülü `A` və `B` vektorları verilib. Onların ən kiçik elementlərini tapmalı.

779. `n`-ölçülü `A` və `B` vektorları verilib. Onların müsbət və mənfi ünsürlərinin sayını tapmalı.

780. `n`-ölçülü `A` və `B` vektorları verilib. Onların tək indeksli elementləri və cüt elementlərini ayırmalı.

781. `n×n` ölçülü `A` və `B` matrisləri verilib. Onların izlərini, yəni baş diaqonal elementlərinin cəmisini tapmalı.

782. `n`-ölçülü `A` və `m`-ölçülü `B` vektorları verilib. Bu vektorların uzunluğunu yəni elementlərinin kvadratları cəminin kökünü tapmalı.

783. `n`-ölçülü `A` və `B`, `m`-ölçülü `C` və `D` vektorları verilib. Bu vektor cütlərinin skalyar hasillərini tapmalı.

784. `n`-ölçülü `A` və `B` vektorları verilib. Onların cüt elementlərinin cəmini tapmalı.

785. `n`-ölçülü `A` və `B` vektorları verilib. Onların 3-ə qalıqsız bölünən elementlərinin hasilini tapmalı.

786. `n×n` ölçülü `A` və `B` matrisləri verilib. Onların elementlərinin cəmini tapmalı.

787. `n`-ölçülü `A` və `m` ölçülü `B` vektorları verilib. Onların 3-ə və 5-ə qalıqsız bölünən elementlərinin cəmini tapmalı.

788. Həqiqi ədədlərdən ibarət fayl verilmişdir. Bu fayl elementlərinin kvadratları cəmini tapmalı.

789. Həqiqi ədədlərdən ibarət fayl verilmişdir. Bu fayl elementlərinin hasilinin kvadratını və cəmlərinin mütləq qiymətini tapmalı.

790. Həqiqi ədədlərdən ibarət fayl verilmişdir. Onun elementlərinin ən kiçiyini tapmalı.

791. Həqiqi ədədlərdən ibarət fayl verilmişdir. Onun elementlərinin içində ən böyüyünü tapmalı.

792. Tam ədədlərdən ibarət fayl verilmişdir. Buradakı tək ədədlərin sayını tapmalı.
793. Tam ədədlərdən ibarət fayl verilmişdir. Buradakı cüt ədədlərin hasilini tapmalı.
794. Tam ədədlərdən ibarət fayl verilmişdir. Buradakı 3-ə qalıqsız bölünən elementlərdən ibarət digər fayl qurmalı.
795. Tam ədədlərdən ibarət fayl verilmişdir. Buradakı müsbət ədədlərin cəmini tapmalı.
796. Həqiqi ədədlərdən ibarət fayl verilmişdir. Buradakı mənfi ədədlərin mütləq qiymətcə hasilini tapmalı.
797. Tam ədədlərdən ibarət fayl verilmişdir. Köməkçi fayldan istifadə etməklə bu faylın elementlərini, əvvəlcə müsbət, sonra isə mənfi ədədlər gəlmək şərti ilə digər bir fayla köçürməli.
798. Tam ədədlərdən ibarət fayl verilmişdir. Köməkçi fayldan istifadə etməklə bu faylın elementlərini, əvvəlcə tək, sonra isə cüt ədədlər gəlmək şərti ilə digər bir fayla köçürməli.
799. Proqramlaşdırma fənninin tədrisində ən çox istifadə olunan dərsliklərin adından ibarət fayl qurmalı.
800. Respublika dövlət ali məktəblərinin adlarından ibarət fayl qurmalı.
801. Bakı Dövlət Universitetindəki fakültələrinin adlarından ibarət fayl qurmalı.
802. Həqiqi ədədlərdən ibarət fayl verilmişdir. Bu faylın elementlərini əks düzüm qaydası ilə digər fayla yazmalı.
803. `n` tam ədədi verilib. Hər bir `i`-ci elementi `i*i`-yə bərabər olub, kvadratları verilən `n`-i aşmayan tam ədədlərdən ibarət fayl qurmalı.
804. `n` tam ədədi verilib. Elementləri $b_i=1/i^2$ `(i=1,...,n)` kimi tapılan fayl qurmalı.
805. `a` və `b` natural ədədlərin ƏBOB və ƏKOB-nu hesablamaq üçün proqram qurmalı.
806. `a,b,c,d` natural ədədlərin ƏBOB-nu tapmaq üçün proqram qurmalı.
807. `a,b,c` natural ədədlərin ƏKOB-nu tapmaq üçün proqram qurmalı.
808. `x,y,z` ədədlərinin maksimal və minimal elementlərinin cəminin tapılması üçün proqram qurmalı.
809. 1-9 arasındakı tək ədədlərin faktoriallarının cəmini tapan proqram qurmalı.
810. $\frac{a}{b}$ və $\frac{c}{d}$ (`a,b,c,d` – natural ədədlərdir) kəsrləri verilib. Kəsrin kəsrə bölünməsi üçün proqram qurmalı.
811. $\frac{a}{b}$ və $\frac{c}{d}$ (`a,b,c,d` – natural ədədlərdir) kəsrləri verilib. Kəsrin kəsrə vurulması üçün proqram qurmalı.
812. $\frac{a}{b}$ və $\frac{c}{d}$ (`a,b,c,d` – natural ədədlərdir) kəsrləri verilib. Birinci kəsrdən, ikinci kəsrin çıxılması üçün proqram qurmalı.
813. $\frac{a}{b}$ və $\frac{c}{d}$ (`a,b,c,d` – natural ədədlərdir) kəsrləri verilib. Kəsrlərin toplanması üçün proqram qurmalı.

814. `m` və `n` ədədləri arasındakı cüt ədədlərin faktoriallarının cəmini tapan proqram qurmalı.

815. `n×n` ölçülü `A,B,C` matrisləri verilib. Bu matrislərdən norması ən kiçik olanını çap etməli. Matrisin norması, onun elementlərinin mütləq qiymətlərinin maksimumudur.

816. F faylını təsadüfi tam ədədlərlə doldurub, G faylında F faylındakı cüt ədədləri yerləşdirməli.

817. F faylını təsadüfi tam ədədlərlə doldurub, G faylında F faylındakı verilmiş m ədədinə qalıqsız bölünən ədədləri yerləşdirməli.

818. F faylını təsadüfi tam ədədlərlə doldurub, bu fayldakı maksimal və minimal elementlərin cəmini tapmalı.

819. F faylını təsadüfi tam ədədlərlə doldurub, G faylında F faylının a_i $(i=\overline{1,n})$ elementlərindən $a_1, a_1 \cdot a_2, a_1 \cdot a_2 \cdot a_3, \ldots, a_1 \ldots a_n$ qaydası ilə qurulan ədədləri yerləşdirməli.

820. F faylını təsadüfi tam ədədlərlə doldurub, G faylında F faylında verilmiş K ədədinə qalıqsız bölünən ədədlərdən fərqli ədədləri yerləşdirməli.

821. `F` faylını $a_1 \ldots a_n$ həqiqi ədədləri ilə doldurub, `G` faylında $b_k = \sum_{k=1}^{i} a_k / i$, $i=\overline{1,n}$ düsturu ilə hesablanan ədədləri yerləşdirməli.

Simvollar ardıcıllığı

822. Aşağıdakı ifadələrin qiymətini hesablamalı:

1) `pred('7');` 2) `succ('0')=pred('2') ;`

3) `ord('5')-ord('0');` 4) `succ('9').`

823. **SUMMA** sözünə aid hərflərin sıra nömrələrinin cəmini tapmalı.

824. Sıra nömrələri 65,71 və 69 olan hərflərdən düzələn sözü tapmalı.

825. «A» və «Z» hərfləri arasında, latın hərflərindən başqa digər simvol yoxdursa, `V` dəyişəninə **TRUE** qiymətini, əks halda **FALSE** qiymətini mənimsətməli.

826. «A» və «Z» hərfləri arasında yerləşən bütün hərfləri çap etməli.

827. `N` natural ədədi və `N` sayda simvollar verilib. Bu simvollar arasında neçə «x» simvolunun olduğunu tapmalı.

828. `N` natural ədədi və `N` sayda simvollar verilib. Bu simvollar arasında neçə «x» və neçə «*» simvolu olduğunu tapmalı.

829. «A» və «Z» hərflərini `A,AB,ABC,...,AB...YZ` formasında çıxarmalı.

830. Əgər verilən mətndə «A» hərfi «B» hərfindən çox rast gəlinirsə, **TRUE**, əks halda **FALSE** çıxarmalı.

831. Əgər verilən mətndə **KEY** sözündə olan hərflərin hamısı daxildirsə **YES**, əks halda **NO** sözünü çıxarmalı.

832. `N` natural ədədi və `N` sayda simvollar ardıcıllığı verilib. Bu ardıcıllıqda olan birinci nöqtə işarəsinə qədər olan simvolları çıxarmalı.

833. N natural ədədi və N sayda simvollar ardıcıllığı verilib. Bu ardıcıllıqda olan birinci nöqtə işarəsindən sonra gələn simvolları çıxarmalı.

834. Simvollar ardıcıllığı verilib. Bu ardıcıllıqda olan birinci «*» işarəsinə qədər neçə nida işarəsi olduğunu tapmalı.

835. Simvollar ardıcıllığı verilib. Bu ardıcıllıqda olan birinci nöqtə işarəsinə qədər simvollarda neçə vergül işarəsi olduğunu tapmalı.

836. N natural ədədi və N sayda simvollar ardıcıllığı verilib. Bu ardıcıllıqda vergül və ya nöqtə vergül işarəsindən hansının çox olduğunu təyin etməli.

837. N natural ədədi və N sayda simvollar ardıcıllığı verilib. Bu ardıcıllıqda daxil olan latın hərflərinin sayını tapmalı.

838. N natural ədədi və N sayda simvollar ardıcıllığı verilib. Bu ardıcıllıqda rast gəlinən birinci vergül işarəsinin sıra nömrəsini tapmalı.

839. N natural ədədi və N sayda simvollar ardıcıllığı verilib. Bu ardıcıllıqda rast gəlinən axırıncı nöqtə işarəsinin sıra nömrəsini tapmalı.

840. N natural ədədi və N sayda simvollar ardıcıllığı verilib. Bu ardıcıllıqdakı «A» hərfinin sıra nömrəsini tapmalı.

841. Simvollar ardıcıllığı verilib. Buradakı birinci nöqtə işarəsinə qədər olan hissədə heç olmasa bir latın hərfi olub-olmadığını təyin etməli.

842. Nöqtə işarəsi ilə bitən sətir verilib. Sətirdəki simvolların sayını təyin etməli.

843. Verilmiş sətirdə neçə "b" hərfi olduğunu hesablamalı.

844. Verilmiş sətirdə neçə "k", "t" və "s" hərfi olduğunu hesablamalı.

845. Verilmiş sətirdə neçə "*", ";", ":" simvolunun olduğunu hesablamalı.

846. Tərkibində ":" işarəsi olan verilmiş sətirdə bu işarəyə qədər olan simvolların sayını hesablamalı.

847. Nöqtə işarəsi ilə bitən sətir verilib. Sətirdəki üç hərfdən ibarət sözləri çapa verməli.

848. Verilmiş sətirdə hər bir "*" simvolunu çıxarıb, bu simvoldan fərqli bütün simvolların təkrarını alıb, sətirdə yerləşdirməli.

849. Verilmiş sətirdə neçə abc hərflər qrupu olduğunu hesablamalı.

850. Verilmiş sətirdə bir açılan mötərizə "(" və bir bağlanan mötərizə ")" var. Bu işarələr arasındakı simvolları çapa verməli.

Modullar. Verilənlərin dinamik strukturları

851. Kompleks ədədlər üzərində aşağıdakı əməliyyatların aparılmasını təmin edən alt proqramlardan ibarət modul qurmalı:

1) toplama;
2) çıxma;
3) vurma;
4) bölmə;
5) kompleks ədədin modulu;
6) kompleks ədədin n (n-natural ədəd) tərtibinin təyini.

852. $\frac{P}{Q}$ (P-tam, Q-natural ədəd) adi kəsrləri üzərində aşağıdakı əməliyyatların aparılmasını təmin edən, alt proqramlardan ibarət modul qurmalı:

1) toplama;
2) çıxma;
3) vurma;
4) bölmə;
5) kəsrin ixtisarı;
6) kəsrin n (n-natural ədəd) tərtibinin təyini.

853. Kvadrat matrislər üzərində aşağıdakı əməliyyatların aparılmasını təmin edən, alt proqramlardan ibarət modul qurmalı:

1) iki matrisin toplanması;
2) bir matrisin digərinə vurulması;
3) transponirə olunmuş matrisin təyini;
4) matrisin determinantının hesablanması.

854. Vektorlar üzərində aşağıdakı əməliyyatların aparılmasını təmin edən alt proqramlardan ibarət modul qurmalı:
1) toplama;
2) çıxma;
3) vektorların skalyar hasili;
4) vektorların ədədə vurulması;
5) vektorun uzunluğunun tapılması.

855. P-ci ($2 \le P \le 9$) say sistemində verilmiş natural ədədlər üzərində aşağıdakı əməliyyatların aparılmasını təmin edən alt proqramlardan ibarət modul qurmalı:
1)toplama;
2) çıxma;
3) vurma;
4) bölmə;
5) onluq say sistemindən P-ci say sisteminə keçid;
6) P-ci say sistemindən onluq say sisteminə keçid;
7) münasibət əməliyyatlarını yerinə yetirən funksiyalar.

856. Qraf kimi nöqtələr külliyatını və bu külliyatdakı bəzi nöqtələrini birləşdirən düz xətt parçalarını işarə etsək, alt qraf kimi isə bu qrafın hər hansı alt çoxluğunu qəbul etsək, onda qraflar və alt qraflar üzərində aşağıdakı əməliyyatların aparılmasını təmin edən alt proqramlardan ibarət modul qurmalı:
1) qrafdakı nöqtələrin sayı;
2) qrafdakı düz xətt parçalarının sayı;
3) qrafdakı izolə edilmiş alt qrafların (parçalarla əlaqələndirilməmiş alt qraflar) sayı;
4) qraf – iki qrafın birləşməsi;
5) alt qraf – iki qrafın kəsişməsi;
6) qrafın hər bir təpə nöqtəsindən çıxan düz xətt parçalarının sayı.

857. Birdəyişənli çoxhədlilər üzərində aşağıdakı əməliyyatların aparılmasını təmin edən alt proqramlardan ibarət modul qurmalı:
1) toplama;
2) çıxma;
3) vurma;
4) münasibət əməliyyatları (bərabər, fərqli);
5) natural k tərtibinin alınması;
6) çoxhədlinin törəməsinin hesablanması;
7) x_0 nöqtəsində çoxhədlinin hesablanması.

858. L siyahısına E elementinin hər daxil edilməsi zamanı yeni F elementini daxil edən proqram qurmalı.

859. L siyahısına E elementinin birinci daxil edilməsi zamanı yeni F elementini daxil edən proqram qurmalı.

955. $\begin{cases} 10x_1 - 0{,}17x_2 + 0{,}11x_3 - 0{,}15x_4 = 0{,}17 \\ 0{,}14x_1 + 2{,}1x_2 - 0{,}33x_3 + 0{,}11x_4 = 1{,}00 \\ 0{,}22x_1 + 0{,}34x_2 - 1{,}1x_3 + 0{,}12x_4 = 2{,}00 \\ 0{,}11x_1 + 0{,}13x_2 + 0{,}12x_3 + 1{,}4x_4 = 0{,}13 \end{cases}$

956. $\begin{cases} 10x_1 + 0{,}55x_2 - 0{,}13x_3 + 0{,}34x_4 = 0{,}13 \\ 0{,}13x_1 - 1{,}7x_2 + 0{,}33x_3 + 0{,}17x_4 = 0{,}11 \\ 0{,}11x_1 + 0{,}18x_2 - 2{,}2x_3 - 0{,}11x_4 = 1{,}0 \\ 0{,}13x_1 - 0{,}12x_2 + 0{,}21x_3 + 2{,}2x_4 = 0{,}18 \end{cases}$

957. $\begin{cases} 10x_1 - 0{,}51x_2 + 0{,}12x_3 + 0{,}55x_4 = 0{,}12 \\ 0{,}12x_1 + 1{,}8x_2 - 0{,}22x_3 - 0{,}41x_4 = 0{,}13 \\ 0{,}22x_1 - 0{,}31x_2 + 3{,}1x_3 + 0{,}58x_4 = 1{,}00 \\ 1{,}00x_1 + 0{,}24x_2 - 0{,}30x_3 - 2{,}2x_4 = 3{,}41 \end{cases}$

958. $\begin{cases} 1{,}3x_1 + 0{,}22x_2 - 0{,}14x_3 + 0{,}15x_4 = 1{,}00 \\ 0{,}22x_1 - 3{,}1x_2 + 0{,}42x_3 - 0{,}51x_4 = 6{,}01 \\ 0{,}62x_1 - 0{,}74x_2 + 8{,}5x_3 - 0{,}96x_4 = 0{,}11 \\ 0{,}12x_1 + 0{,}13x_2 + 0{,}14x_3 + 4{,}5x_4 = 0{,}16 \end{cases}$

959. $\begin{cases} 1{,}8x_1 + 0{,}19x_2 + 0{,}20x_3 - 0{,}21x_4 = 0{,}22 \\ 0{,}51x_1 - 5{,}00x_2 - 0{,}49x_3 - 0{,}48x_4 = 0{,}47 \\ 0{,}61x_1 + 0{,}62x_2 - 6{,}3x_3 + 0{,}64x_4 = 0{,}65 \\ 0{,}11x_1 - 0{,}15x_2 + 0{,}22x_3 - 3{,}8x_4 = 0{,}42 \end{cases}$

960. $\begin{cases} 1{,}7x_1 - 0{,}18x_2 + 0{,}19x_3 - 0{,}57x_4 = 1{,}00 \\ 0{,}11x_1 - 4{,}3x_2 + 0{,}15x_3 - 0{,}17x_4 = 1{,}9 \\ 0{,}12x_1 + 0{,}14x_2 + 1{,}6x_3 + 0{,}18x_4 = 2{,}00 \\ 0{,}71x_1 - 0{,}13x_2 - 0{,}41x_3 + 5{,}2x_4 = 1{,}00 \end{cases}$

961. $\begin{cases} 10x_1 - 2{,}01x_2 + 2{,}04x_3 + 0{,}17x_4 = 0{,}18 \\ 0{,}33x_1 - 7{,}7x_2 + 0{,}44x_3 - 0{,}510x_4 = 0{,}19 \\ 0{,}31x_1 + 0{,}17x_2 - 2{,}10x_3 + 0{,}54x_4 = 0{,}21 \\ 0{,}17x_1 + 1{,}00x_2 - 0{,}13x_3 + 2{,}1x_4 = 0{,}31 \end{cases}$

962. $\begin{cases} 23{,}4x_1 - 1{,}42x_2 - 0{,}54x_3 + 0{,}21x_4 = 0{,}66 \\ 1{,}44x_1 - 5{,}3x_2 + 1{,}43x_3 - 1{,}27x_4 = -1{,}44 \\ 0{,}63x_1 - 1{,}32x_2 - 6{,}5x_3 + 1{,}43x_4 = 0{,}94 \\ 0{,}56x_1 - 0{,}88x_2 - 0{,}67x_3 - 23{,}8x_4 = 0{,}73 \end{cases}$

963. $\begin{cases} 6,3x_1 - 0,76x_2 + 1,34x_3 + 0,37x_4 = 1,21 \\ 0,54x_1 + 8,3x_2 - 0,74x_3 - 1,27x_4 = 0,86 \\ 0,24x_1 - 0,44x_2 + 3,5x_3 + 0,55x_4 = 0,25 \\ 0,43x_1 - 1,21x_2 - 2,32x_3 - 14,1x_4 = 1,55 \end{cases}$

964. $\begin{cases} 14,3x_1 + 0,87x_2 - 1,57x_3 - 0,58x_4 = 2,34 \\ 0,63x_1 - 5,7x_2 - 2,34x_3 + 0,66x_4 = 0,77 \\ 1,57x_1 + 0,66x_2 - 5,7x_3 + 1,15x_4 = -0,24 \\ 0,88x_1 - 0,67x_2 + 0,55x_3 - 4,5x_4 = 0,56 \end{cases}$

965. $\begin{cases} 17,1x_1 - 0,83x_2 + 1,44x_3 - 0,72x_4 = 1,35 \\ 0,64x_1 - 8,5x_2 - 0,43x_3 + 0,88x_4 = 0,77 \\ 0,38x_1 + 1,42x_2 + 6,3x_3 - 1,55x_4 = 0,28 \\ 0,83x_1 - 0,66x_2 + 0,58x_3 + 12,2x_4 = -0,47 \end{cases}$

966. $\begin{cases} 8,5x_1 + 1,27x_2 - 2,37x_3 + 0,57x_4 = 1,47 \\ 1,47x_1 - 2,8x_2 + 0,56x_3 - 1,21x_4 = 0,86 \\ 0,66x_1 + 1,31x_2 - 6,3x_3 + 0,43x_4 = -0,55 \\ 0,57x_1 - 0,78x_2 - 0,56x_3 - 8,3x_4 = 0,27 \end{cases}$

967. $\begin{cases} 6,8x_1 + 1,32x_2 - 0,63x_3 - 0,87x_4 = 1,43 \\ 0,57x_1 + 3,6x_2 - 1,24x_3 - 0,23x_4 = 0,33 \\ 0,82x_1 - 0,32x_2 + 14,2x_3 + 1,48x_4 = -0,84 \\ 0,56x_1 - 1,20x_2 - 1,2x_3 - 6,4x_4 = 0,45 \end{cases}$

968. $\begin{cases} 14,2x_1 + 2,34x_2 - 0,88x_3 + 0,53x_4 = 0,72 \\ 0,71x_1 - 11,5x_2 + 0,53x_3 - 0,67x_4 = -0,18 \\ 0,55x_1 - 0,93x_2 - 14,2x_3 + 1,32x_4 = 0,68 \\ 0,44x_1 - 0,25x_2 + 1,92x_3 - 10,8x_4 = 0,43 \end{cases}$

969. $\begin{cases} 1,8x_1 + 0,21x_2 + 0,13x_3 - 0,22x_4 = 0,22 \\ 0,33x_1 - 2,2x_2 - 1,0x_3 + 0,17x_4 = 0,11 \\ -1,0x_1 + 0,11x_2 + 20,0x_3 - 0,45x_4 = 1,00 \\ 0,7x_1 - 0,17x_2 - 0,22x_3 + 3,3x_4 = 0,21 \end{cases}$

970. $\begin{cases} 1,1x_1 - 0,17x_2 + 0,72x_3 - 0,34x_4 = 0,17 \\ 0,81x_1 + 1,2x_2 - 0,91x_3 + 0,17x_4 = 1,0 \\ 0,17x_1 - 0,18x_2 + 10,0x_3 + 0,23x_4 = 0,21 \\ 0,13x_1 + 0,17x_2 - 0,99x_3 + 3,5x_4 = 2,71 \end{cases}$

971. $\begin{cases} 13,2x_1 - 0,83x_2 - 0,44x_3 + 0,62x_4 = 0,68 \\ 0,83x_1 + 4,2x_2 - 0,56x_3 + 0,77x_4 = 1,24 \\ 0,58x_1 - 0,37x_2 + 12,4x_3 - 0,62x_4 = 0,87 \\ 0,35x_1 + 0,66x_2 - 1,38x_3 - 9,3x_4 = -1,08 \end{cases}$

972. $\begin{cases} 7,3x_1 + 1,24x_2 - 0,38x_3 - 1,43x_4 = 0,58 \\ 1,07x_1 - 7,7x_2 + 1,25x_3 + 0,66x_4 = -0,66 \\ 1,56x_1 + 0,66x_2 + 14,4x_3 - 0,87x_4 = 1,24 \\ 0,75x_1 - 1,22x_2 - 0,83x_3 + 3,7x_4 = 0,92 \end{cases}$

973. $\begin{cases} 14,2x_1 + 0,32x_2 - 0,42x_3 + 0,85x_4 = 1,32 \\ 0,63x_1 - 4,3x_2 + 1,27x_3 - 0,58x_4 = -0,44 \\ 0,84x_1 - 2,23x_2 - 5,2x_3 - 0,47x_4 = 0,64 \\ 0,27x_1 + 1,37x_2 + 0,64x_3 - 12,7x_4 = 0,85 \end{cases}$

974. $\begin{cases} 6,4x_1 + 0,72x_2 - 0,83x_3 + 4,2x_4 = 2,23 \\ 0,58x_1 - 8,3x_2 + 1,43x_3 - 0,62x_4 = 1,71 \\ 0,86x_1 + 0,77x_2 - 18,3x_3 + 0,88x_4 = -0,54 \\ 1,32x_1 - 0,52x_2 - 0,65x_3 + 12,2x_4 = 0,65 \end{cases}$

975. $\begin{cases} 22x_1 - 3,17x_2 + 1,24x_3 - 0,84x_4 = 0,46 \\ 1,5x_1 + 21,1x_2 - 0,45x_3 + 1,44x_4 = 1,5 \\ 0,86x_1 - 1,44x_2 + 6,2x_3 + 0,28x_4 = -0,12 \\ 0,48x_1 + 1,25x_2 - 0,63x_3 - 9,7x_4 = 0,35 \end{cases}$

976. $\begin{cases} 11,5x_1 + 0,62x_2 - 0,83x_3 + 0,92x_4 = 2,15 \\ 0,82x_1 - 5,4x_2 + 0,43x_3 - 0,25x_4 = 0,62 \\ 0,24x_1 + 1,15x - 3,3x_3 - 1,42x_4 = -0,62 \\ 0,73x_1 - 0,81x_2 + 1,27x_3 - 6,7x_4 = 0,88 \end{cases}$

Aşağıdakı qeyri-xətti tənliklər sistemini **eps=0,001** dəqiqliyi ilə Nyuton üsulu ilə həll etməli.

977. $\begin{cases} \sin(x+1) - y = 1,2 \\ 2x + \cos y = 2 \end{cases}$

978. $\begin{cases} \cos(x-1) + y = 0,5 \\ x - \cos y = 3 \end{cases}$

979. $\begin{cases} \sin x + 2y = 2 \\ \cos(y-1) + x = 0,7 \end{cases}$

980. $\begin{cases} \cos x + y = 1,5 \\ 2x - \sin(y-0,5) = 1 \end{cases}$

981. $\begin{cases} \sin(x+0,5) - y = 1 \\ \cos(y-2) + x = 0 \end{cases}$

982. $\begin{cases} \cos(x+0,5) + y = 0,8 \\ \sin y - 2x = 1,6 \end{cases}$

983. $\begin{cases}\sin(x-1)=1,3-y\\ x-\sin(y+1)=0,8\end{cases}$

984. $\begin{cases}2y-\cos(x+1)=0\\ x+\sin y=-0,4\end{cases}$

985. $\begin{cases}\cos(x+0,5)-y=2\\ \sin y-2x=1\end{cases}$

986. $\begin{cases}\sin(x+2)-y=1,5\\ x+\cos(y-2)=0,5\end{cases}$

987. $\begin{cases}\sin(y+1)-x=1,2\\ 2y+\cos x=2\end{cases}$

988. $\begin{cases}\cos(y-1)+x=0,5\\ y-\cos x=3\end{cases}$

989. $\begin{cases}\sin y+2x=2\\ \cos(x-1)+y=0,7\end{cases}$

990. $\begin{cases}\cos y+x=1,5\\ 2y-\sin(x-0,5)=1\end{cases}$

991. $\begin{cases}\sin(y+0,5)-x=1\\ \cos(x-2)+y=0\end{cases}$

992. $\begin{cases}\cos(y+0,5)+x=0,8\\ \sin x-2y=1,6\end{cases}$

993. $\begin{cases}\sin(y-1)+x=1,3\\ y-\sin(x+1)=0,8\end{cases}$

994. $\begin{cases}2x-\cos(y+1)=0\\ y+\sin x=-0,4\end{cases}$

995. $\begin{cases}\cos(y+0,5)-x=2\\ \sin x-2y=1\end{cases}$

996. $\begin{cases}\sin(y+2)-x=1,5\\ y+\cos(x-2)=0,5\end{cases}$

997. $\begin{cases}\sin(x+1)-y=1\\ 2x+\cos y=2\end{cases}$

998. $\begin{cases}\cos(x-1)+y=0,8\\ x-\cos y=2\end{cases}$

999. $\begin{cases}\sin x+2y=1,6\\ \cos(y-1)+x=1\end{cases}$

1000. $\begin{cases}\cos x+y=1,2\\ 2x-\sin(y-0,5)=2\end{cases}$

1001. $\begin{cases}\sin(x+0,5)-y=1,2\\ \cos(y-2)+x=0\end{cases}$

1002. $\begin{cases}\cos(x+0,5)+y=1\\ \sin y-2x=2\end{cases}$

Diferensial tənliklər üçün qoyulmuş aşağıdakı Koşi məsələlərini `[0,1]` parçasında `h=0,1` addımı ilə Eyler və ya Runqe-Kutta üsulları ilə həll etməli.

1003. $y'=x+y^2,\ y(0)=0,5;$

1004. $y'=2x+y^2,\ y(0)=0,3;$

1005. $y'=x^2+xy,\ y(0)=0,2;$

1006. $y'=x^2+y,\ y(0)=0,4;$

1007. $y'=0,2x+y^2,\ y(0)=0,1;$

1008. $y'=x^2+2y,\ y(0)=0,1;$

1009. $y'=xy+y^2,\ y(0)=0,6;$

1010. $y'=x^2+y^2,\ y(0)=0,7;$

1011. $y'=x^2+0,2y^2,\ y(0)=0,2;$

1012. $y'=0,3x+y^2,\ y(0)=0,4;$

1013. $y'=0,1x+0,2y^2,\ y(0)=0,3;$

1014. $y'=x+0,3y^2,\ y(0)=0,3;$

1015. $y'=2x^2+xy,\ y(0)=0,5;$

1016. $y'=0,1x+2xy,\ y(0)=0,8;$

1017. $y'=x^2+0,2xy,\ y(0)=0,6;$

1018. $y'=3x^2+0,1xy,\ y(0)=0,2;$

1019. $y'=x^2+3xy,\ y(0)=0,3;$

1020. $y'=x^2+0,1y^2,\ y(0)=0,7;$

ƏDƏBİYYAT

1. Əliyev A.Y. İnformatika, hesablama texnikası və proqramlaşdırmanın əsasları. Bakı, Mütərcim, 1998, 216 s.
2. Əliyev A.Y., Piriverdiyev V.Ə. Riyazi analizin təqribi hesablama üsulları. Bakı, Azərb. EA nəşriyyatı, 1993, 139 s.
3. Əliyev A.Y., Piriverdiyev V.Ə. Cəbrin təqribi hesablama üsulları. Bakı, Azərb. EA nəşriyyatı, 1993, 110 s.
4. Əliyev A.Y., Piriverdiyev V.Ə. Diferensial və inteqral tənliklərin təqribi hesablama üsulları. Bakı, İrşad nəş., 1993, 175 s.
5. Mehdiyeva Q.Y., Əliyev A.Y., Piriverdiyev V.Ə. Proqramlaşdırma üzrə məsələlər. Bakı, Bakı Universiteti, 2004, 106 s.
6. Мехтиева Г.Ю., Алиев А.Ю., Пиривердиев В.А. Практикум по программированию. Баку, Бакинский Университет, 2004, 113 с.
7. Абрамов С.А., Гнездилова Г.Г., Капустина Е.Н., Селюн М.И.. Задачи по програмированию М., Наука, 1988, 224 с.
8. Культин Н. C/C++ в задачах и примерах. СПб., БХВ – Петербург, 2001, 288 с.
9. Ляхович В.Ф., Крамаров С.О. Основы информатики. Ростов, Феникс, 2004, 704 с.
10. Петров А.В., Алексеев В.Е., Ваулин А.С. и другие. Вычислительная техника и программирование. М. Высшая школа, 1990, 479 с.
11. Пильщиков В.Н. Сборник упражнений по языку Паскаль. М., Наука, 1989, 160с.
12. Подбельский В.В. Язык Си + +. М. Финансы и статистика. 2001, 560 с.
13. Пярнпуу А.А. Програмирование на современных алгоритмических языках. М., Наука, 1990, 384с.
14. Светозарова Г.И., Мельников А.А., Козловский А.В. Практикум по програмированию на языке Бейсик. М., Наука, 1988, 368с.
15. Семакин И.Г., Шестаков А.П. Основы программирования. М. Академия, 2004, 432 с.
16. Фаронов В.В. Turbo Pascal 7.0. М. изд. ОМД Групп, 2003, 616 с.
17. Фигурнов В.Э. IBM PC для пользователя краткий курс. М., ИНФРА, 1997, 480 с.

Mündəricat

Кандидат физико-математических наук, доцент
Алиев Айдын Юнус оглы

Информатика и программирование

Учебное пособие для высших учебных заведений
Баку, Мутарджим, 2008

Çapa imzalanıb: 16.04.08.
Format: 60x84 1/16. Qarnitur: Times.
Ofset kağızı: əla növ. Həcmi: 25,25 ş.ç.v. Tiraj: 500.
Sifariş №59. Qiyməti müqavilə ilə.

«Mütərcim» Nəşriyyat-Poliqrafiya Mərkəzi
Bakı, Rəsul Rza küç., 125
tel./faks: **596 21 44**
e-mail: mutarjim@mail.ru