Proqramçının kitabxanası

Abdulla Qəhrəmanov, Sevda Sadıqova, İlahə Cəfərova

SCRATCH 2.0

Proqramlaşdırma dili

II

“Proqramçının kitabxanası” seriyasından

A.Qəhrəmanov, S.Sadıqova, İ.Cəfərova

SCRATCH-2

Proqramlaşdırma dili

II hissə

Bakı - 2017

AZƏRBAYCAN RESPUBLİKASI TƏHSİL NAZİRLİYİ

Bu vəsait tamamilə Azərbaycan Respublikası Təhsil Nazirliyinin Təhsildə inkişaf və innovasiyalar üzrə qrant müsabiqəsi çərçivəsində həyata keçirilən layihə çərçivəsində hazırlanmışdır. Təhsil Nazirliyi nəşrin məzmununa görə məsuliyyət daşımır.

Əziz SCRATCH sevən oxucu!

Lifelong Kindergarten MIT (Massachusetts Institute of Technology) Media Lab (Massaçuset Texnologiyalar İnstitutunun Media Laboratoriyasının Ömür boyu Uşaq Bağçası) qrupunun layihəsi olan SCRATCH müasir dövrdə erkən yaşdan proqramlaşdırmanın tədrisi istiqamətində təqdim edilən ən uğurlu layihələrdən biridir. SCRATCH kiçik yaşlı uşaqlara proqramlaşdırmanı öyrətmək üçün nəzərdə tutulmuş pulsuz yeni tədris vasitəsidir.

2016-cı ildə Sizə təqdim edilmiş Azərbaycan dilində hazırlanmış "SCRATCH 2.0 proqramlaşdırma dili. I hissə" kitabının davamı olan bu kitab "SCRATCH 2.0 proqramlaşdırma dili. II hissə" adlanır və dilin sərbəst öyrənilməsi üçün nəzərdə tutulmuş daha çətin proqramların alqoritminin təsviri və proqram kodlarını təsvir edir. Müəlliflər bu vəsaiti də hazırlayarkən dünyada mövcud ingilis, türk və rus dilində hazırlanmış oxşar bir çox vəsaitləri araşdırmışlar. Təqdim olunan bu vəsait də dünya təcrübəsi nəzərə alınmaqla hazırlanmışdır.

*Vəsaitdə verilən material yaşı **7 - 14** arasında olan, oxumağı bacaran, kompüteri idarə etmək üzrə baza bacarıqlarına malik olan uşaqlar üçün nəzərdə tutulub.*

Yuxarıda deyildiyi kimi təqdim edilən bu vəsait SCRATCH 2.0 proqramlaşdırma dilinin sərbəst öyrənilməsi üçün Azərbaycan dilində hazırlanmış ilk vəsaitin II hissəsidir. I hissədə əsasən SCRATCH 2.0 proqramlaşdırma dilinin operatorları ilə tanış olmaq, nisbətən sadə proqramlar yazmaq və sadə alqoritmləri bu proqramlaşdırma dilində necə realizə etmək nəzərdə tutulmuşdur.

Sizə təqdim edilən II hissəsində isə SCRATCH 2.0 proqramlaşdırma dilində daha çətin alqoritmlərin necə realizə edilməsi nümunələrlə verilibdir.

Əziz oxucumuz, əgər kitabda nəyisə başa düşməsən, onda aşağıdakı elektron poçt və ya telefon nömrəsi vasitəsilə bizimlə əlaqə saxla və suallarını bizə çatdır.

e-mail: abdullaqehreman@gmail.com

Telefon: (+99450) 3503920

Hörmətlə: Müəlliflər

Mündəricat

1. SCRATCH nədir?

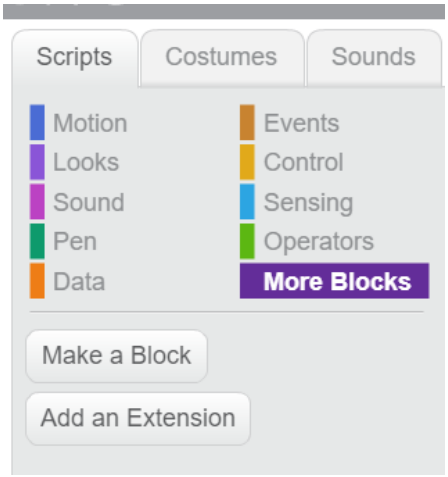
Scratch – Lifelong Kindergarten MIT (Massachusetts Institute of Technology) Media Lab (Massaçuset Texnologiyalar İnstitutunun Media Laboratoriyasının Ömür boyu Uşaq Bağçası) qrupunun layihəsidir.

Scratch proqramının ilk test versiyası Mitçel Reznikin (Mitchel Resnick) rəhbərliyi altında yuxarıda adı çəkilən qrup tərəfindən 2002-ci ildə yaradılmışdır. Daha sonra o təkmilləşdirilmişdir.

Bu kitab 2016-cı ildə çap edilmiş “Scratch 2.0 proqramlaşdırma dili. I hissə” adlı kitabın davamıdır. Bu kitabda bir çox maraqlı proqramlar və onların qurulması alqoritmi təsvir edilmişdir. Düşünürük ki, Sizə maraqlı olacaq.

2. More blocks (Digər bloklar) qrupu ilə iş

Bildiyimiz kimi bütün proqramlaşdırma dillərində altproqramlar və prosedurlar yaratmaq üçün imkanlar var. Scratch 2.0 proqramlaşdırma dilinin 10 əmrlər qrupundan sonuncusu **Digər bloklar (More blocks)** qrupu bu funksiyanı yerinə yetirmək üçün nəzərdə tutulub. Bu qrup seçilmiş sprayt üçün istifadəçi prosedurlarını yaratmağa imkan verir. Bu prosedurlar bənövşəyi rəngdə olurlar.



Proqram yenidən başladılanda birinci dəfə bu bloku seçəndə ekranda 2 düymə görünür:

- **Make a Block (Blok yarat)**
- **Add an Exension (Bir uzantı əlavə et)**

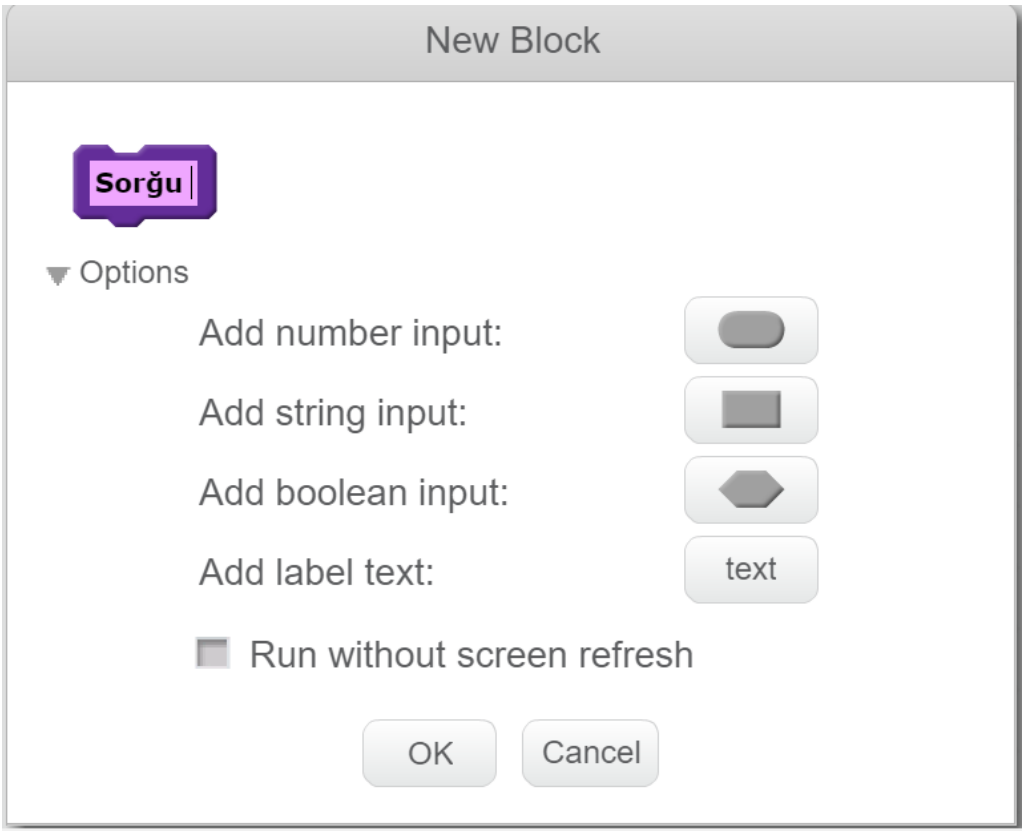
Hər hansı bir digər blok yaratmamışdan əvvəl **Make a Block** düyməsini sıxın. Bu düyməni sıxdıqda aşağıdakı pəncərə açılır.

Bu pəncərədə **Options (Parametrlər)**, **OK (Təsdiq)** və **Cancel (İmtina)** düymələri var. İşi davam etməmişdən əvvəl **Options** blokuna ad verməliyik. Tutaq ki, **Sorğu** olmalıdır. Ad verdikdən sonra nəticə belə olar:

Bloka ad verdikdən sonra **Options** düyməsinə sıxırıq və sonra aşağıda açılan **New block (Yeni blok)** pəncərəsindəki parametrlərdən istifadə etməklə blokumuzu istədiyimiz formada yaradırıq. Sonra ya **OK** düyməsini sıxmaqla seçimimizi təsdiq edirik, ya da **Cancel** düyməsini sıxmaqla seçim rejimindən imtina edirik.

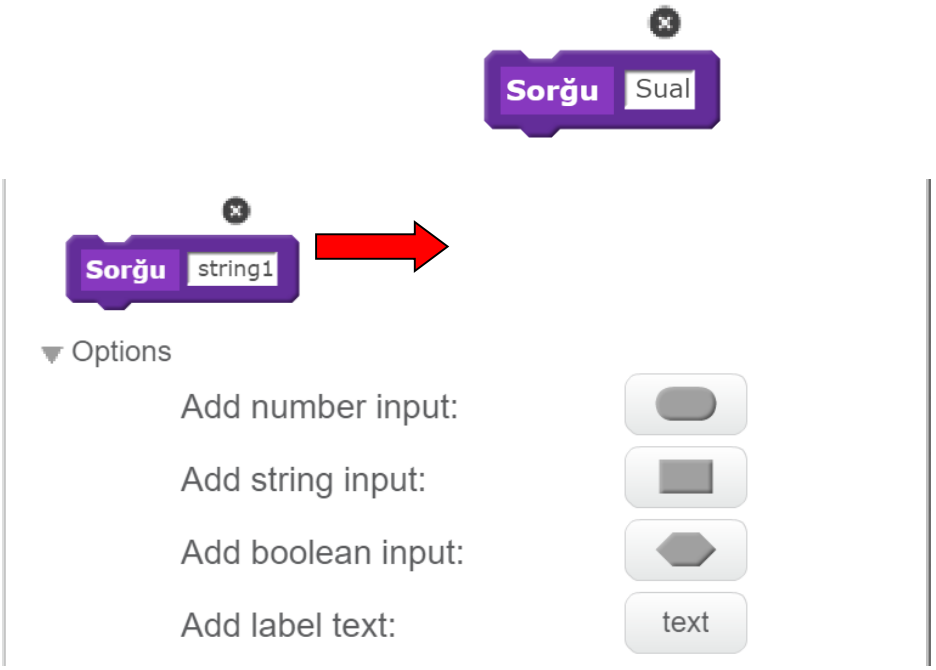
New block (Yeni blok) pəncərəsində yalnız **Options (Parametrlər)** düyməsi var. **Options** düyməsini sıxdıqda sağda seçim üçün aşağıdakı əlavə bloklar görünür:

- Add number input: (Ədədi sahə əlavə etmək)
- Add string input: (Sətir sahəsi əlavə etmək)
- Add boolean input: (Məntiqi sahə əlavə etmək)
- Add label input: (Ad mətni əlavə etmək)
- Run without screen refresh (Ekranı yeniləmədən icra etmək)

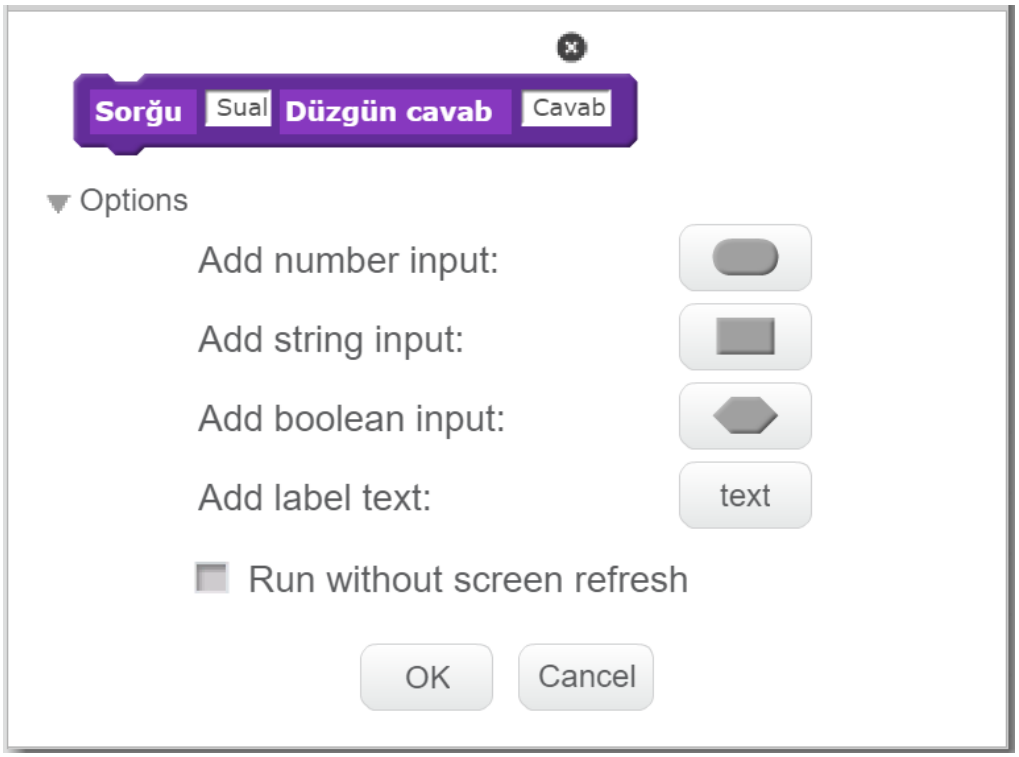


Tutaq ki, bizə belə bir blok yaratmaq lazımdır:

Bunun üçün əvvəlcə **String (Sətir)** əlavə blokunu götürüb ona ad veririk.



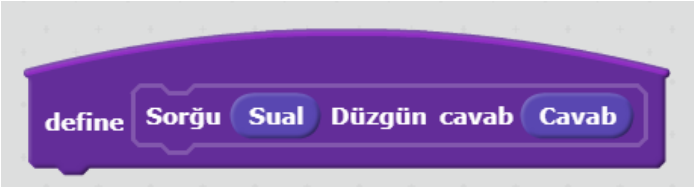
Daha sonra **label text (Ad mətni)** blokunu seçib **Düzgün cavab:** adlandırırıq. Sonda yenə də **String (Sətir)** əlavə blokunu seçib onu **Cavab** adlandırırıq.



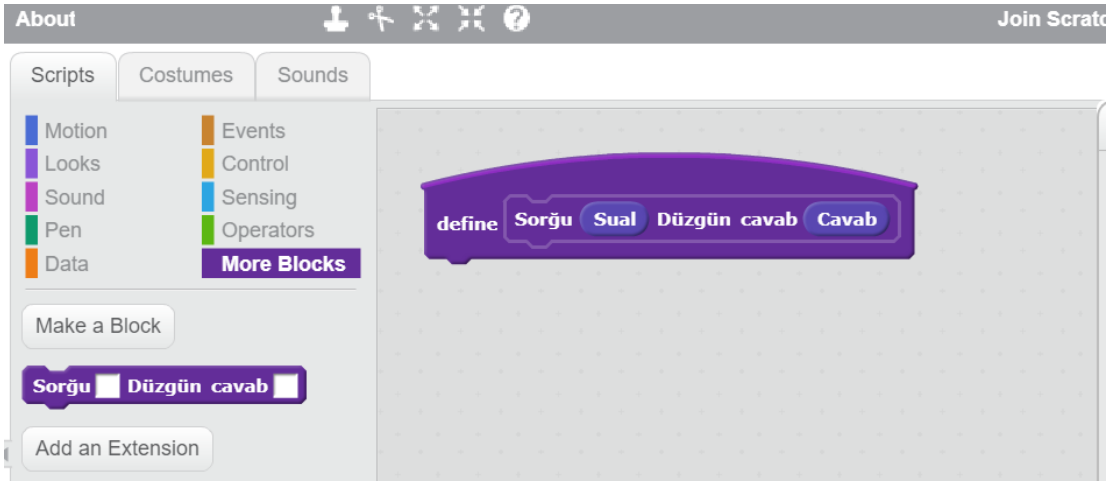
Əgər blokların yığmı zamanı hansısa səhvə yol vermişiksə, onda yığımdakı silinməli blokun üzərində ⊗ düyməsini sıxmaqla onu silmək olar.

OK düyməsini sıxıb bloku yadda saxlayırıq. Bu halda **Digər bloklar (More blocks)** qrupunda

bloku və ekranda sağda yerləşən işçi sahədə aşağıdakı başlıq bloku əmələ gəlir.



Biz bu başlığın altında altproqram yaratmaq üçün lazım olan blokları yığmalıyıq.



Növbəti addım kimi **Digər bloklar (More blocks)** qrupunda yaradılan yeni blokdan yeni yazacağımız proqramda necə istifadə ediləcəyi haqqında söhbət açaq.

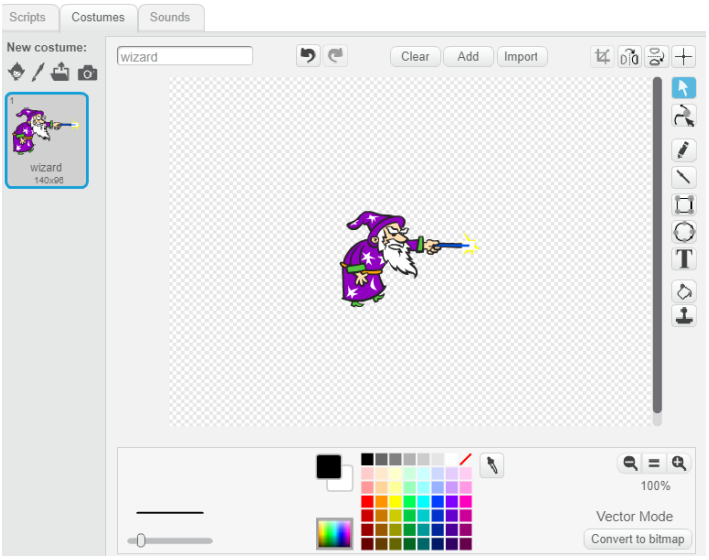
2.1. Əyləncəli viktorina

Tutaq ki, biz məktəb viktorinası üçün proqram yazırıq. Şərt də belədir: Viktorina iştirakçılarına 3 sual verilir. Onların cavabları əsasında düzgün cavabların sayı hesablanır və nəticə elan edilir.

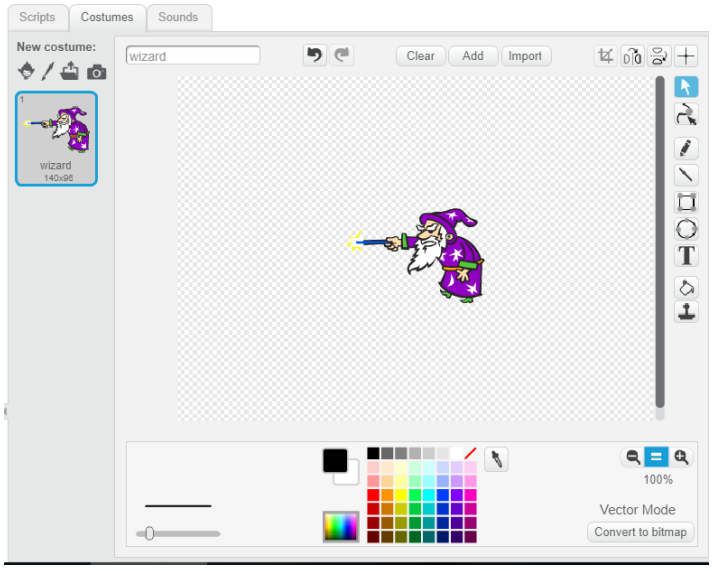
Gəlin personajı (spraytı) dəyişək. Bunun üçün əvvəlcə pişik spraytını silək. Daha sonra spraytlar kitabxanasından **People** qrupundan **Wizard** spraytını seçirik.

Scratch 2.0 proqramının daxili qrafik redaktorunun köməyilə bu spraytın istiqamətini üfüqi istiqamətdə dəyişirik. Bunun üçün seçdiyimiz sprayt üçün **Costumes** düyməsini sıxıb daxili qrafik redaktora daxil oluruq. Qrafik redaktorun

yuxarı sağ küncündə  üfüqi dəyişmə düyməsini sıxıb spraytın istiqamətini dəyişirik.



İstiqamət dəyişənə qədər görünüş.



İstiqamət dəyişəndən sonrakı görünüş.

İstiqaəmətdəyişmə əməliyyatı bitdikdən sonra **Script** düyməsini sıxıb yenidən bloklarla işləmək rejiminə qayıdırıq.

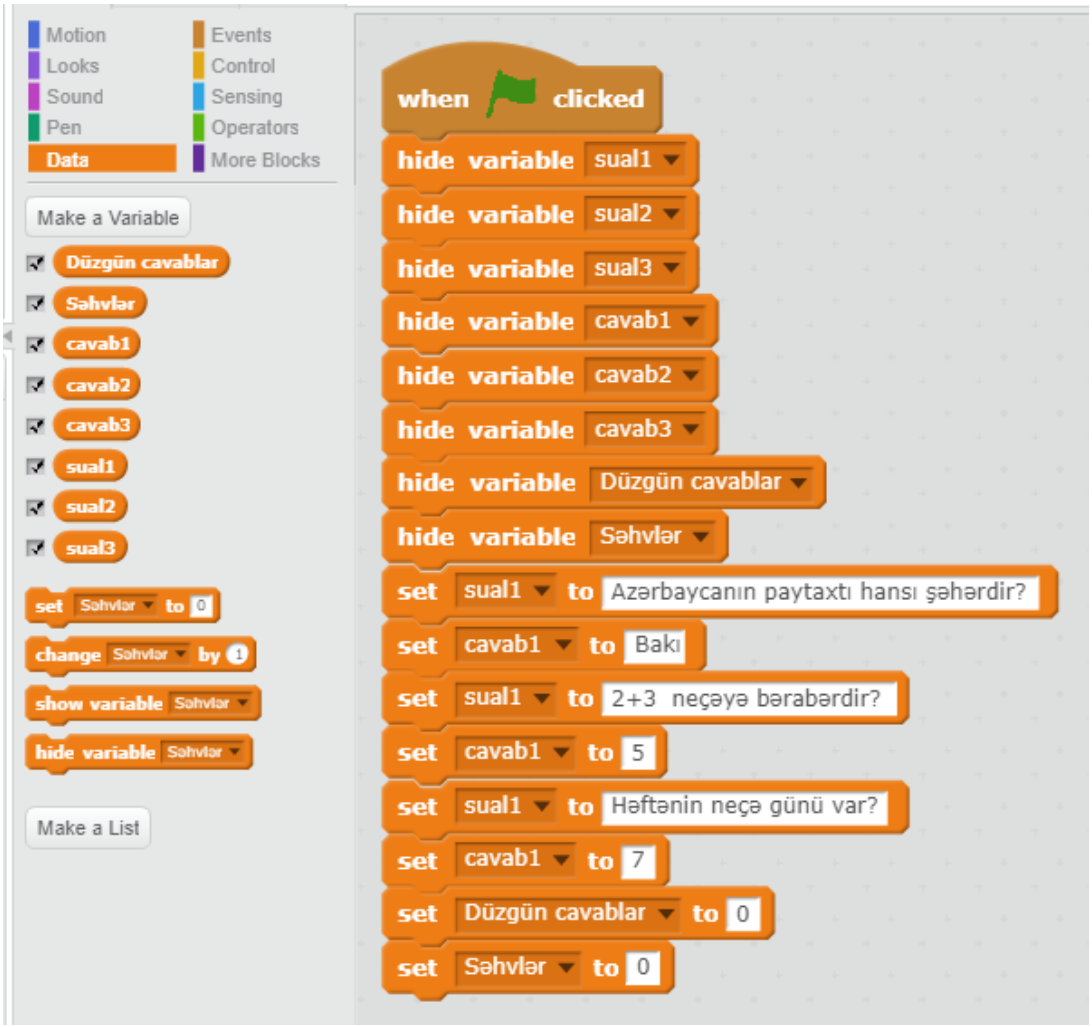
Bundan sonra layihəmiz üçün proqramı yazmağa başlayırıq.

Əvvəlcə biz **Data** qrupunda **sual1**, **sual2** və **sual3** adlı 3 dəyişən yaradırıq. Daha sonra **cavab1**, **cavab2** və **cavab3** adlı daha 3 dəyişən yaradırıq. Statistika aparmaq üçün bizə **Düzgün cavablar** və **Səhvlər** adlı daha 2 dəyişən lazımdır.

Yaratdığımız dəyişənlərin qiyməti səhnədə görünməz olsun deyə proqram hissəsində **Data** qrupundakı **hide variable** əmrindən istifadə edirik. Düzdür, bu işi

Data qrupunda həmin dəyişənlərəin adlarını bildirən sətirlərdəki qeydiyyatı götürməklə də olar. Amma biz ciddi bir proqram yazmaq istəyiriksə, onda **Data** qrupundakı **hide variable** əmrindən istifadə etsək daha yaxşı olar. Bu əmrlər aşağıda verilib.

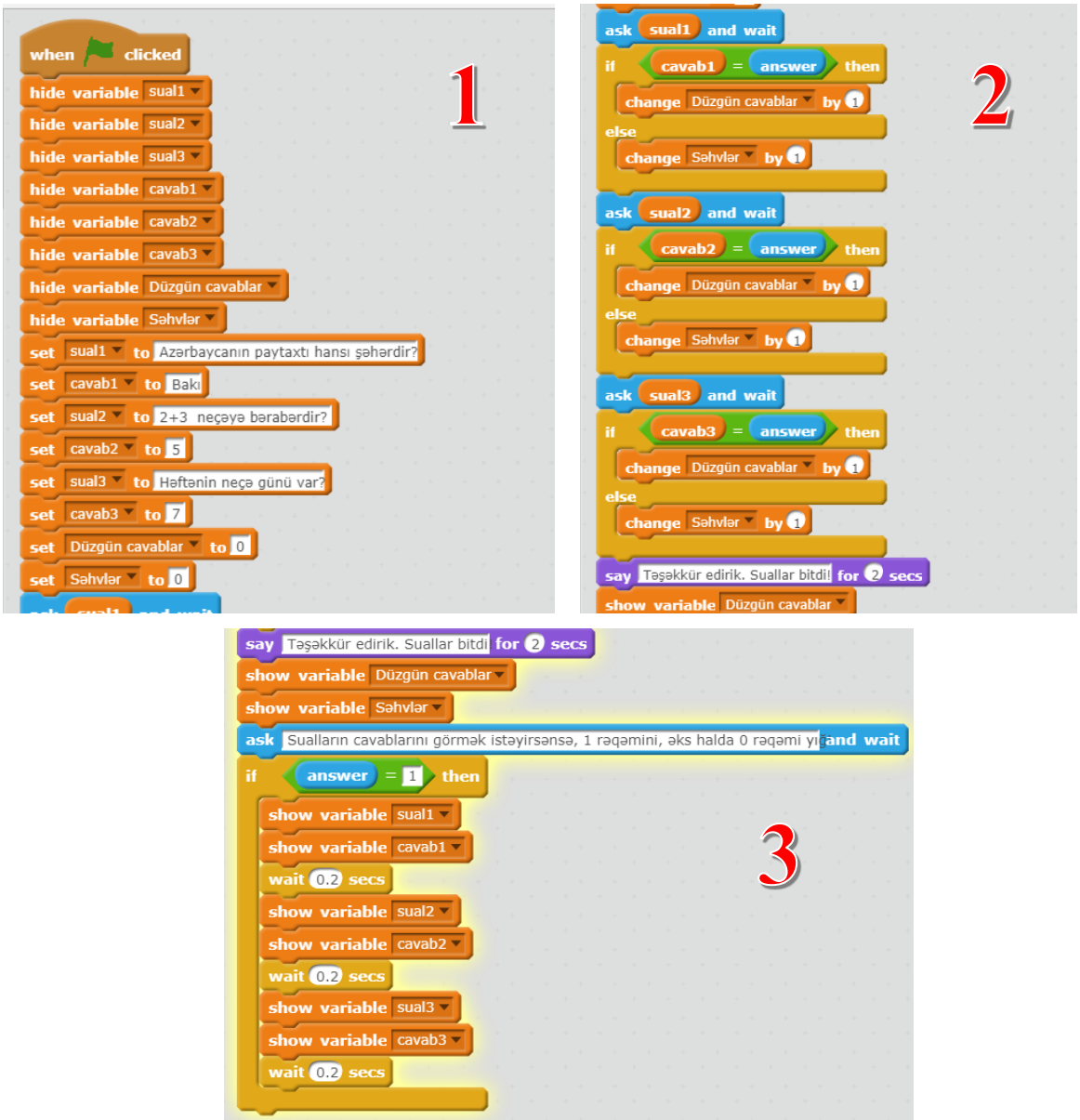
Növbəti addımda suallarımızı, onların cavablarını proqramımıza daxil edək. Statistika aparmaq üçün nəzərdə tutulan **Düzgün cavablar** və **Səhvlər** adlı dəyişənlərə isə başlanğıcda sıfır qiymətini mənimsətmək lazımdır. Bunun üçün sağdakı əmrlər zəncirindən istifadə etmək olar:



Daha sonra proqram növbə ilə ekrana 3 sualı çıxarıb onların cavabını gözləyəcək. Proqram verilən cavablar əsasında düzgün və səhv cavabların sayını hesablayır və statistikanı ekranın yuxarı küncündə çap edir.

Bundan başqa sonda proqram sualları və düzgün cavabları çap etməyə icazə istəyir. Əgər razılıq verilsə, onda həmin məlumat əks olunur.

Proqramın kod hissəsi böyük olduğundan, biz onu 3 hissəyə bölmüşük və və hissələri nömrələmişik.



2.2. Əyləncəli viktorina proqramının optimallaşdırılması

Bu vəsaitin **2.** bəndində Scratch 2.0 proqramlaşdırma dilində altproqramın yaradılmasından söhbət etmişdik. Gəlin **Əyləncəli viktorina** proqramının işini bir qədər optimallaşdıraq. Bəs nədən başlamalı?

Əvvəlcə diqqət edək ki, “Əyləncəli viktorina” proqramını yazarkən gördük ki, onun həcmi çox böyükdür və onu bu kitabda bir səhifədə göstərmək üçün 3 hissəyə ayırıb nömrələdik və sonra səhifəyə yerləşdirdik. İkinci diqqət eləsək görərik ki, proqram daxilində təkrarlanan əmrlər blokları var. Əgər biz təkrarlanmaları azalda və ya qısalda bilsək onda proqramımızı optimallaşdıra bilərik.

Bəs yaxşı, optimal proqram kodu deyəndə nə başa düşməliyik. Yaxşı proqram kodu əmrlərinin sayı minimal olan və digər proqramçı tərəfindən asan başa düşülən proqramdır. Gəlin bu dediklərimizə əsaslanaraq proqramımızı optimallaşdıraq.

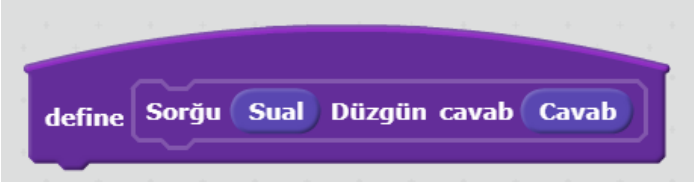
Şəkildən göründüyü kimi, proqram mətnində qırmızı rənglə çərçivəyə alınmış hissələr prinsipcə bir-birinə oxşardır və təkrarlanır.

Bu proqram cəmi 3 suallıq viktorina üçün nəzərdə tutulub. Bəs sulların sayı çox olsa necə? Belə bir proqramın həcmi dəfələrlə böyük olacaq.

Belə hallarda altproqramlardan istifadə etmək olar.

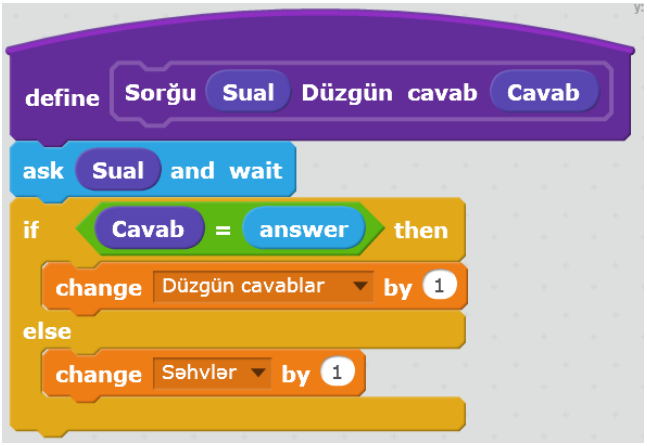
Biz artıq əvvəlki dərslərdə altproqramı necə yaratmaqla tanış olmuşduq. Gəlin biz **Digər bloklar (More blocks)** qrupunda təkrar

blokunu yaradaq. Bu halda ekranda sağda yerləşən işçi sahədə aşağıdakı başlıq bloku əmələ gələcək.

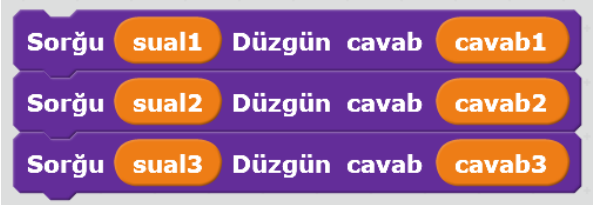


Biz bu başlığın altında altproqram yaratmaq üçün lazım olan blokları yığmalıyıq.

Yuxarıdakı proqramda təkrar olunan proqram kodunu bu başlığın altında yazaq. Nəticə belə olacaq:

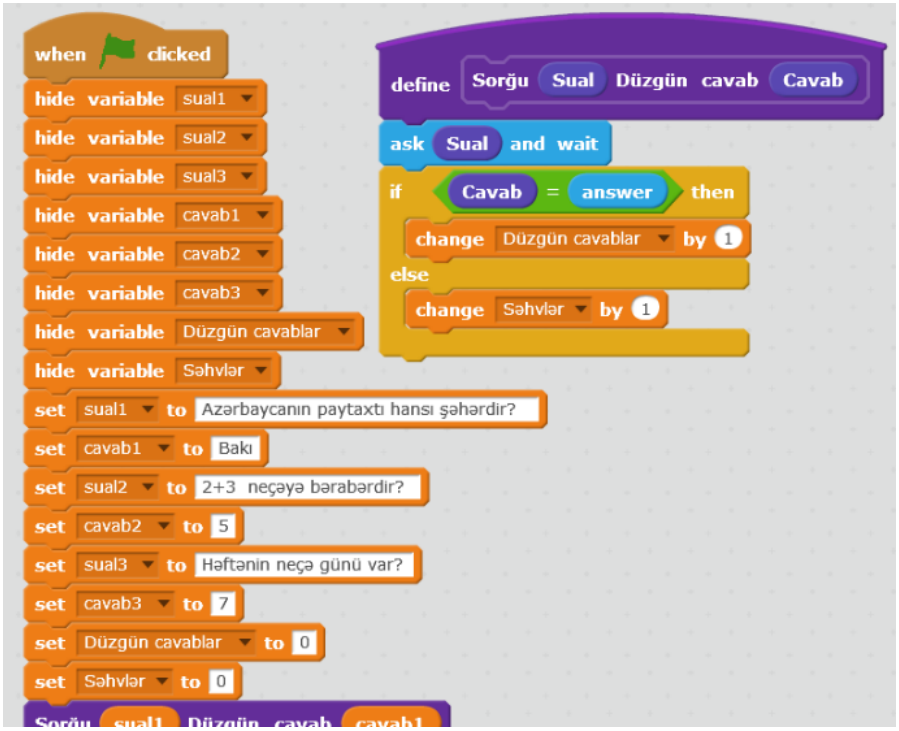


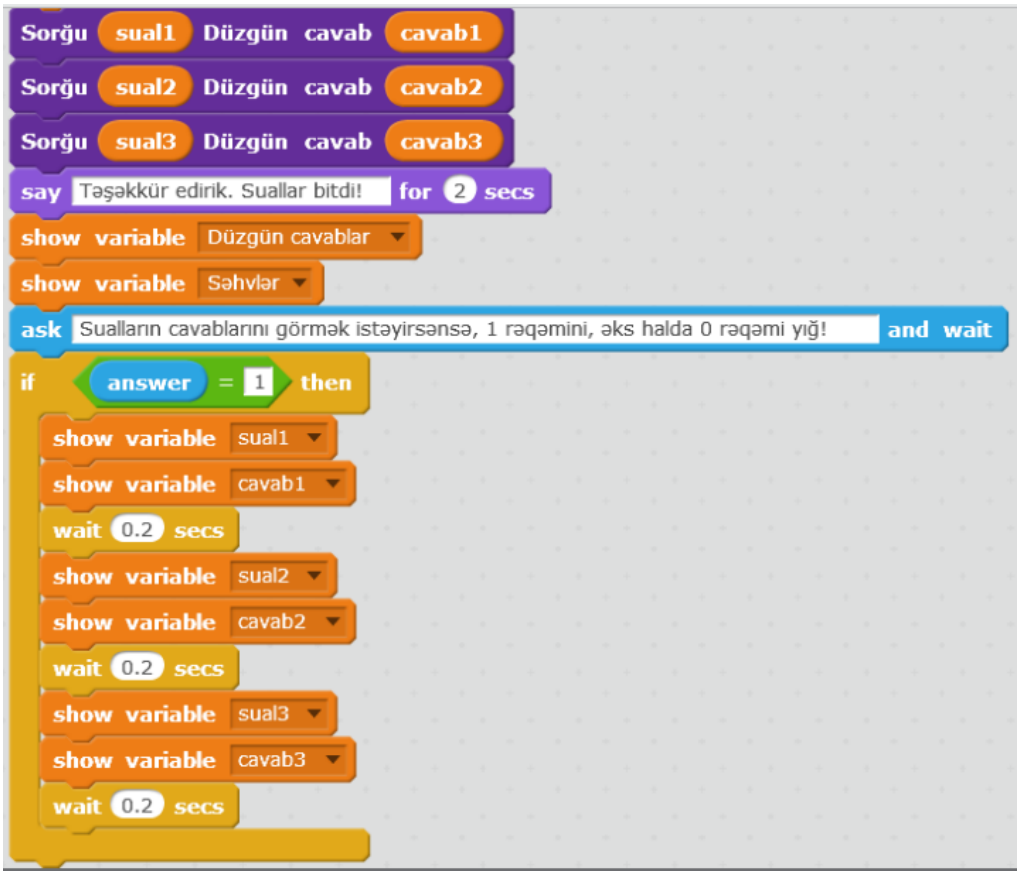
Yuxarıda qırmızı çərçivəyə alınmış proqram hissəsi isə belə olacaq:



3 suallı viktorina proqramı üçün biz 8 əmrə qənaət ediriksə, onda çox suallı viktorina proqramı üçün bu qənaət daha çox əmrə şamil olunacaq.

Proqramın tam mətni aşağıdakı kimi olacaq (2 hissəyə bölünüb):

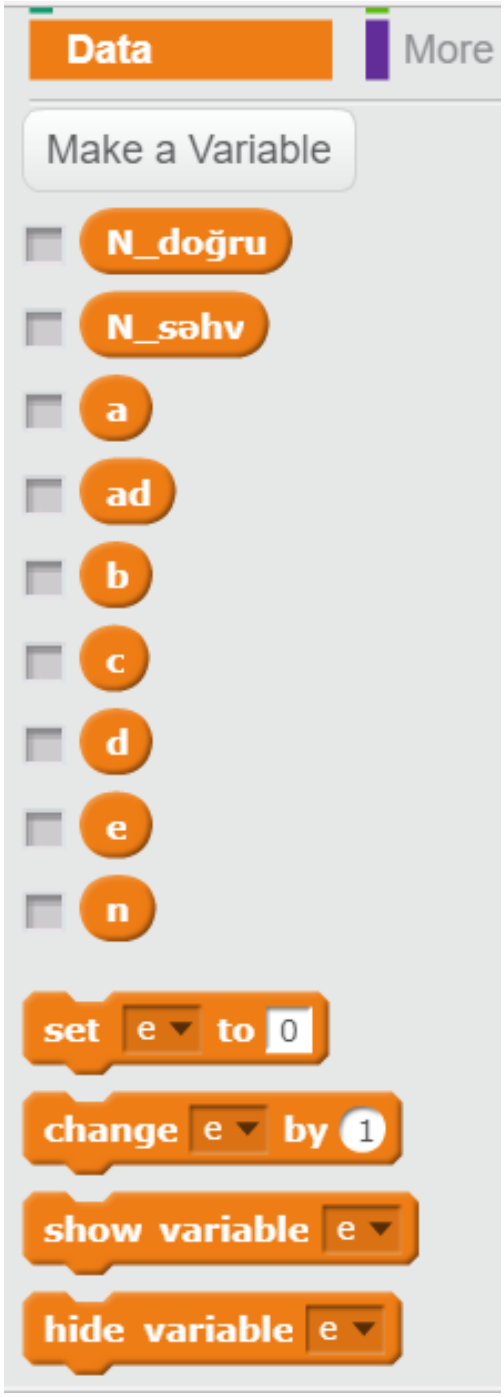




Əlbəttə ki, proqramın digər hissələrini altproqram vasitəsilə optimallaşdıra bilərik. Bunu Sizə sərbəst etməyi tövsiyyə edirik.

3. Toplama

Gəlin riyaziyyatdan toplama bacarığını yoxlayan bir proqram yazaq. Proqramın ssenarisi belə olacaq:



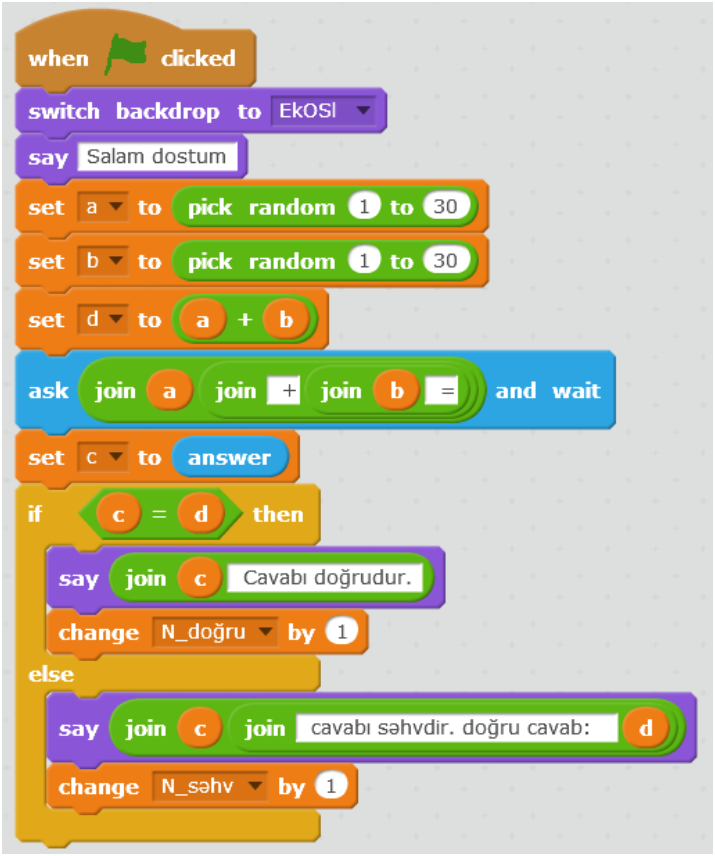
- Əvvəlcə **Scratch 2.0** proqramının **Data** qrupunda soldakı şəkildə gördüyünüz dəyişənlər yaradılacaq.
- Sonra istifadəçinin adı soruşulacaq (**ad** dəyişəni).
- Növbəti addımda istifadəçiyə adı ilə müraciət olunaraq ondan neçə çalışma həll etmək istədiyi soruşulacaq (**n** dəyişəni).
- İstifadəçiyə adı ilə müraciət olunaraq onun indi riyaziyyatdan neçə toplama əməllərini yerinə yetirəcəyi bildiriləcək.
- **N_doğru** (doğru cavabların sayı) və **N_səhv** (səhv cavabların sayı) adlı dəyişənlərinə "**0**" qiyməti mənimsədiləcək.
- Ekranda **n** dəfə təkrarlanmayan **a+b=** tipli çalışma görünəcək. Müxtəlifliyi almaq üçün **Scratch 2.0** proqramının **Operators** qrupundan **pick random** funksiyasından istifadə ediləcək.

- Proqram özü düzgün cavabı tapıb **d** dəyişəninə mənimsədək.
- Daha sonra istifadəçiyə müraciət edib ondan düzgün cavabı daxil etməyi xahiş edəcək və istifadəçinin cavabını **c** dəyişəninə mənimsədəcək.
- Alınan cavabı proqramın hesabladığı cavabla tutşduracaq (**c=d**).
- Cavab doğru olanda bu haqda məlumat verəcək.
- Cavab səhv olanda isə səhv haqqında məlumat verməkdən başqa düzgün cavabı da ekranda əks etdirəcək.

- Cavab düz olanda **N_doğru** dəyişəninin, əks halda **N_səhv** dəyişəninin üzərinə 1 vahid əlavə edəcək.
- Dövr bitdikdən sonra **e** dəyişəninə **N_doğru*100/n** düsturu ilə mənimsəmə faizini mənimsədəcək.
- Ümumi, doğru, səhv çalışmaların sayını və mənimsəmə faizini ekranda çıxaracaq.
- Əgər mənimsəmə faizi 100% olarsa, istifadəçini təbrik edəcək.

Gəlin əvvəlcə personajı (standart **pişik** spraytını) və fonu dəyişək. Bunun üçün spraytlar kitabxanasından **Wizard2** adlı spraytı və fonlar kitabxanasından damalı dəftər vərəqinə oxşayan **EkOSl** fonunu səhnəyə əlavə edək. Ağ fonu və **pişik** spraytını silək.

İndi isə bir sualla işləyən proqram yazaq. Proqram başlayan kimi səhnənin fonunu **EkOSl** fonuna dəyişir və ekranda **Salam dostum** ifadəsini çap edir.



Salamlaşmadan sonra aşağıdakı 2 əmrin köməyilə **a** və **b** dəyişənlərinə ixtiyari qiymətlər mənimsədilir.

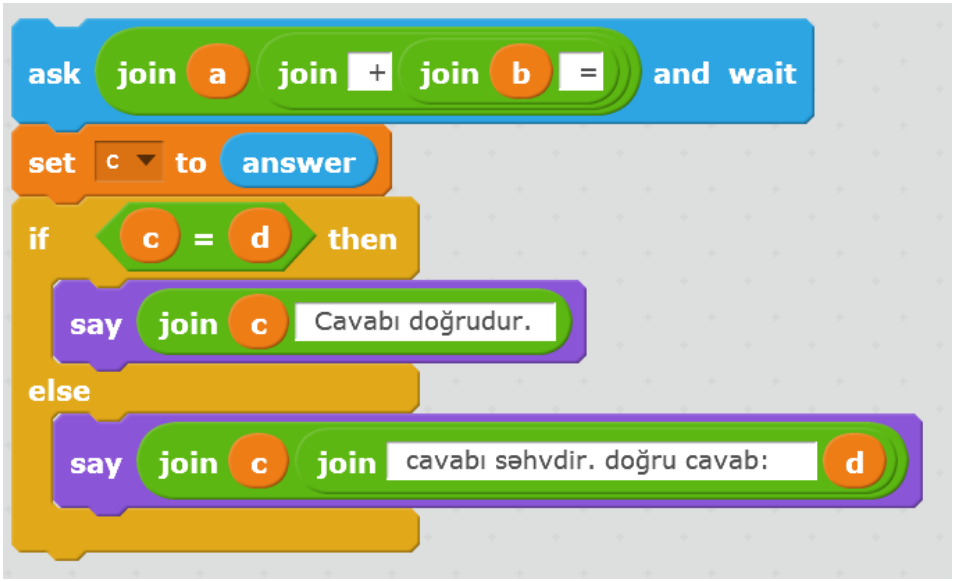
Burada istifadə edilən 

əmri onu bildirir ki, 1 ədədi ilə 30 ədədi arasında ixtiyari ədəd götürülür. 1 və 30 ədədlərini dəyişməklə seçimin imkanlarını dəyişə bilərik. Hətta 3 və ya 4 rəqəmli ədədlər də götürmək olar. Bu seçim Sizin ixtiyarınızdadır.

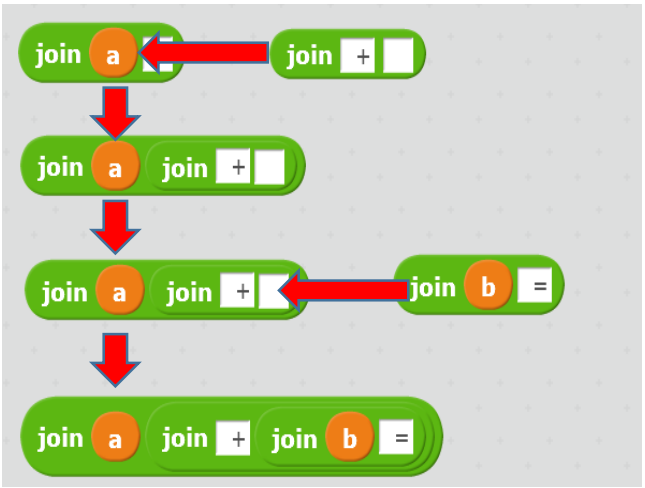
a və **b** dəyişənlərinə ixtiyari qiymət mənimsədilən kimi proqram **a** və **b** dəyişənlərinin cəmini hesablayıb **d** dəyişəninə mənimsədir. Bunun üçün **Operators**

qrupundan  operatoru götürülüb **set ___ to ____** əmrinə yerləşdirilir və **Data** qrupundan **a** və **b** dəyişənləri götürülüb boş xanalara qoyulur. Əmrin son nəticəsi belə olacaq:

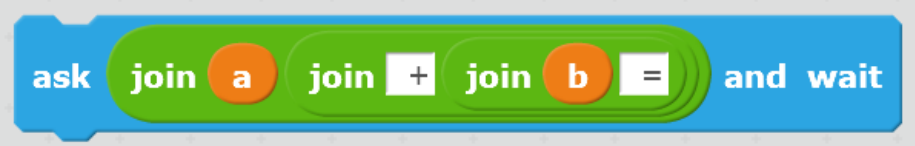
Proqram toplama əməlinin dəqiq cavabını hesabladıqdan sonra artıq istifadəçidən onun cavabını soruşur. Bu aşağıdakı əmrlərlə edilir:



Bu proqram blokunda birinci olan **ask** əmri ekranda hesablanası riyazi əməli göstərmək üçün **Operators** qrupundakı **join** operatorundan istifadə edir. Diqqət edirsinizsə, bu əmrdə **join** operatorundan 3 dəfə istifadə edilib. Səbəbi ondadır ki, adi **join** əmri 2 operantı birləşdirmək üçündür. Amma bizim halda operantların sayı dörddür. Bunlar **a** və **b** dəyişənləri, **"+"** və **"="** simvollarıdır. Buna görə 3 ədəd **join** operantı sağdakı şəkildə verilən qaydada bir-birinə birləşdirilir:



Birləşmənin son nəticəsi belə olacaq:



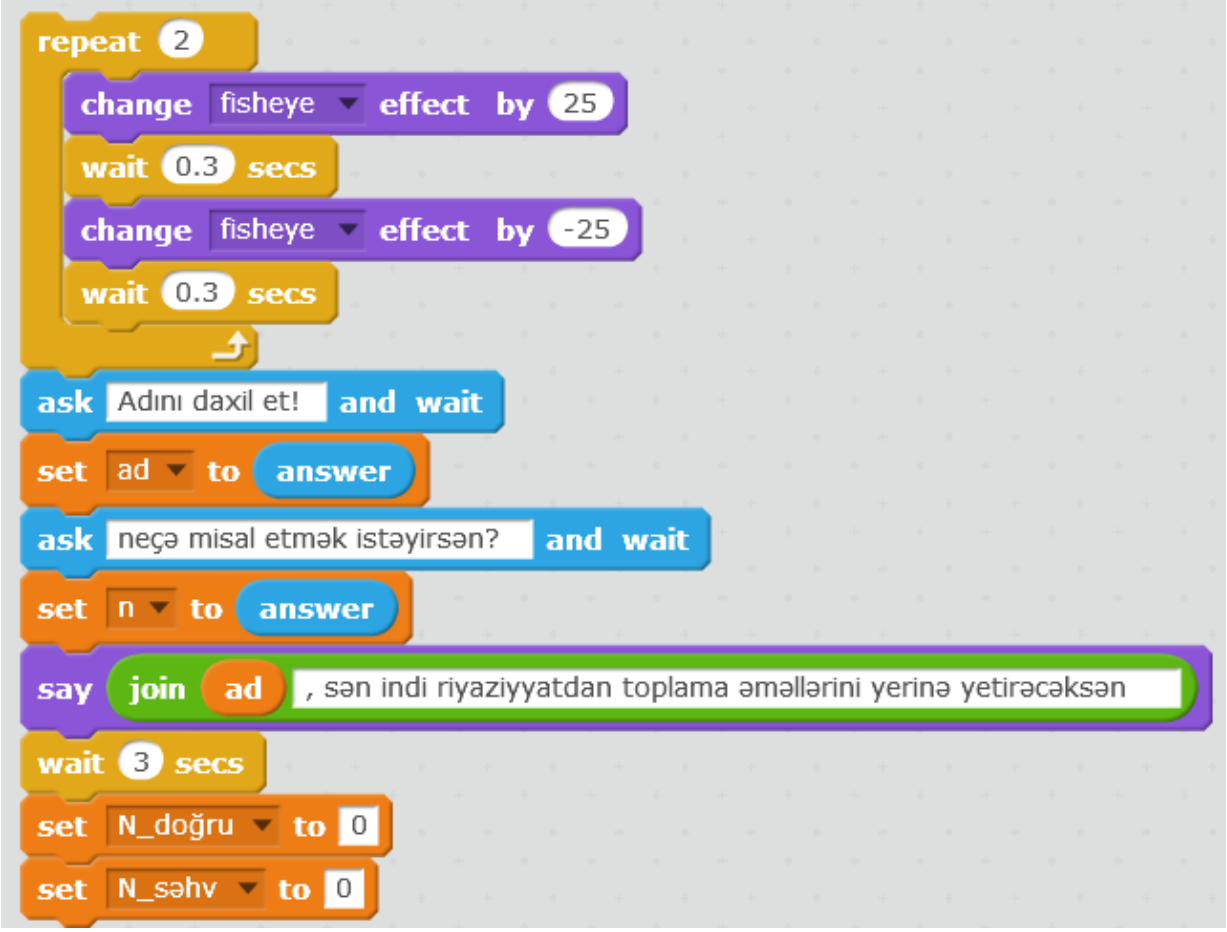
İstifadəçi misalın cavabını daxil etdikdən sonra proqram bu cavabı aşağıdakı əmrlə **c** dəyişəninə mənimsədir:

Artıq daxil edilən cavabın doğruluğunu yoxlaya bilərik. Bunun üçün **Control** qrupundan **if ___ else ___** əmrindən, **Data** qrupundan **a** və **b** dəyişənlərindən, **Looks** qrupundan **say** əmrindən və **Operators** qrupundan  operatorundan istifadə edəcəyik. Proqram bloku belə olacaq:

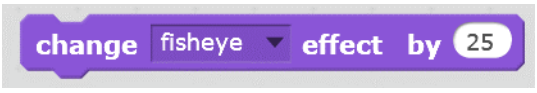
Burada yuxarıda verdiyimiz alqoritmə əsasən alınan cavab proqramın hesabladığı cavabla tutşdurulur (c=d). Cavab doğru olanda bu haqda məlumat verilir. Cavab səhv olanda isə səhv haqqında məlumat verməkdən başqa düzgün cavabı da ekranda əks etdirir.

Bizim proqram bir çalışma üçün işlədi. Amma bizim məqsədimiz daha genişdir. Biz istifadəçinin istədiyi sayda çalışmanı ona təqdim etməliyik və sonda ona yekun nəticə verməliyik.

Bunun üçün proqrama əlavə bloklar salmaq lazımdır. Birinci blok **Salam dostum** dan sonra proqrama əlavə edilən aşağıdakı blok olacaq:



Burada dövr əmri olan **repeat** 2 dəfə təkrar olunur və **Looks** qrupundakı **change** əmri **fisheye** rejimində arada 0.3 saniyə fasilə ilə müsbət və mənfi istiqamətdə yerinə yetirilir. Bu salamlaşma zamanı personaja hərəkət verir.

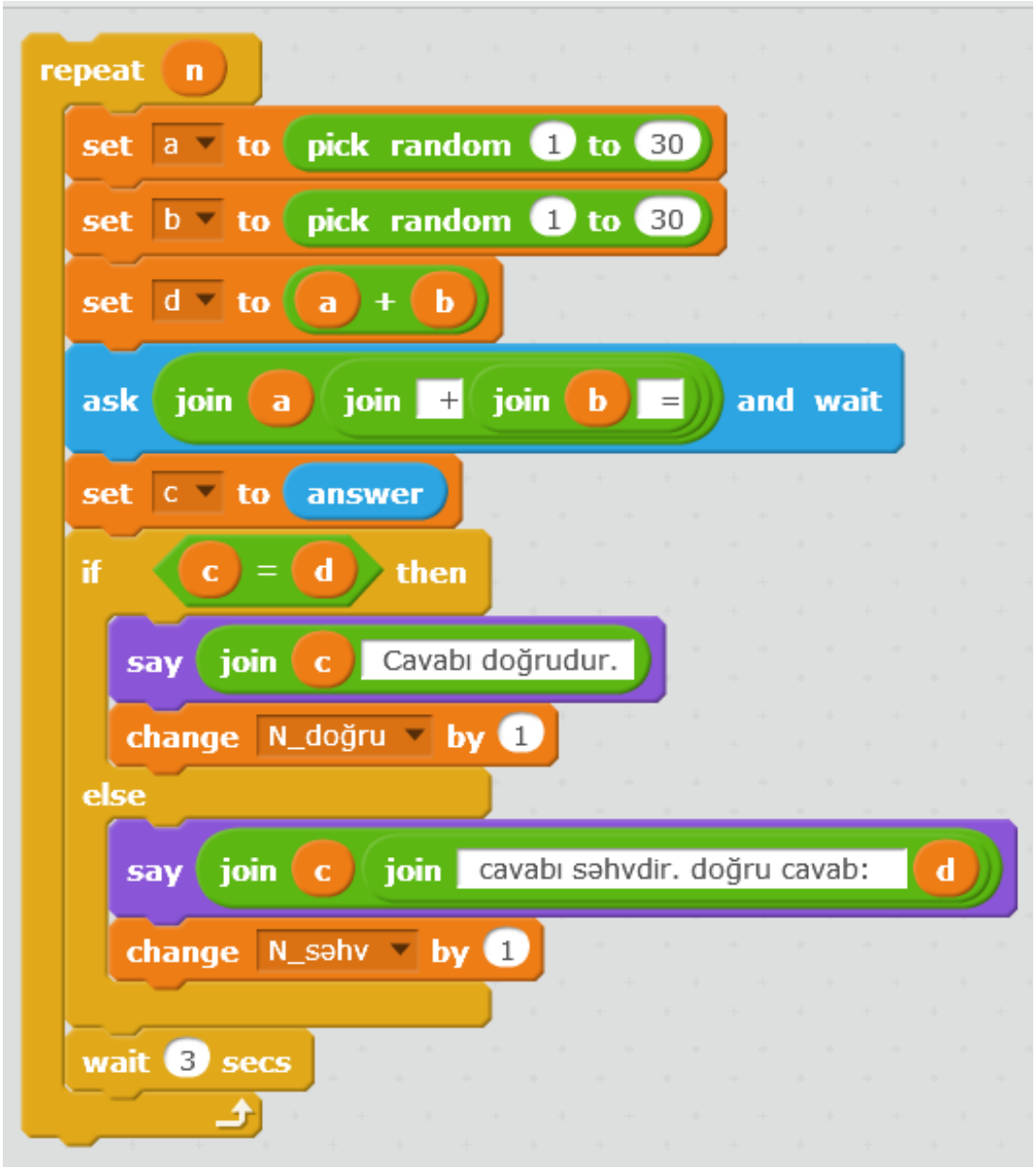


Daha sonra ask əmri ilə istifadəçinin adı daxil edilir və **ad** dəyişəninə mənimsədilir.

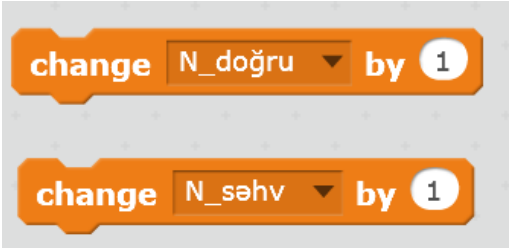
Növbəti **ask** əmri ilə istifadəçidən neçə misal etməsi soruşulur və bu məlumat da **n** dəyişəninə mənimsədilir.

Bundan sonra ekranda istifadəçinin adı və "**sən indi riyaziyyatdan toplama əməllərini yerinə yetirəcəksən**" məzmunlu mətn çıxır. Bu mətn ekranda

görünəndən 3 saniyə sonra **N_doğru** və **N_səhv** adlı dəyişənlərə sıfır qiyməti mənimsədilir. Bu o məqsədlə edilir ki, doğru və səhv cavablarının sayı toplanası dəyişənlər başlanğıcda təmizlənmiş olsun. Artıq bundan sonra bizim proqram daxil edilən say (**n**) qədər çalışmanı istifadəçiyə verməlidir. Bunun üçün biz **n** dəfə təkrarlanan dövr qurmalıyıq və dövrün içinə bir sual üçün qurduğumuz proqram blokunu salmalıyıq. Proqrama əlavə edilən hissə belə olacaq:



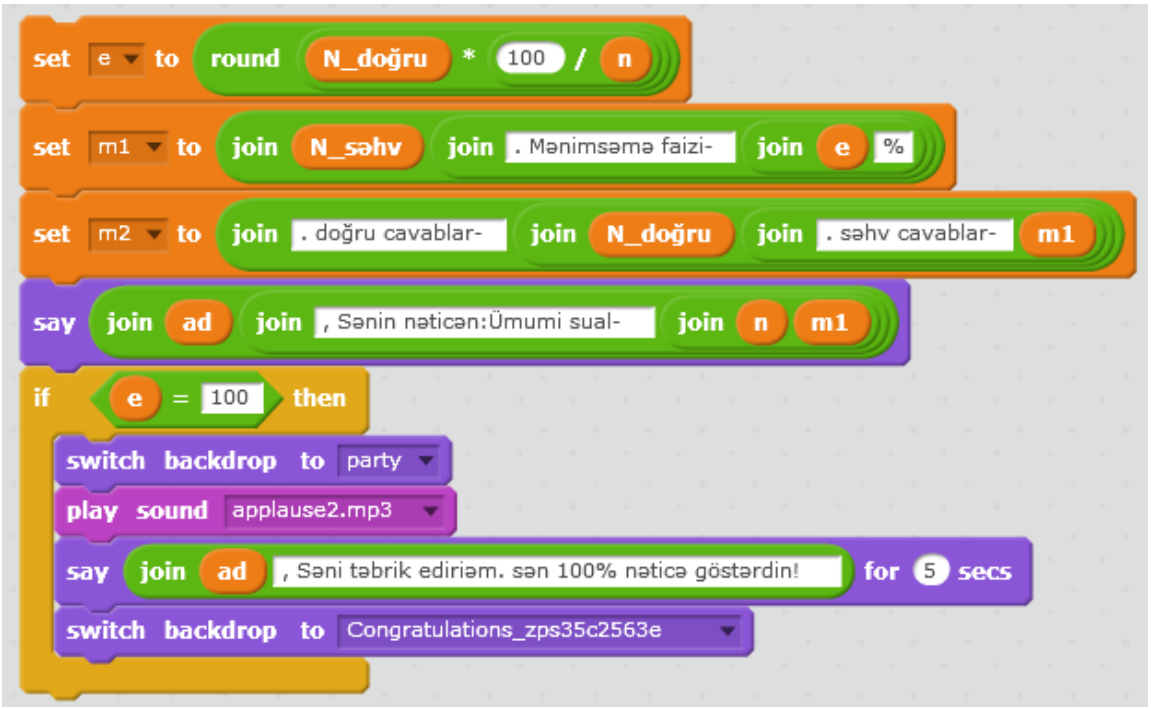
Bu hissəni biz artıq araşdırmışdıq. Sadəcə olaraq doğru və səhv cavabların sayını tapmaq üçün proqrama 2 əmr əlavə edilib.



Beləliklə, proqram əsas işini yerinə yetirdikdən sonra istifadəçinin cavabları əsasında yekun məlumatı çap etməlidir. Aşağıdakı əmr **N_doğru*100/n** düsturu ilə mənimsəmə faizini tapıb **e** dəyişəninə mənimsədir. Burada **Operators** qrupundakı **round** operatorundan istifadə edilmişdir. Bu operator bölmənin nəticəsini tama qədər yuvarlaqlaşdırır. Məsələn, üç sualdan birinə düzgün cavab verən istifadəçinin mənimsəmə faizi 1/3=0.33333333% edir. **round** operatoru belə halda nəticəni 1/3=33% verəcək.

Ümumi, doğru, səhv çalışmaların sayı və mənimsəmə faizi ekranda çıxır.

Əgər mənimsəmə faizi 100% olarsa, proqram istifadəçini təbrik edəcək. Bunun üçün proqrama əlavə iki fon (**party** və **Congratulations_zps35c2563e** adlı fonlar) və musiqi (**applause2.mp3**) daxil edilir. 100% nəticə olanda **party** fonu görünür və musiqi ifa edilir. Sonra 5 saniyə ərzində 100% nəticə göstərdiyinə görə təbrik edilir. Bundan sonra səhnədə **Congratulations_zps35c2563e** adlı fon görünür.

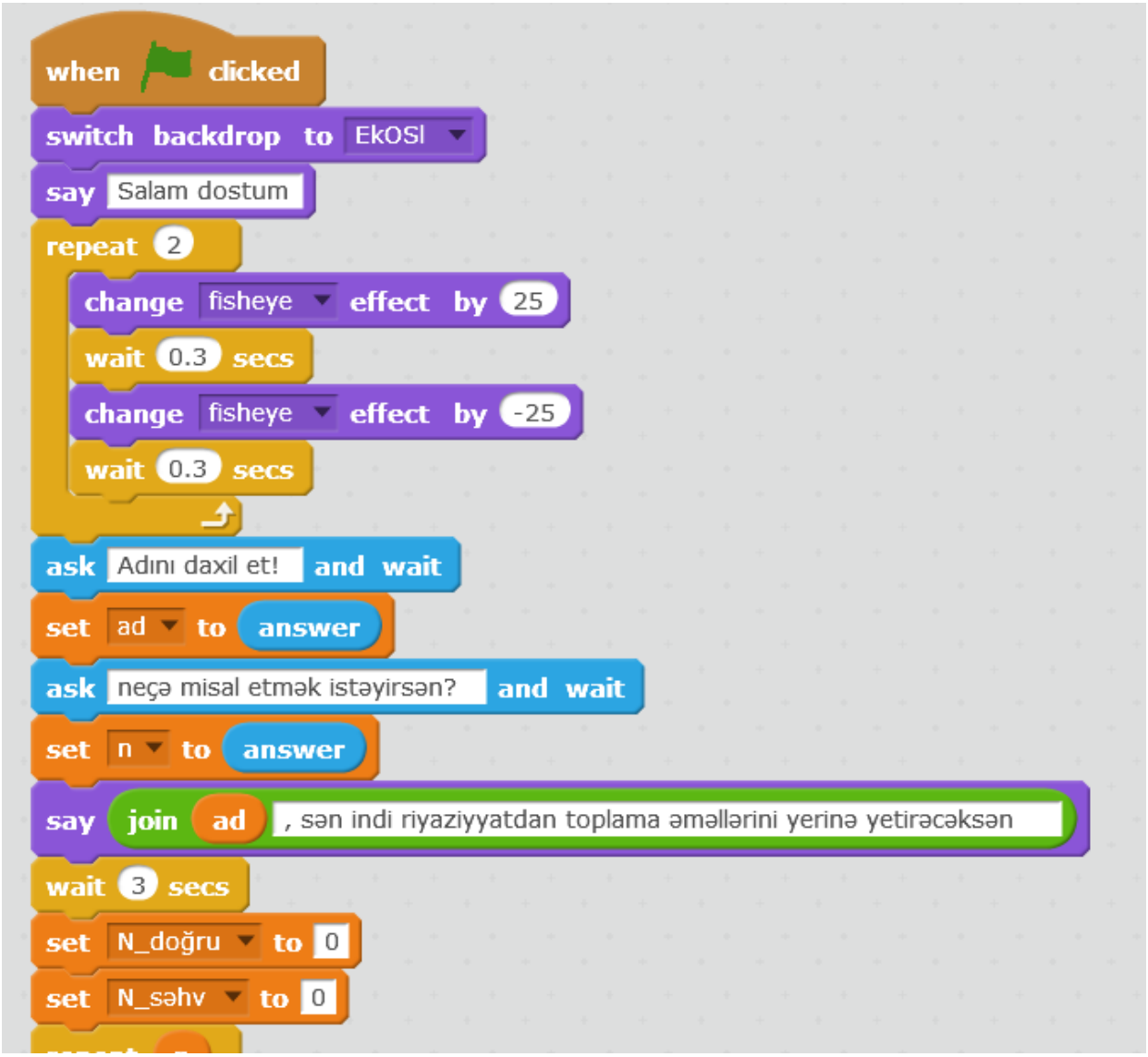


Bir şeyə diqqət edin. Bu blokda **join** operatorundan 9 dəfə istifadə edilir. Belə halda əmrin yazılışı böyüyür və oxumaq çətin olur. Belə hallarda əlavə 2 **m1** və **m2**

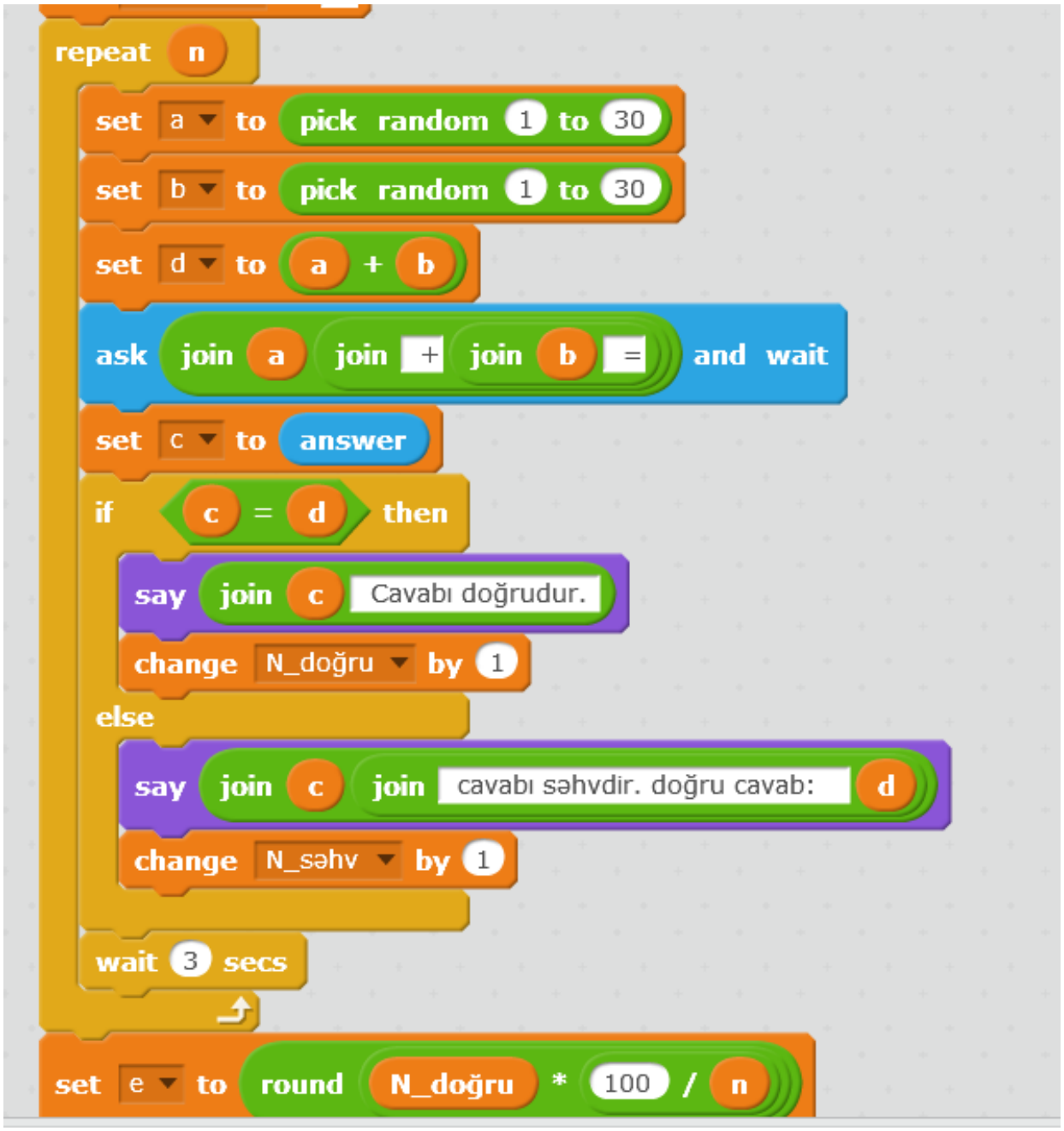
dəyişənləri yaradıb **join** operatorunun aralıq nəticələrini onlara mənimsətməklə yazılışı oxunaqlı etmək olar.

Proqramın tam mətni bir neçə hissəyə bölünməklə aşağıda verilib.

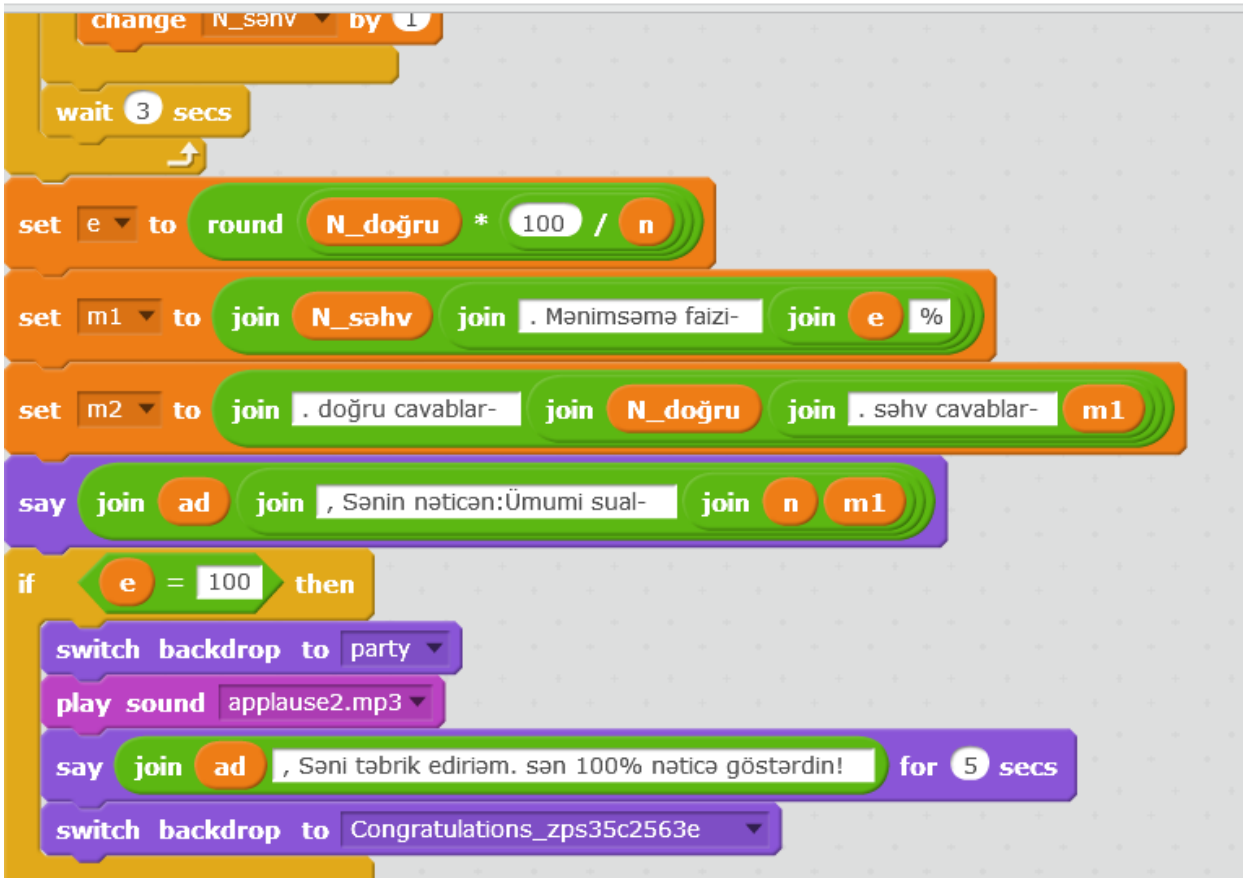
Başlanğıc



II hissə


repeat n
set a to pick random 1 to 30
set b to pick random 1 to 30
set d to a + b
ask join a join + join b = and wait
set c to answer
if c = d then
say join c Cavabı doğrudur.
change N_doğru by 1
else
say join c join cavabı səhvdir. doğru cavab: d
change N_səhv by 1
wait 3 secs
set e to round N_doğru * 100 / n

III hissə


change N_səhv by 1
wait 3 secs
set e to round N_doğru * 100 / n
set m1 to join N_səhv join . Mənimsəmə faizi- join e %
set m2 to join . doğru cavablar- join N_doğru join . səhv cavablar- m1
say join ad join , Sənin nəticən:Ümumi sual- join n m1
if e = 100 then
switch backdrop to party
play sound applause2.mp3
say join ad , Səni təbrik ediriəm. sən 100% nəticə göstərdin! for 5 secs
switch backdrop to Congratulations_zps35c2563e

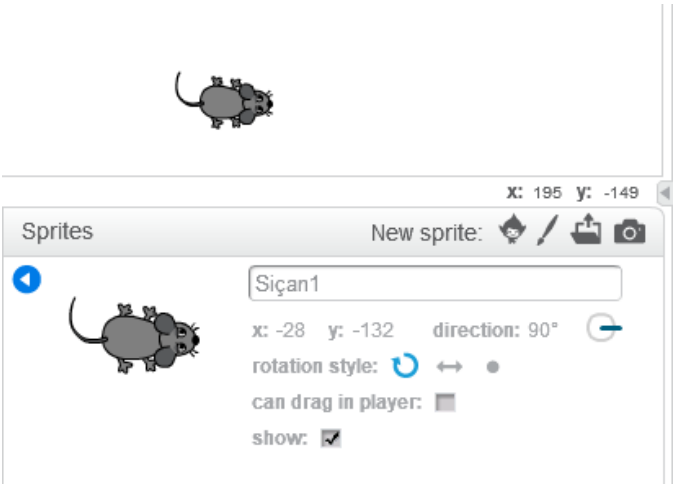
4. Siçanlar

Növbəti proqram **Sensing** qrupuna daxil olan **mouse down?** əmri ilə işləməyə həsr ediləcək.

4.1. Siçanların mouse qurğusu ilə idarəsi

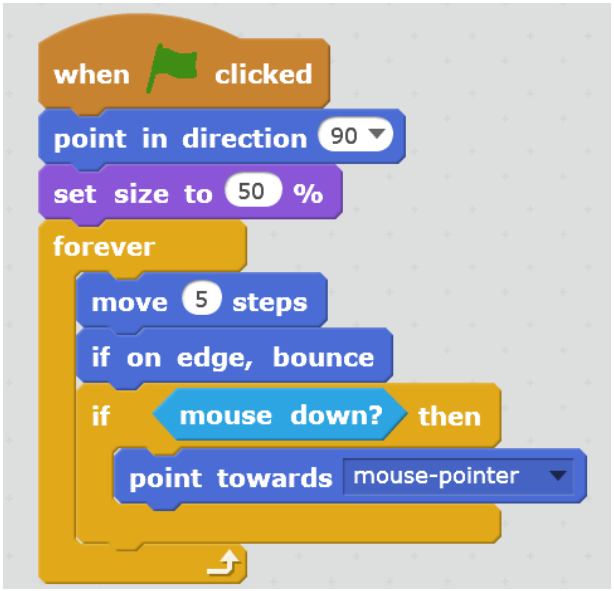
Proqramın ssenarisi belədir: Səhnədə bir neçə siçan qaçışır. Biz kursoru mouse qurğusu vasitəsilə səhnənin istənilən nöqtəsinə gətirib mouse qurğusunun sol düyməsinə sıxdıqda bütün siçanlar hərəkət istiqamətlərini dəyişib həmin nöqtəyə yönəlirlər.

Gəlin əvvəlcə personajı (standart **pişik** spraytını) dəyişək. Bunun üçün spraytlar kitabxanasından **Animals** qrupundan **Mouse1** adlı spraytı səhnəyə əlavə edək. **Pişik** spraytını silək. **Mouse1** spraytının adını dəyişib **Siçan1** qoyaq.



Şəkildə gördüyünüz soldakı mavi rəngli fondakı düyməsini sıxıb dəyişikliyi təsdiq edək. Nəticədə səhnədə adı **Siçan1** olan bir sprayt görünəcək.

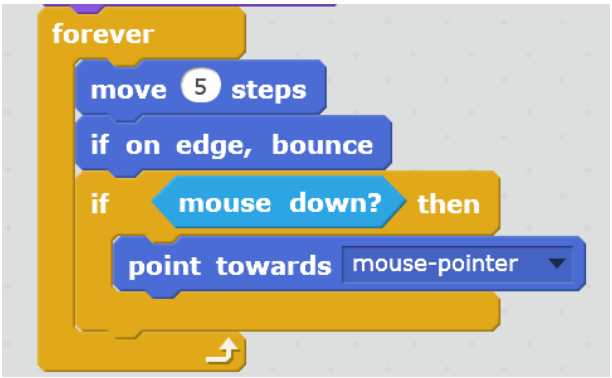
İndi isə **Siçan1** spraytı üçün sağdakı proqram kodunu yazaq.



Yaşıl bayraqla başlat əmri proqramı başlatmaq üçündür. Növbəti **point in direction-90** əmri **Siçan1** spraytının hərəkət istiqamətini sağa yönəldir.

Burada istifadə edilən **set size to 50%** əmri spraytın ölçüsünün hansı vəziyyətdə olmasından asılı olmayaraq onun ölçüsünü orijinal ölçüsünün 50%-i vəziyyətinə gətirir.

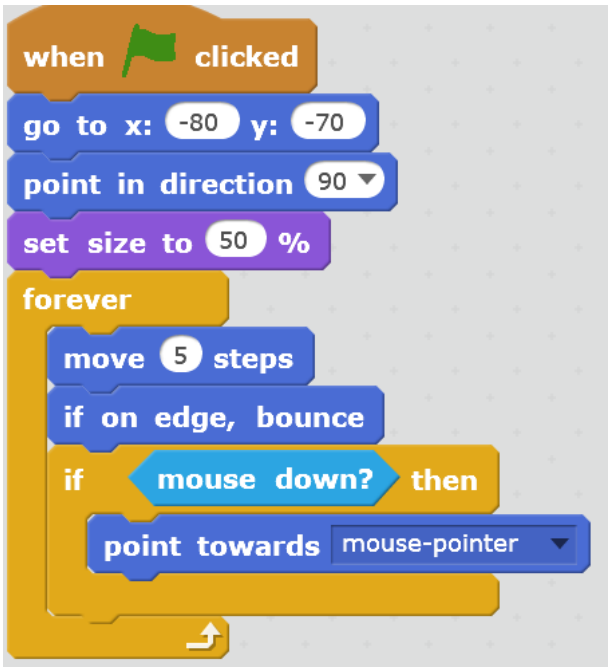
Proqram kodunun son hissəsini **forever** əmrinin daxilindəki əmrlər qrupu təşkil edir. **move 5 steps** sprayt 5 addım irəli gedir. Daha sonra **if on edge, bounce** əmri ilə sprayt səhnənin kənarına toxunduqda geriyə qayıdır.



Dövr daxilindəki son iki əmr bizim mouse qurğusunun sol düyməsinə sıxdıqda bütün siçanın hərəkət istiqamətini dəyişməyə xidmət edir.

Burada olan **if mouse down? then (əgər siçanın düyməsi sıxılmışsa onda)** əmrinin şərti ödənərsə, onda **point towards mouse pointer (smouse-un göstərici səmtinə dönməli)** əmri spraytın istiqamətini mouse qurğusunun sol düyməsi sıxılanda kursorun olduğu istiqamətə yönəldir.

Gəlin proqramımıza daha bir element əlavə edək. **go to x: 80 y: -70** əmri ilə siçanımıza səhnədə başlanğıc nöqtəsi müəyyən edək. Onda proqramımızın son variantı belə olar:

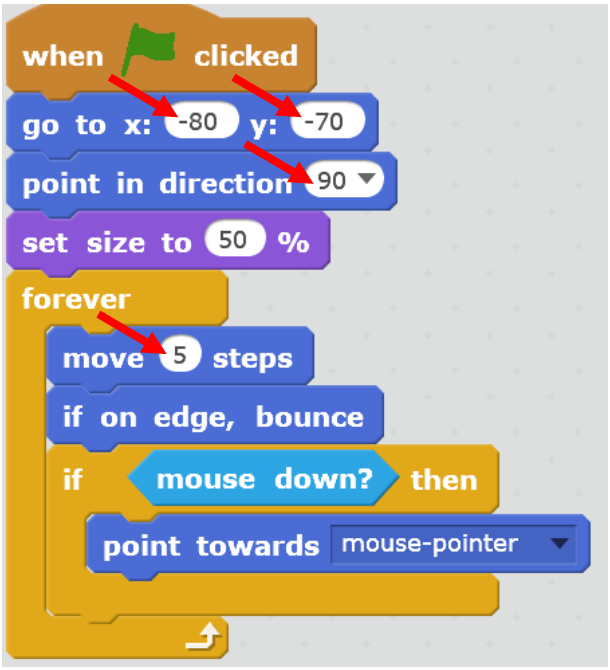


İndi isə səhnəni tam ekrana böyüdək və **yaşıl bayrağı** sıxıb proqramı başladaq. Siçan səhnə boyu hərəkət etməyə başlayır. İndi mouse qurğusunu səhnə boyu hərəkət etdirib müəyyən nöqtələrdə sol düyməni sıxın. Görəcəksiniz ki, sol düymə sıxılanda siçan (sprayt) istiqamətini kursor olan nöqtəyə yönəldir. Deməli, istədiyimiz nəticəni aldıq.

Amma bizim son məqsədimiz bir yox bir neçə siçanı idarə etməkdir. Gəlin siçanlarımızın sayını dördə çatdıraq. Bunun üçün səhnədə və ya spraytlar bölməsində **Siçan1** adlı spraytı seçib onun surətini (dublikatını) çıxardırıq. Bunun üçün ya spraytın üzərində sağ düyməni sıxıb **dublicate** sətrini seçirik, ya da Scratch

proqramının interfeysində 2 nömrəli sahədən 1 nömrəli

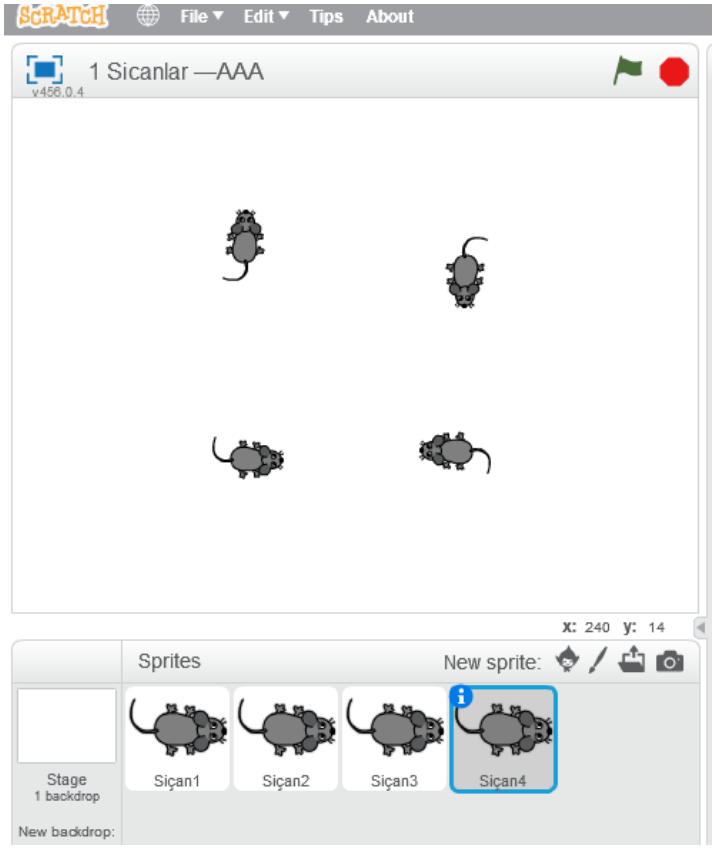
(Dublicate) aləti seçib kursoru surətini çıxarmaq istədiyimiz spraytın üzərinə gətirməli və onun üzərində sıxmalıyıq. Əməliyyatdan sonra həmin spraytın skriptləri ilə birlikdə dublikatı alınacaq.



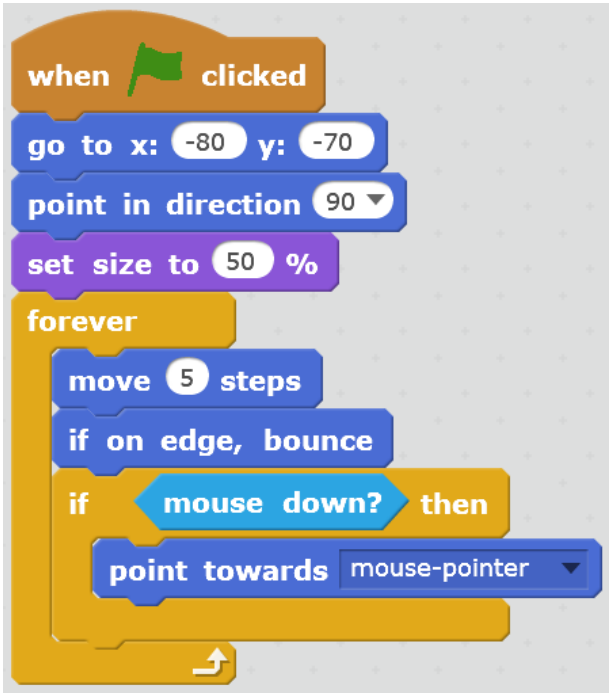
Yeni alınan spraytların adlarını uyğun olaraq **Siçan2**, **Siçan3**, **Siçan4** qoyuruq (adətən Scratch 2.0 proqramı nömrəli ad olanda addəyişməni avtomatik edir).

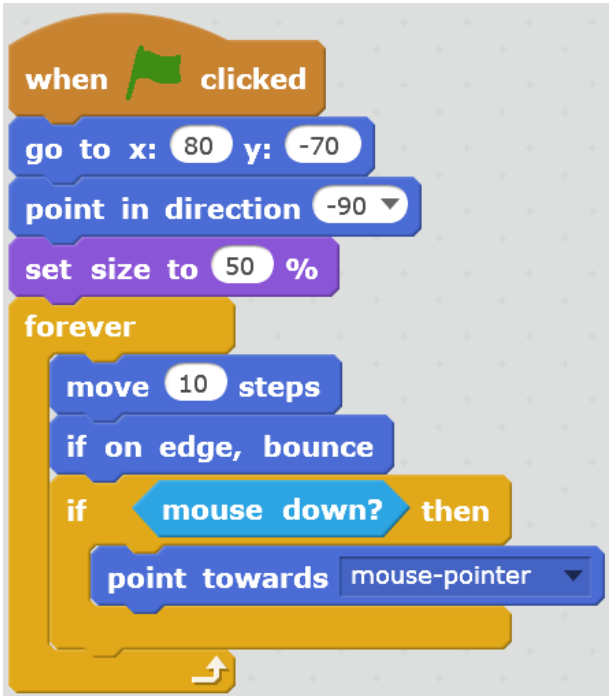
go to, **point in direction** və **move** əmrlərinin parametrlərini istədiyimiz formada dəyişirik.

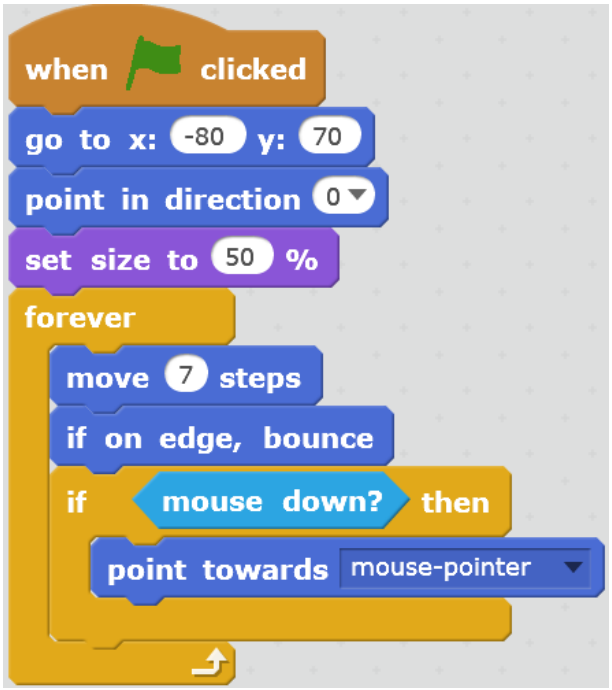
Aşağıda 4 siçan üçün olan variant verilibdir:

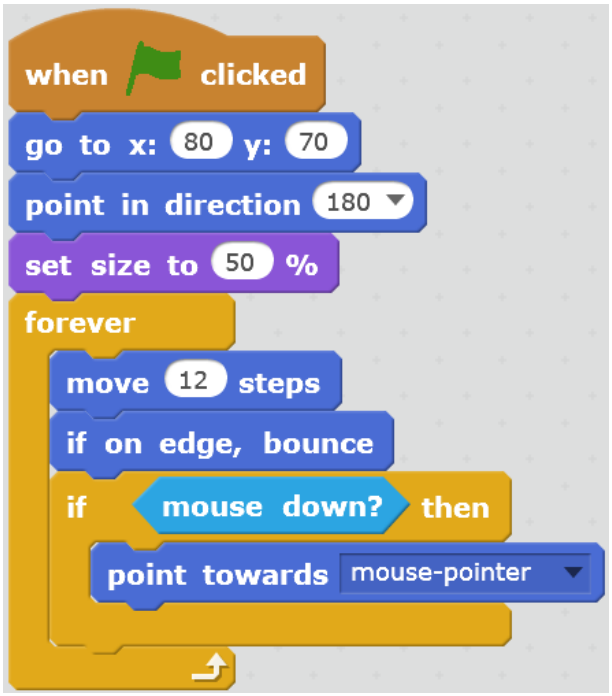


Dörd siçan (sprayt) üçün fərqli variant üçün əmrlər aşağıdakı kimi ola bilər:









Yenə də səhnəni tam ekrana böyüdək və **yaşıl bayrağı** sıxıb proqramı başladaq. Siçanlar səhnə boyu müxtəlif istiqamətlərdə hərəkət etməyə başlayır. İndi mouse qurğusunu səhnə boyu hərəkət etdirib müəyyən nöqtələrdə sol düyməni sıxın. Görəcəksiniz ki, sol düymə sıxılanda dörd siçanın hamısının istiqaməti kursor olan nöqtəyə yönəlir. Deməli, artıq istədiyimiz son nəticəni tam şəkildə aldıq.

5. Novruz payı

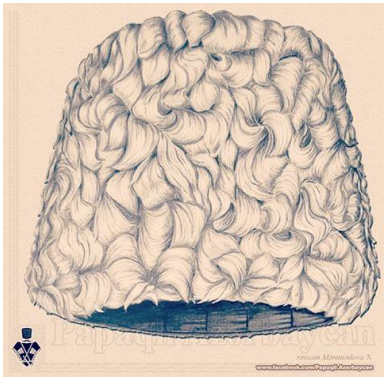
Xalqımızın bir çox gözəl adət və ənənələri, bayramları var. Belə bayramlardan biri də Novruz bayramıdır. Novruz qədim və böyük bayram kimi yazın gəlişi ilə əlaqədar olaraq qeyd edilir və sevincli hadisələrə-qışın bitməsi və yazın başlamasına həsr olunur. Təbiətin oyanması Novruz ilə başlayır. Novruz bayramı qədim adətlər və oyunlarla çox zəngindir. Qədim adətlərə misal olaraq "Xıdır İlyas"ı (məhsuldarlıq, çiçəklənmə simvolu), "Kos-kosa" əyləncəli meydan oyununu, papaqatma və fal açmanı göstərmək olar.

Gəlin biz də papaqatma adətinin əlamətləri əsasında bir oyun quraq. Oyunu **Novruz payı** adlandıraq. Oyunun ssenarisi belədir.

Səhnənin aşağısında papaq var. Yuxarıda solda həyatın (canların) sayı verilib. Yəni oyunçu həyatların sayı qədər əlavə oynaya bilər.

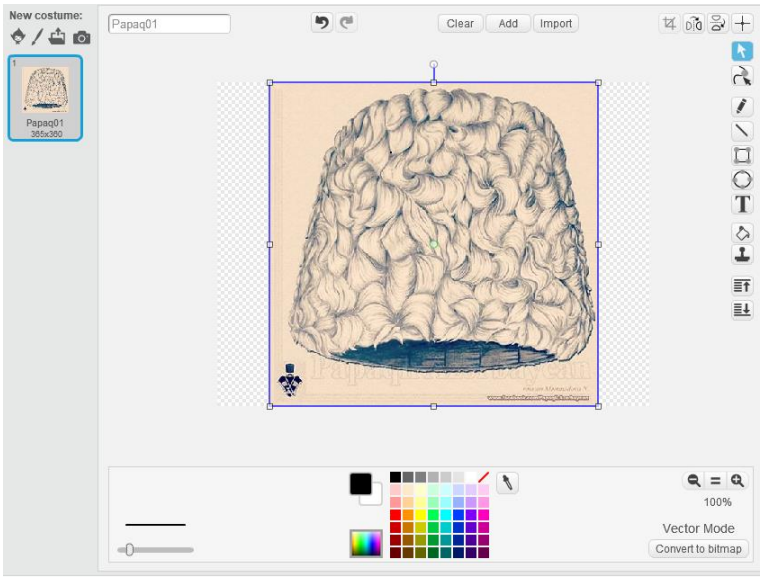
Yaşıl bayraqla başlat düyməsini sıxan kimi yuxarıdan üç obyekt düşməyə başlayır: şəkərbura, paxlava və tonqal. Oyunçu şəkərbura və paxlavanı tutanda onun hesabına əlavə xal yazılacaq. Tonqala toxunanda isə həyatlardan birini itirir. Şəkərbura oyunçuya əlavə 1 xal, paxlava isə 2 xal verir. 20 əlavə xal toplayan oyunçu yeni bir həyat əldə edir. Oyun ya həyatlar tam qurtaranda, ya da həyatların sayı 10-a çatanda bitir. 10 ədəd həyat əldə edən oyunçu qalib hesab edilir.

Google.az saytından bir ədəd milli papaq, şəkərbura, paxlava və tonqal şəkli tapın. Həmin şəkilləri kompüterinizə yükləyin. Bundan sonra yüklənmiş şəkillərin kənarlarını təmizləyin.

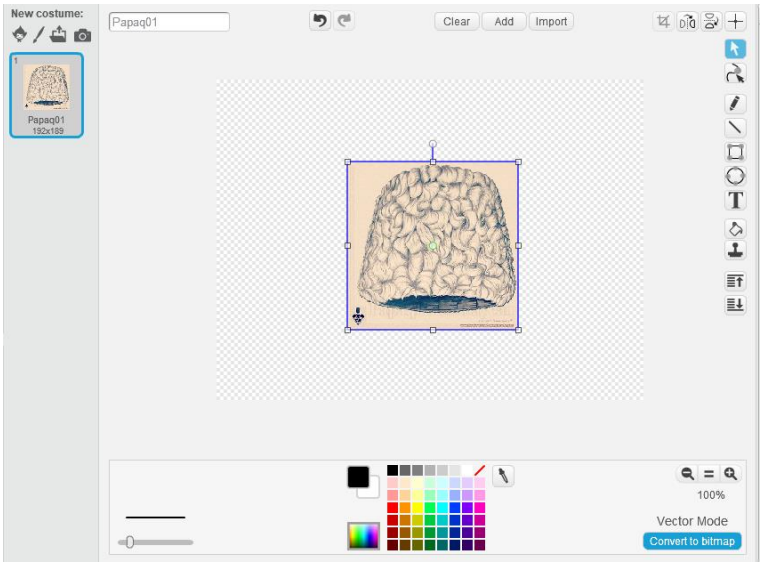


Gəlin internetdə yuxarıda gördüyümüz və ya ona oxşar bir papaq şəkli axtaraq. Biz gördüyünüz şəkli http://www.pictame.com/user/papaqli.azerbaycan/2281523914 ünvanından tapmışıq. Amma yenə də seçim Sizin ixtiyarınızdadır.

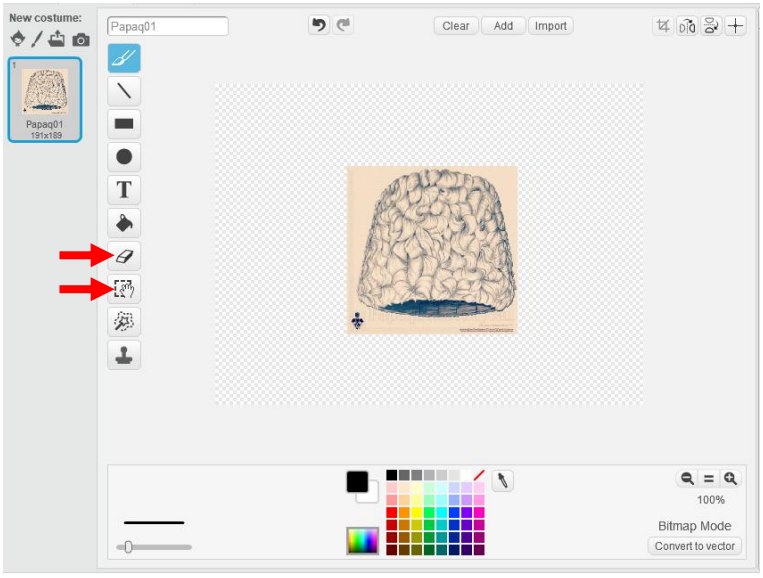
Yeni sprayt kimi həmin bu şəkli kənar fayl kimi proqrama yükləyək. Bundan sonra daxili qrafik redaktora daxil olaq.



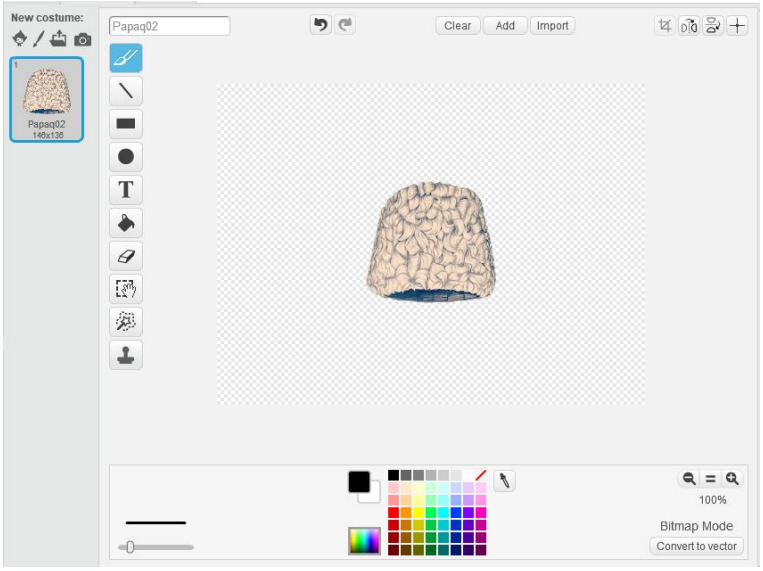
Qrafik redaktorda açılmış şəkli seçib onun ölçüsünü kiçildək.



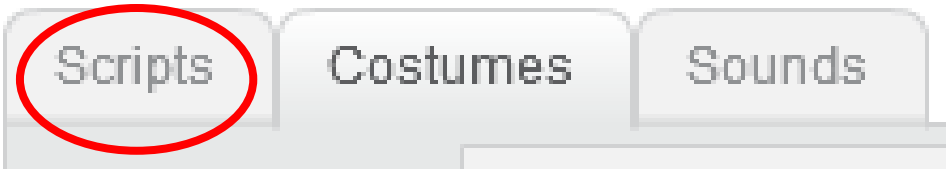
Əgər qrafik rejim rastr rejimində deyilsə, Qrafik redaktorun aşağı sağ küncündə **Convert to bitmap** düyməsini sıxıb qrafik redaktoru rastr rejiminə keçirin.



Qrafik redaktorun rastr rejimində solda açılan menyuda **Pozan** və **Düzbucaqlı seçim** alətlərinin köməyilə papağın kənarlarındakı artıq hissələri silirik.



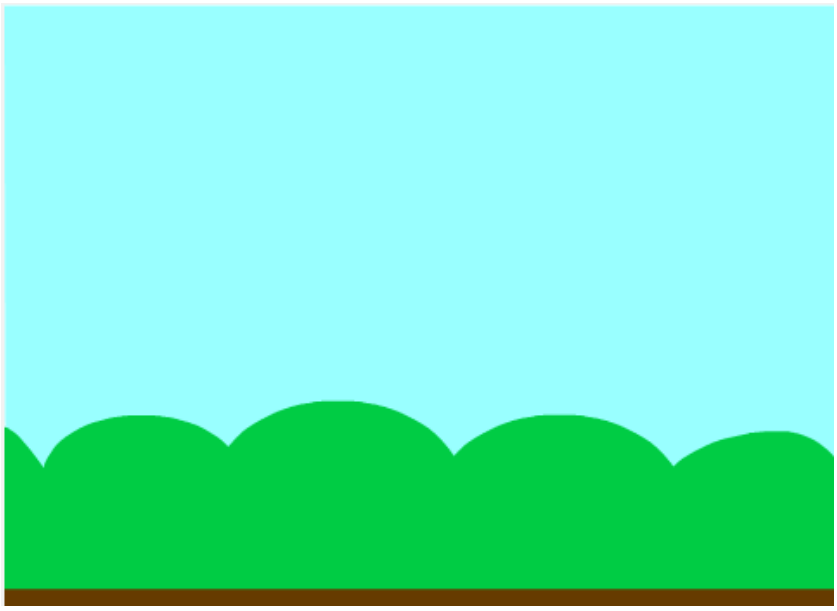
Artıq bizim bir obyektimiz hazırdır. Daha sonra interfeysin yuxarı hissəsindəki menyudan **Scripts** düyməsini sıxıb əsas menyuya qayıdırıq.



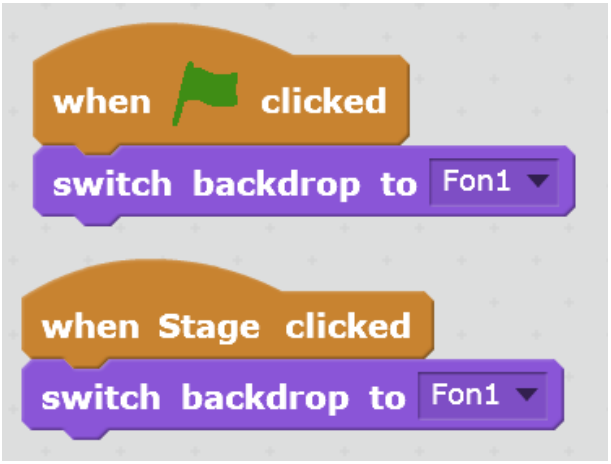
Oxşar qaydada şəkərbura, paxlava, səməni və tonqal şəkillərini proqrama yükləyib onların üzərində işləyirik. Bu personajların adını uyğun olaraq **şəkərbura**, **paxlava**, **səməni** və **tonqal** adlandırırıq.

Oyunumuz Novruzla bağlı olduğundan bizə uyğun fon lazımdır. Ya internetdən Novruzla bağlı fon tapa bilərik, ya da ozümüz uyğun fon yarada bilərik. Özümüz fon yaratmaq üçün səhnəni seçirik, daxili qrafik redaktora daxil oluruq və orada aşağıda gördüyümüz qradiyent fonu qururuq. Sonda fonun adını dəyişib **Fon1** qoyuruq.

İstəsəniz fonlar kitabxanasından **blue sky** adlı fonu da götürə bilərsiniz. Onda fonun adını yenə də dəyişib **Fon1** qoyuruq.



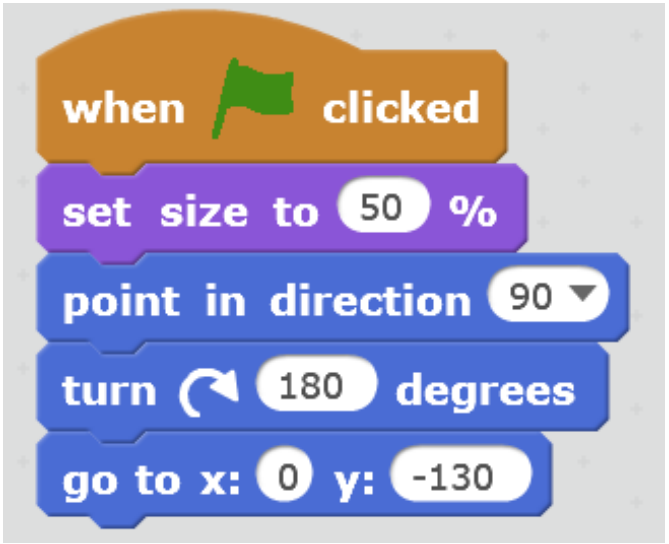
Proqram başlayan kimi ekranda **Fon1** adlı fon görünməlidir. Bunun üçün səhnəni seçirik və onun üçün aşağıdakı əmrləri əlavə edirik.



Birinci iki əmr **Yaşıl bayraq** düyməsini sıxanda fonu dəyişir. İkinci iki əmr isə səhnənin üzərinə sıxanda fonu dəyişir. Hər iki halda səhnədə **Fon1** adlı fon görünür.

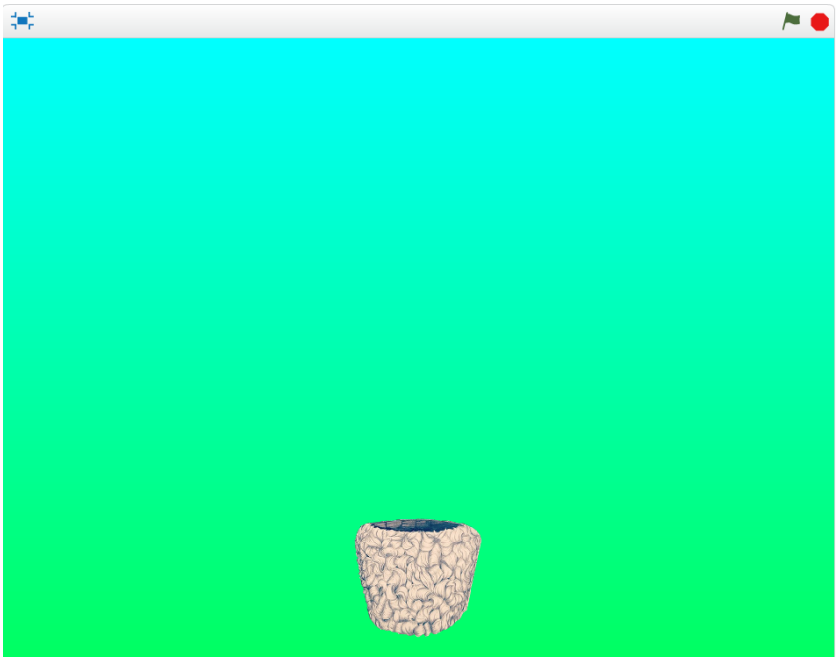
Növbəti addım papağın idarə edilməsidir. Birinci papağın ölçüsü səhnə üçün böyükdür, onu kiçiltmək lazımdır. İkincisi papağın aşağı tərəfi yuxarıya istiqamətlənməlidir. Çünki göydən düşən hədiyyəni yalnız belə vəziyyətdə tuta bilərik.

Bunun üçün **papaq** personajı üçün ilk olaraq aşağıdakı əmrləri verək.



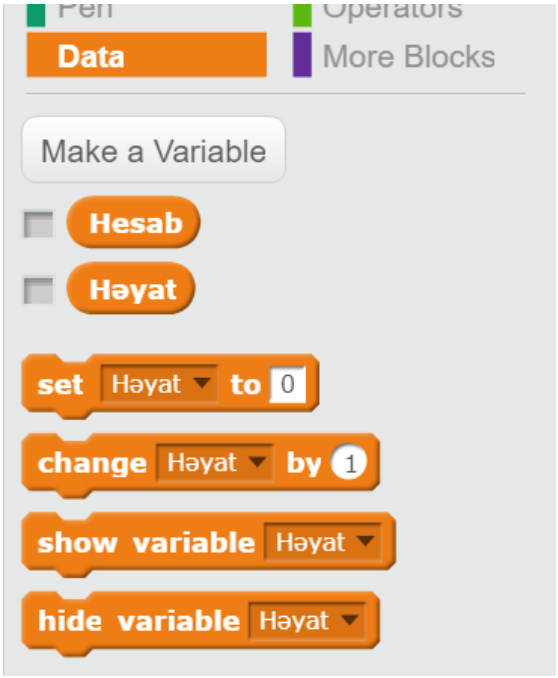
Bu əmrlər bloku belə işləyir:

- Yaşıl bayraq düyməsi sıxılan kimi papağın ölçüsü 50% ölçüyə gətirilir.
- İstiqamət sağa götürülür.
- Papaq 180 dərəcə çevrilir.
- Papaq səhnənin aşağısında mərkəzdə ((0;-130) nöqtəsində) yerləşdirilir.



Yuxarıda dediyimiz kimi, oyunda hər tutduğu hədiyyə üçün oyunçu xal toplayır. Bundan başqa oyunun əvvəlində oyunçunun 4 ədəd həyatı var. Oyunçu həyatların sayı qədər əlavə oynaya bilər.

Oyunçu şəkərbura və paxlavanı tutanda onun hesabına əlavə xal yazılacaq. Tonqala toxunanda isə həyatlardan birini itirir. Şəkərbura oyunçuya əlavə 1 xal, paxlava isə 2 xal verir. 20 əlavə xal toplayan oyunçu yeni bir həyat əldə edir. Oyun ya həyatlar tam qurtaranda, ya da həyatların sayı 10-a çatanda bitir. 10 ədəd həyat əldə edən oyunçu qalib hesab edilir. Deməli bu xalları və həyatları hesablamaq üçün bizə iki dəyişən yaratmaq lazımdır.



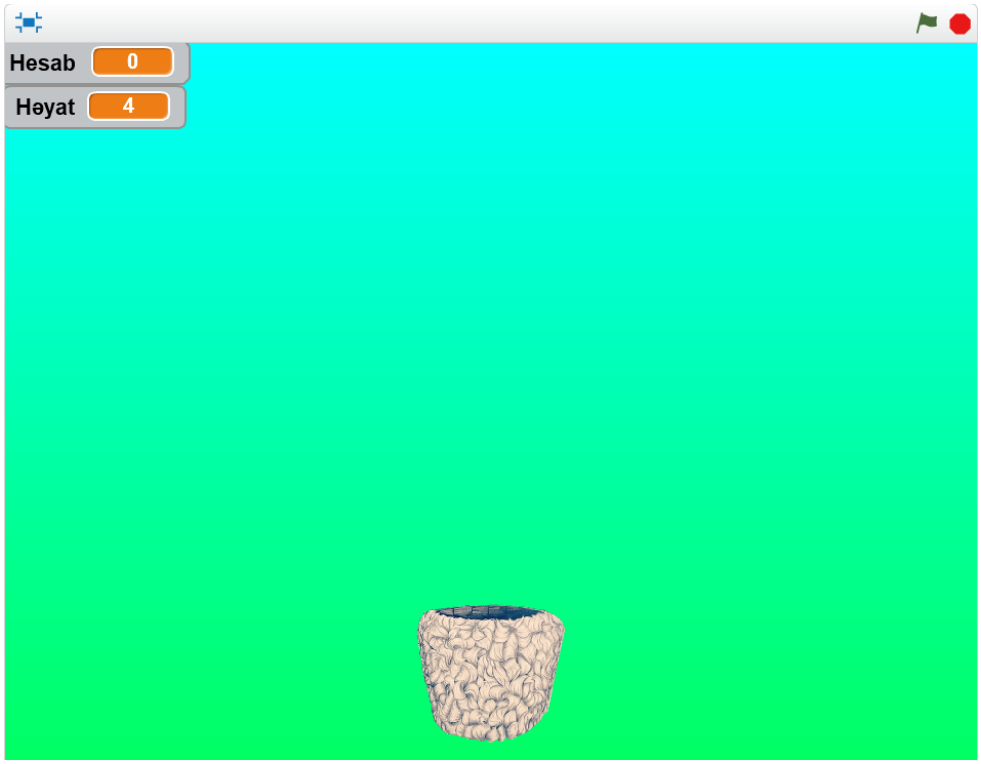
Gəlin bu dəyişənlərdən birini **Həyat**, digərini isə **Hesab** adlandıraq.

Papaq personajı üçün proqram koduna əlavə əmrlər daxil edək.

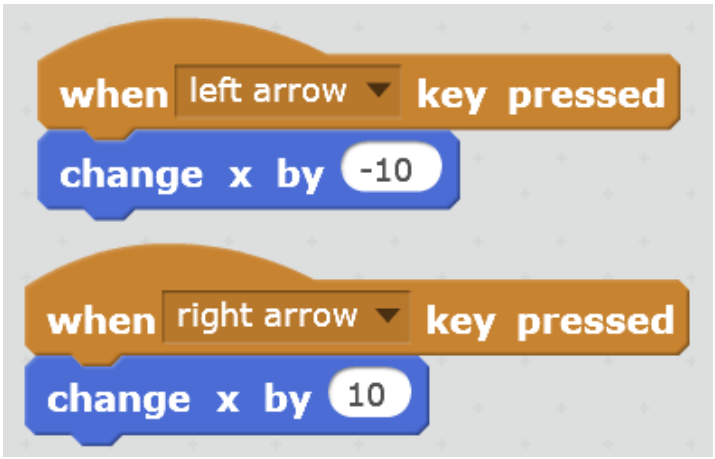
Bu əmrlərdən birincisi **Hesab** dəyişəninə "0" qiymət mənimsədir.

İkinci əmr **Həyat** dəyişəninə ilkin qiymət kimi "4" (4 həyat və ya can) mənimsədir.

Üçüncü və dördüncü əmr **Hesab** və **Həyat** dəyişənlərini səhnədə görünən edir.

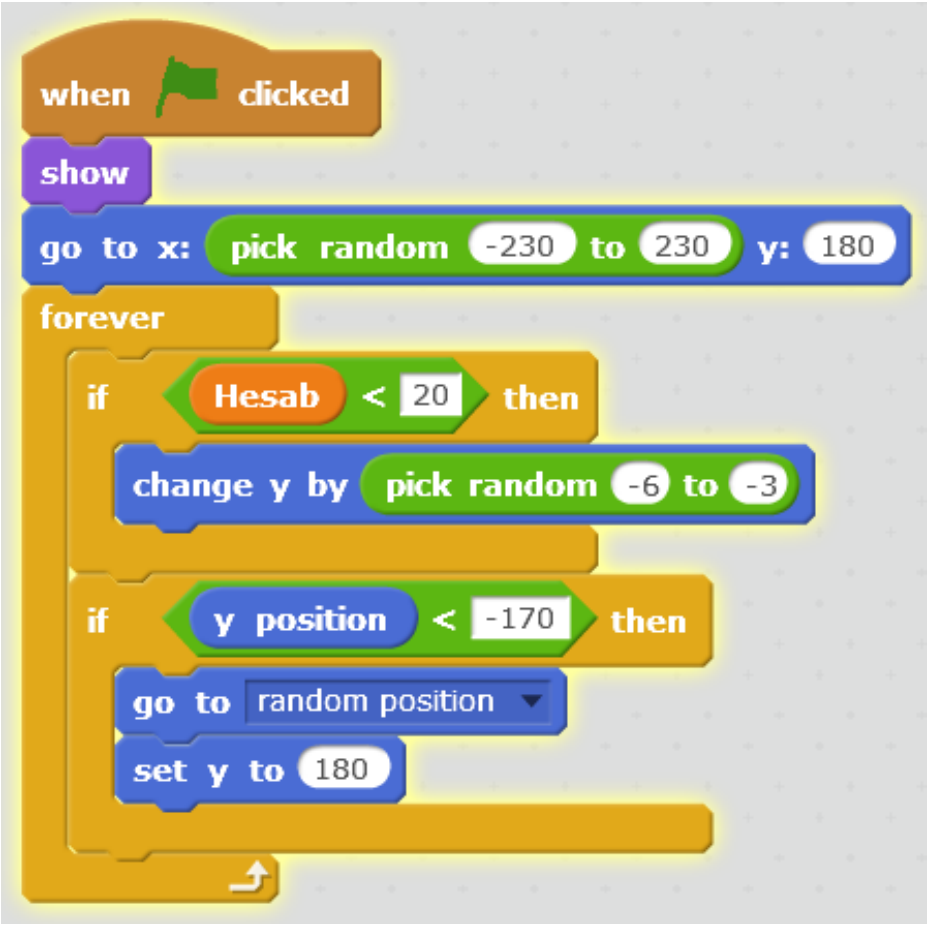


Papağı səhnədə idarə etmək üçün klaviaturada **sola** və **sağa** oxlarından istifadə edək. Aşağıda bu əmrləri görürsünüz.



Papaq personajı üçün proqram kodunu yazmağı müvəqqəti dayandırıb, şəkərbura, paxlava və tonqal üçün proqram yazaq. Bu 3 personajın əmrlərinin başlanğıc hissəsi

bir-birinə oxşardır. Gəlin əvvəlcə tonqal üçün proqram kodu yazaq. Aşağıda həmin proqram kodunu görürsünüz.



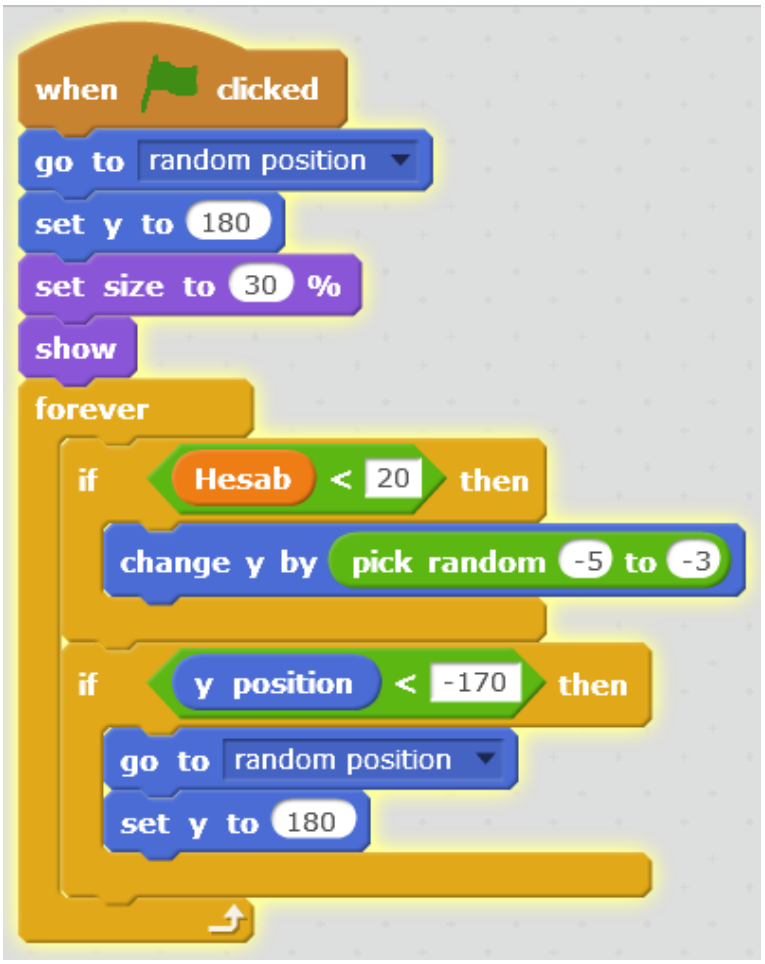
Yazdığımız əmrləri şərh edək.

Yaşıl bayrağı sıxan kimi tonqal səhnənin yuxarı sətrində ixtiyari yerdə peyda olur. Daha sonra **forever** əmrinin daxilində iki **if** əmri verilib. Birinci **if** əmrində **Hesab** dəyişəninin qiymətinin 20 xala çatması yoxlanılır. Əgər xal 20-yə çatmayıbsa, onda sadəcə olaraq tonqalın mövqeyi **y** oxu boyunca -6 ilə -3 arasında seçilmiş ixtiyari ədəd qədər azalır. Başqa sözlə tonqal səhnədə aşağı doğru hərəkət edir.

İkinci **if** əmrində tonqalın səhnənin aşağısına çatması yoxlanılır. Əgər tonqal aşağı çatmışdırsa, onda tonqal yenidən səhnənin yuxarı sətrində ixtiyari yerdə peyda olur.

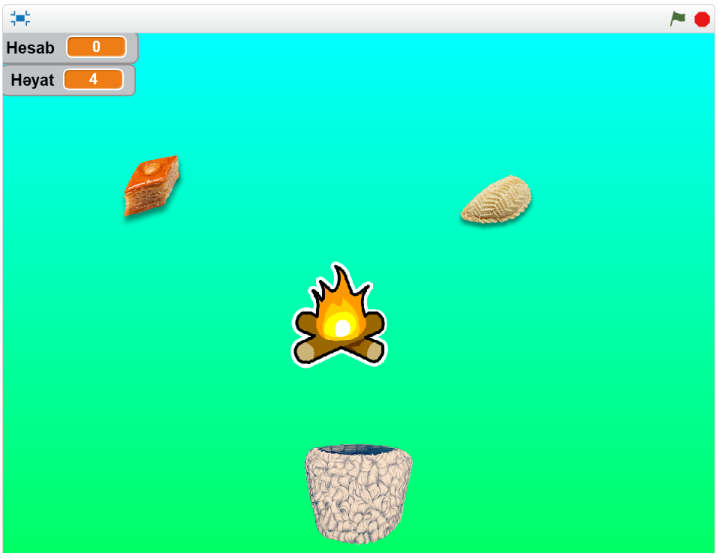
İndi isə şəkərbura üçün proqram kodu yazaq. Bu kod prinsipcə tonqal üçün olan koda bənzəyir. Sadəcə olaraq **random** əmrinin bir başqa variantı tətbiq edilib. Bir də kiçik olsun deyə şəkərburanın ölçüsü 30% götürülüb.

Aşağıda dediyimiz fərqi görürsünüz.



Eyni əmrləri paxlava personajı üçün də yazırıq.

Personajlar üçün proqram kodunu yazdıqdan sonra tam ekran rejiminə keçin və yaşıl bayraq düyməsini sıxıb proqramı başladın.



Ekranda şəkildə gördüyünüz görüntüyə oxşar vəziyyəti görəcəksiniz. Şəkərbura, paxlava və tonqal yuxarıdan müxtəlif yerlərdən düşürlər. Sağ və sol papağı hərəkət

etdirin. İstədiyimiz hərəkət var. Amma fərqli etsək oyunda dinamiklik daha da artar. Müxtəlif personajlar (şəkərbura, paxlava və tonqal) üçün düşmə sürətini dəyişmək məqsədilə **pick random__to__** əmrində qiymətləri aşağıdakı şəkildə verilən kimi dəyişin.

İndi isə proqramın davamını yazaq. Oyunun sonrakı ssenarisi belədir. Papağı hərəkət etdirərək yuxarıdan düşən şəkərbura və ya paxlavanı tutmaq lazımdır ki, xal ehtiyatını artırasan. Şəkərbura tutanda 1 xal, paxlava tutanda isə 2 xal qazanacaqsan. Xalın 20-yə çatırsa, onda həyatın (canın) sayı bir vahid artır və xal ehtiyatın yenidən hesab aparmaq üçün sıfırlanır. Əgər papaqla tonqal tutursansa, onda həyatlarının sayı bir vahid azalır. Həyatların sayı sıfıra çatanda oyunçu uduzur və oyun bitir. Yox əgər sən 10-cu həyat qazanırsansa, oyunçu qalib gəlir və bu dəfə də oyun bitir. Amma bu dəfə bitmə qələbə ruhu ilə olur.

Yuxarıda təsvir eydiyimiz ssenari aşağıdakı qaydada reallaşdırılacaq.

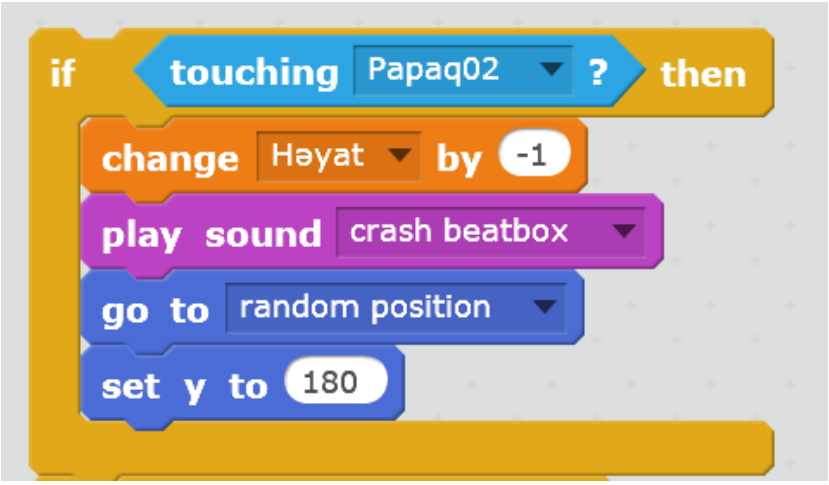
- Tonqal papağa toxunanda spesifik səs çıxır və **Həyat** dəyişəninin qiyməti 1 vahid azalır. **Həyat** dəyişəninin qiyməti sıfıra bərabər olarsa oyunun bitdiyi və oyunçunun uduzduğu bildirilir. Əgər hələ həyat varsa, onda oyun davam edir.
- Şəkərbura və ya paxlava papağa toxunanda isə **Hesab** dəyişəninin qiyməti şəkərbura üçün 1 vahid, paxlava üçün 2 vahid artırılır. **Hesab** dəyişəninin qiyməti 20 və ya daha çoxdursa, onda **Həyat** dəyişənin qiyməti 1 vahid artırılır. **Hesab** dəyişəninin qiyməti 20 xal azaldılır. Şəkərbura və ya paxlava yerini səhnənin üstünə dəyişdirir. Oyun davam edir. Burada da şəkərbura və ya paxlava tutanda fərqli səslər çıxarılır.

Adətən yeni sprayt yaradan zaman Scratch proqramı həmin spraytı **pop** adlı audio səslə təchiz edir. Yox əgər Sizdə Scratch bunu etmirsə, onda **şəkərbura**, **paxlava** və **tonqal** spraytlarına **Sounds** bölməsindən **Effects** qrupundan **pop** səsi əlavə edin. Bundan başqa **səməni** spraytına **Effects** qrupundan **zoop** səsini, **tonqal** spraytına **Human** qrupundan **crash beatbox** səsini, **Papaq02** spraytına isə **Music Loops** qrupundan **dance around** səsini əlavə edin.

“Əvvəlcə **tonqal** personajına əlavə ediləsi əmrlərlə tanış olaq. Bu əmrlər bloku aşağıdakı şəkildə verilib.

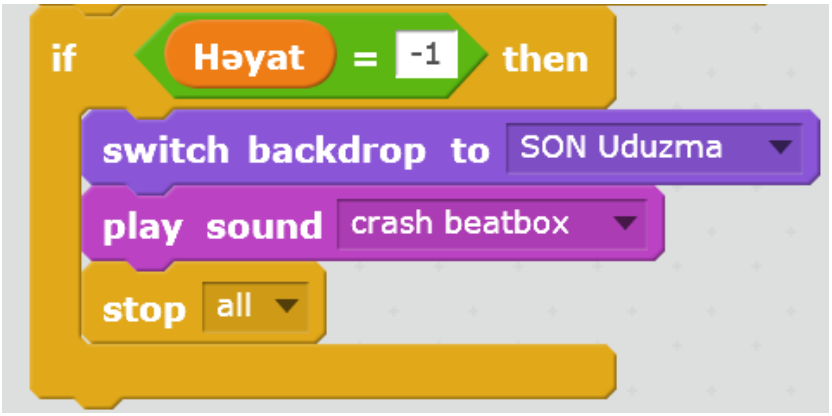
if_____then əmrində **Papaq02 obyektinə toxunma** (**touching Papaq02 ?**) şərti ödənəndə aşağıdakı əmrlər yerinə yetirilir:

- **Həyat** dəyişəninin qiyməti 1 vahid azaldılır.
- **crash beatbox** səsi çıxarılır.
- İxtiyarı mövqeyə gedir.
- **y** dəyişəninə 180 qiyməti mənimsədir. Ən yuxarı sətrin nömrəsi.



Növbəti **if_____then** əmrində **Həyat=-1** şərti ödənəndə aşağıdakı əmrlər yerinə yetirilir:

- **Son Uduzma** adlı fonu səhnədə göstərir.
- **crash beatbox** səsi çıxarılır.
- Bütün əməliyyatlar durdurulur.

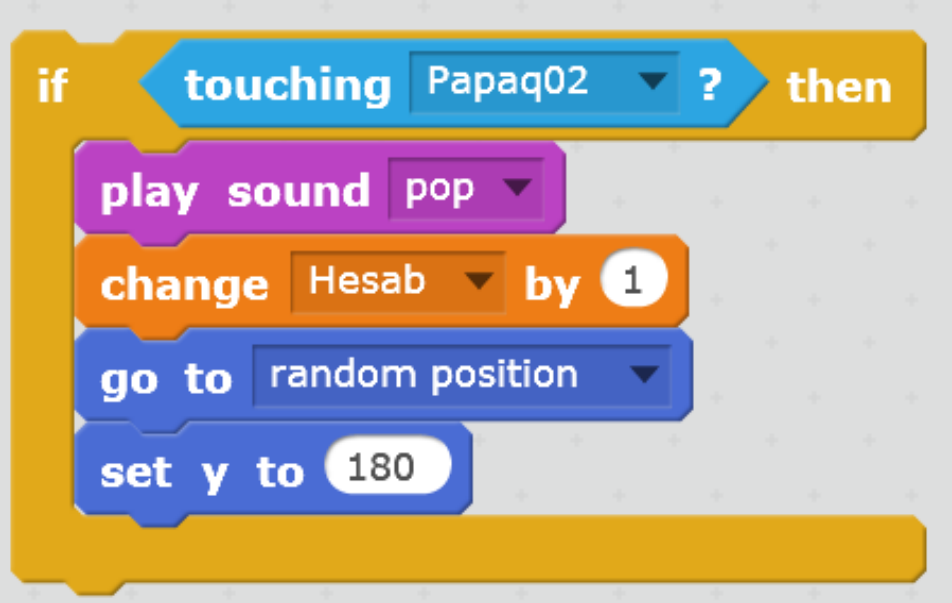


tonqal adlı personaj üçün bütün əmrlər ümumilikdə belə görünəcək:

Şəkərbura personajı üçün əlavə edilən əmrlər aşağıda verilib:

if_____then əmrində **Papaq02 obyektinə toxunma** (**touching Papaq02 ?**) şərti ödənəndə aşağıdakı əmrlər yerinə yetirilir:

- **pop** səsi çıxarılır.
- **Hesab** dəyişəninin qiyməti 1 vahid artırılır.
- İxtiyarı mövqeyə gedir.
- **y** dəyişəninə 180 qiyməti mənimsədir. Ən yuxarı sətrin nömrəsi.

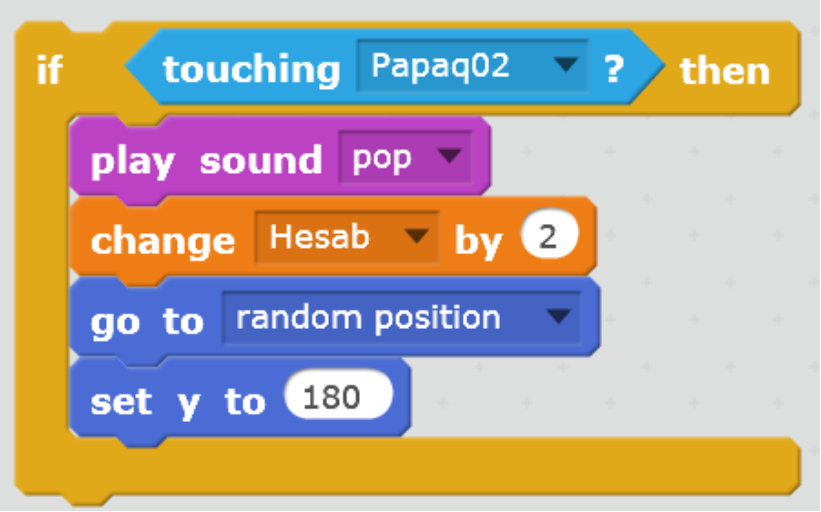


Bu dəyişikliklərdən sonra **Şəkərbura** personajının əmrləri bütövlükdə belə görünəcək.

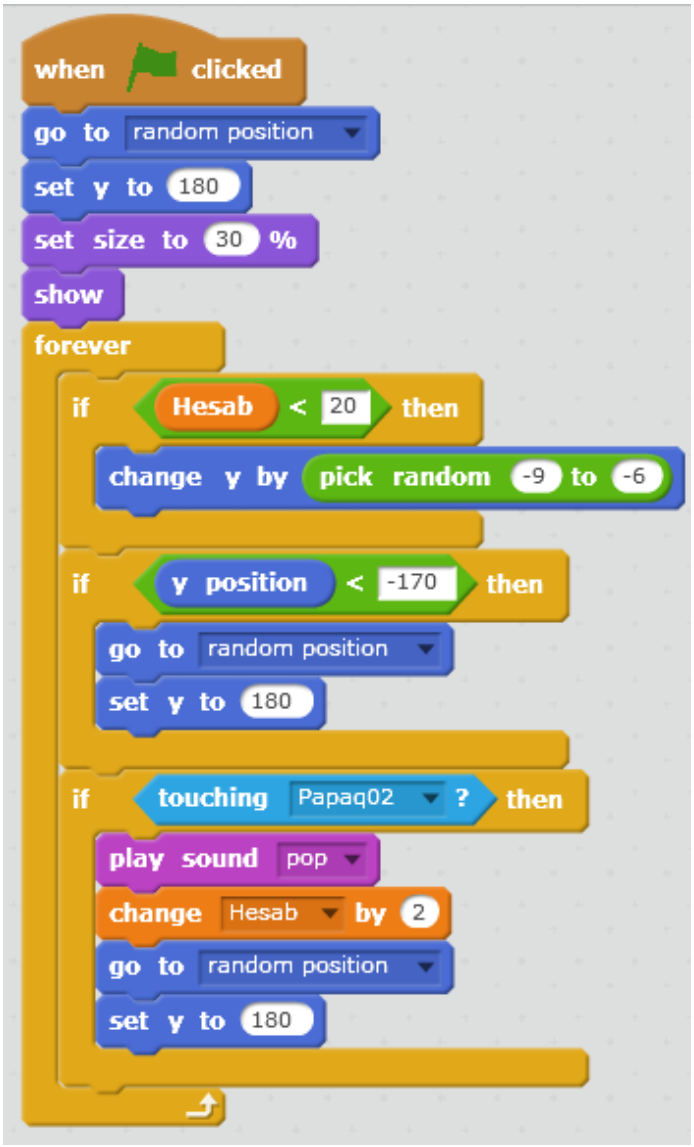
Paxlava personajı üçün əlavə edilən əmrlər aşağıda verilib:

if_____then əmrində **Papaq02 obyektinə toxunma** (**touching Papaq02 ?**) şərti ödənəndə aşağıdakı əmrlər yerinə yetirilir:

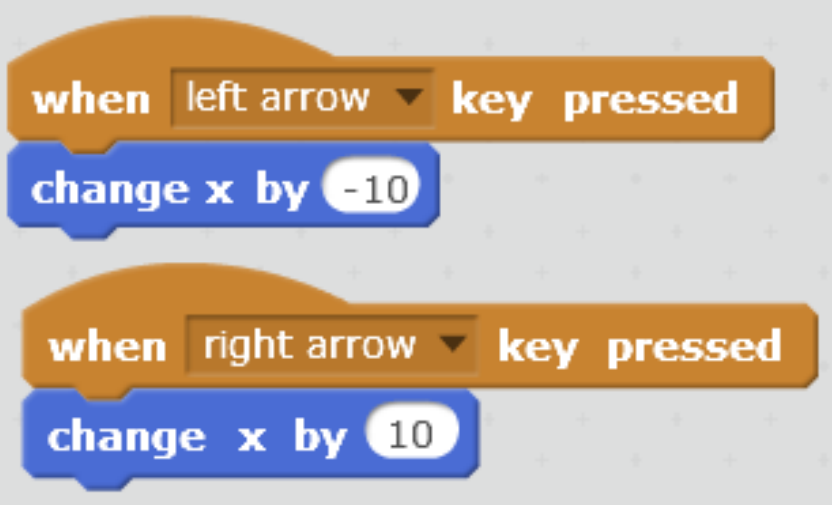
- **pop** səsi çıxarılır.
- **Hesab** dəyişəninin qiyməti 2 vahid artırılır.
- İxtiyarı mövqeyə gedir.
- **y** dəyişəninə 180 qiyməti mənimsədilir. Ən yuxarı sətrin nömrəsi.



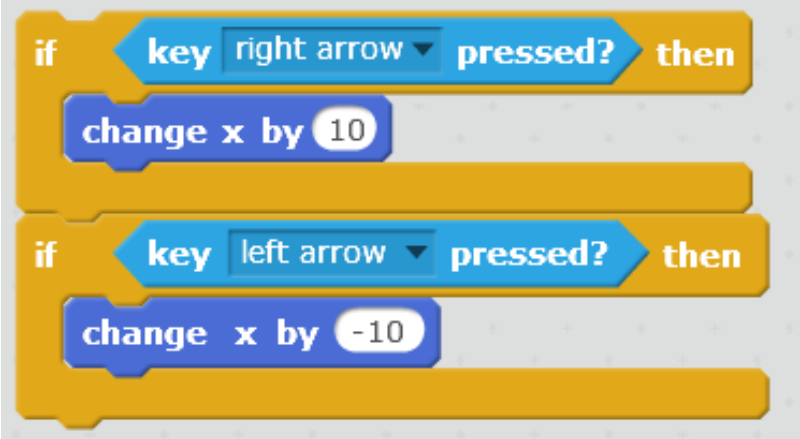
Bu dəyişikliklərdən sonra **Paxlava** personajının əmrləri bütövlükdə belə görünəcək.



Tam ekran rejiminə keçək, **Yaşıl bayraq** düyməsini sıxaq və proqramı başladaq. Aha, proqramımız işləyir. Xallar toplanır, həyatların sayı azalır. Amma fikir verirsinizsə, sola və sağa oxları ilə idarəetmə bir qədər ağır işləyir. Gəlin oxlarla idarəetməni bir qədər başqa formada edək. **Papaq** personajı üçün yazılan əmrlərdə

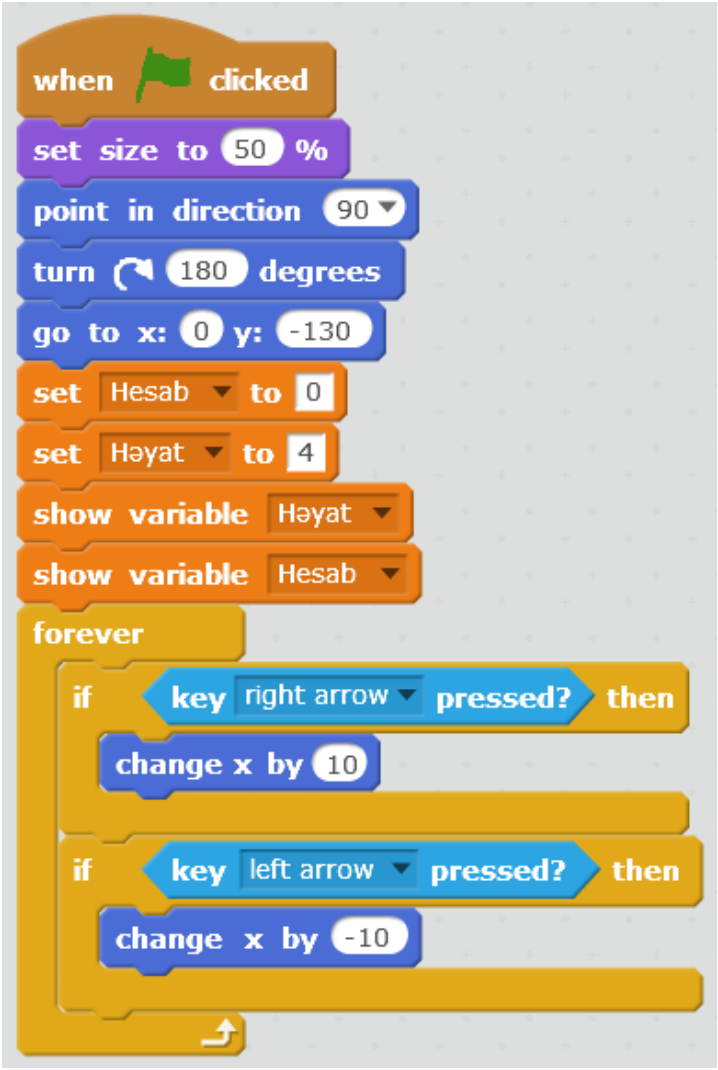


əmrlərini ləğv edək və onların yerinə aşağıdakı əmrlər blokunu **forever** əmrinin daxilinə salaq.



Tam ekran rejiminə keçək. Proqramı başladaq. Artıq oxlarla işləyəndə ləngimələr yoxdur. İstədiyimizə nail olduq.

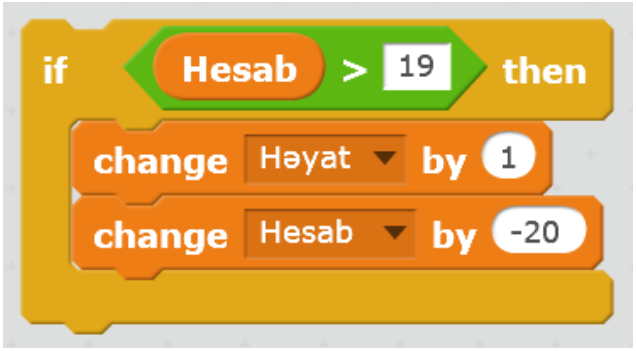
Bu dəyişiklikdən sonra görünüş aşağıdakı formada olacaq:



Proqramı işlədərkən görürük ki, oyunçu uduzanda (**tonqal** personajı üçün yazılmış proqramın işinin nəticəsi) ekranda sağdakı görünüş alınır və proqram işini tamamlayır.

Yuxarıda dediyimiz kimi, şəkərbura və ya paxlava papağa toxunanda **Hesab** dəyişəninin qiyməti şəkərbura üçün 1 vahid, paxlava üçün 2 vahid artırılır. **Hesab** dəyişəninin qiyməti 20 və ya daha çoxdursa, onda **Həyat** dəyişənin qiyməti 1 vahid artırılır. **Hesab** dəyişəninin qiyməti 20 xal azaldılır. **Hesab** dəyişəninin artırılmasını nəzərdən keçirdik.

Bəs **Hesab** dəyişəninin qiyməti 20 və ya daha çoxdursa, onda nə etməli: **Həyat** dəyişənin qiymətini 1 vahid artırmalı və **Hesab** dəyişəninin qiymətini 20 xal azaltmalı? Bunun üçün **Papaq** personajına aşağıdakı əmrləri əlavə etmək lazımdır:

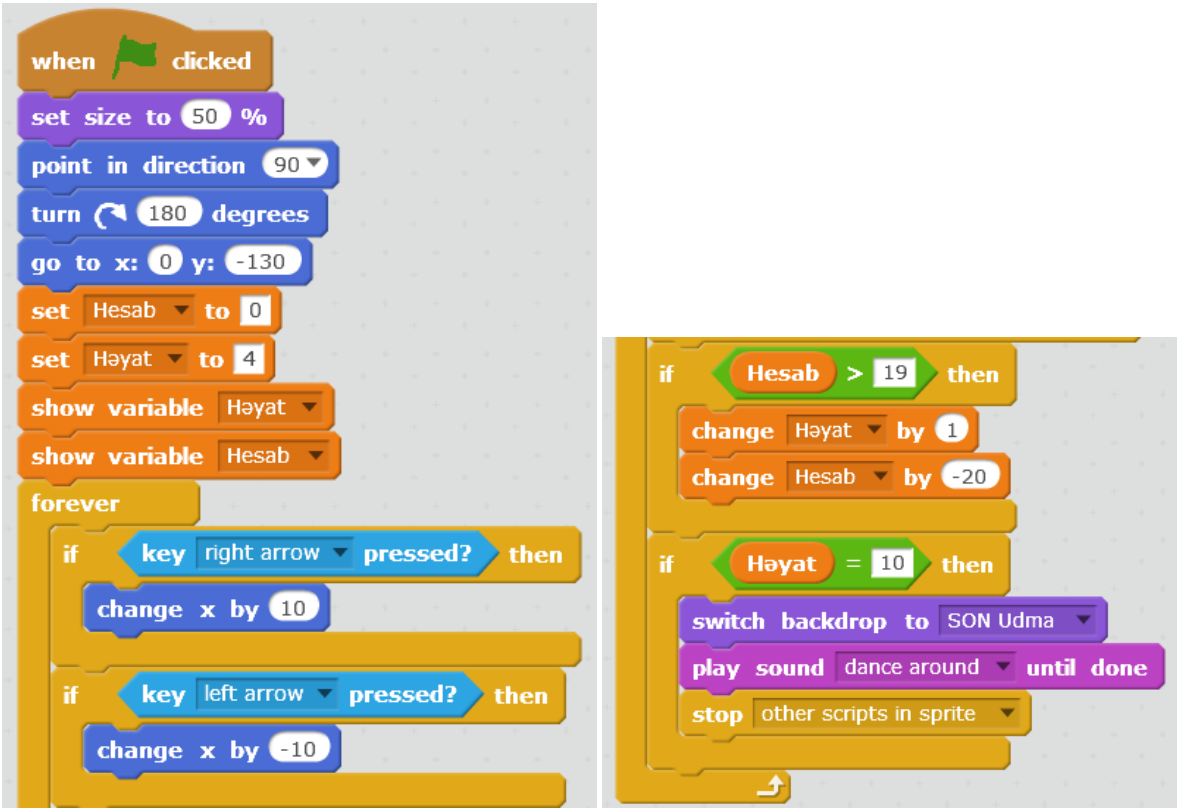


Əmrdən göründüyü ki, **Hesab** dəyişənin qiyməti 19-dan artıq olarsa, onda **Həyat** dəyişəninin qiyməti 1 vahid artırlılır. Növbəti əmrlə isə **Hesab** dəyişəninin qiyməti 20 vahid azaldılır. Nə üçün burada 20 yox, 19 götürülüb? Məsələ bundadır ki, Scratch proqramında böyükdür-bərabərdir şərtini yoxlayan operator yoxdur. Belə halda "20-dən böyük-bərabərdir" şərtini eyni güclü "19-dan böyükdür" şərti ilə əvəz edirik. Bu əmrlər blokunu **forever** əmrinin daxilinə əlavə etmək lazımdır.

Bəs oyunçu udanda, yəni **"10 həyat qazananda"** nə etmək lazımdır? Bu halda da **forever** əmrinin daxilinə aşağıdakı əmrləri əlavə edirik:

Əmrlərə nəzər salsaq görərik ki, Əgər **Həyat** dəyişəni 10 qiymətini alanda fonda **Son Udma** adlı təsvir görünəcək. Bundan əlavə **dance around** adlı musiqi səslənəcək və bütün digər əmrlər (skriptlər) işini dayandıracaq.

Əmrlərin ümumi görüntüsü belə olacaq (Əmrlərin görüntüsü böyük olduğundan səhifəyə yerləşmir. Bu səbəbdən 2 yerə bölünüb):

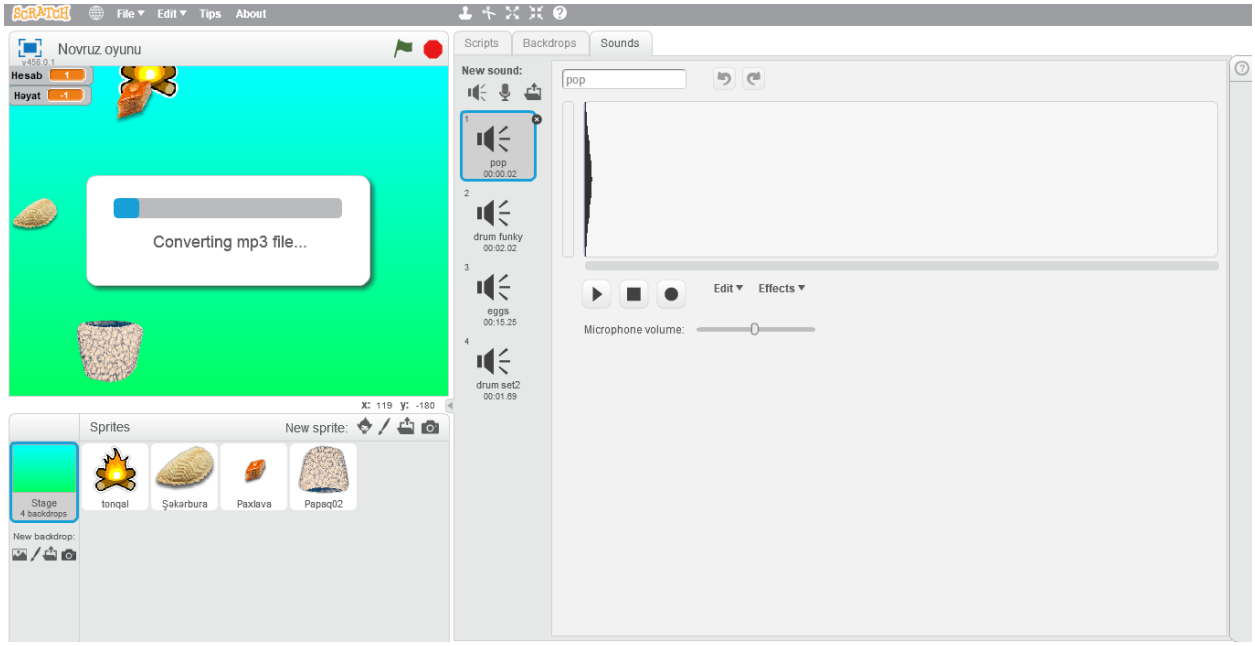


Artıq tam ekrana keçib oyunu oynaya bilərik.

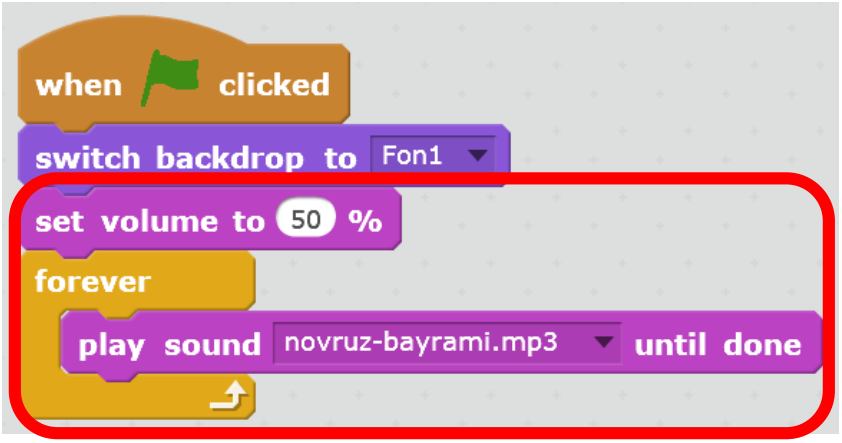
Əla! Oyun istədiyimiz formada işləyir. Amma yenə də nəsə çatmır. Oyunda bahar əhval-ruhiyyəsi duyulmur.

Gəlin oyun oynanılan zaman oyuna arxa səs fonu əlavə edək. Bu məqsədlə sevimli müğənnimiz Məmmədbağır Bağırzadənin ifasında **"Novruz bayramı"** adlı gözəl mahnını **mp3** formatda internetdən yükləyək. Daha sonra səhnəni seçək. **Sound** rejiminə keçək. Həmin mahnını əlavə fayl kimi səhnəyə əlavə edək.

Şəkildən göründüyü kimi, həmin musiqi **mp3** formatdan Scratch proqramının daxili audio fayl formatına çevrilir.



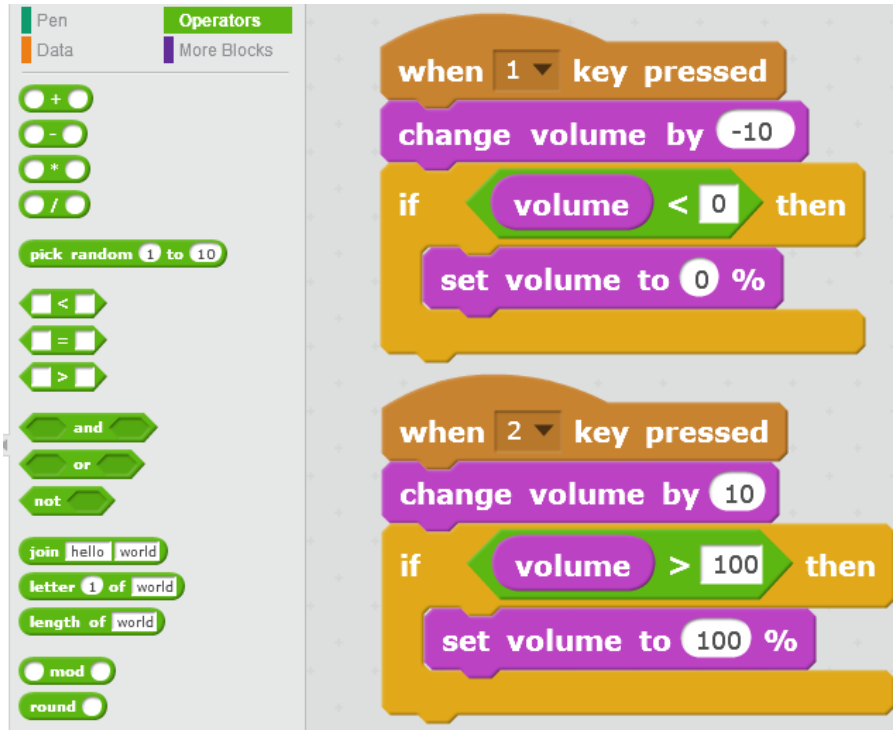
Çevrilmə bitdikdən sonra səhnə üçün olan yazdığımız proqrama seçdiyimiz musiqini ifa edən aşağıdakı əmrləri əlavə edək:



Görürük ki, səsin ucalığını 50% götürmüşük.

Artıq proqramı başladanda bu gözəl musiqi səslənir.

Bəs görəsən musiqinin səsini oyunun gedişi zamanı artırıb-azalda bilərikmi? Bunun üçün klavişlə idarəetmə əmrindən istifadə etmək olar. Gəlin bizdə “1” klavişini səsi azaltma, “2” klavişini səsi ucaltma üçün istifadə edək. Əmrlər aşağıdakı kimi olacaq:



Yuxarıda dediyimiz kimi, başlanğıcda səsin ucalığını 50% götürürük. Oyun prosesində "1" və "2" klavişlərini sıxmaqla səsin ucalığını artırıb-azaldırıq. Amma burada bir halı nəzərə almaq lazımdır ki, səsin ucalığı mənfi ola bilməz. Həmçinin 100%-dən də çox ola bilməz.

Sonda yenidən tam ekran rejiminə keçin, proqramı başladın, bahar əhval-ruhiyyəsini hətta qışda da yaşayın və oyundan həzz alın. İstədiyiniz vaxt səsin yüksəkliyini dəyişin.

Ev tapşırığı: Bu proqramda "3" klavişini sıxdıqda səsin tamamilə kəsilməsi (**mute** rejimi) üçün əmrlər bloku yazın.

6. Futbol

Yəqin ki, dünyada idmanı və onun populyar növü olan futbolu sevməyən az adam olar. Gəlin növbəti proqramımızı futbola həsr edək. Əvvəlcə pişik spraytını silək. Yeni sprayt daxil etmək üçün **New sprite:** bölməsindəki yeni sprayt əlavə etmək üçün birinci düyməni sıxıb spraytlar kitabxanasına daxil olaraq **People** qrupundan **Jaime Walking** spraytını seçirik. Bu sprayta üstünlük verməyimizin səbəbi onun duruşu və hərəkət almaq üçün 5 kostyumunun olmasıdır. Əlbəttə ki, Siz digər bir spraytı seçə, ya da İnternetdən istədiyiniz başqa bir personajı yükləyə bilərsiniz. Sadəcə olaraq həmin personaj çox kostyumlu və uyğun duruşlu olmalıdır. **Jaime Walking** spraytının adını dəyişək. Məsələn **Qurban** qoyaq (Azərbaycanda populyar futbolçu olmuş və hal-hazırda məşqçilik fəaliyyətilə məşğul olan **Qurban Qurbanovun** şərəfinə).
İkinci bir sprayt kimi **Sports** mövzusundan **Ball-Soccer** adlı topu götürürük. Bu spraytın da adını dəyişib **Top** qoyaq.
Bizə bir sprayt da düzəltmək lazımdır. Daxili qrafik redaktordan istifadə edərək yaşıl rəngdə şaquli vəziyyət də durmuş düzbucaqlı qururuq. Bu düzbucaqlı qapının arxasında yerləşdiriləcək və top ona toxunanda qol hesaba alınacaq.

Əlbəttə ki, söhbət futboldan gedirsə, bizə mütləq arxa fonda futbol meydançasının şəklini yerləşdirmək lazımdır. Bunun səhnəni seçirik. Interfeysin **Stage** bölməsində **New backdrop** sahəsində **Yeni fon** düyməsini sıxmaqla səhnə üçün nəzərdə tutulmuş kitabxanadan **Sports** mövzusundan **goal1** fonunu seçirik.
Dediklərimizi səhnədə təxminən aşağıda gördüyümüz formada yerləşdiririk:

Burada vacib olan yaşıl düzbucaqlının yerləşməsidir. Oyunçu ilə topun yerini proqram özü tənzimləyəcək.

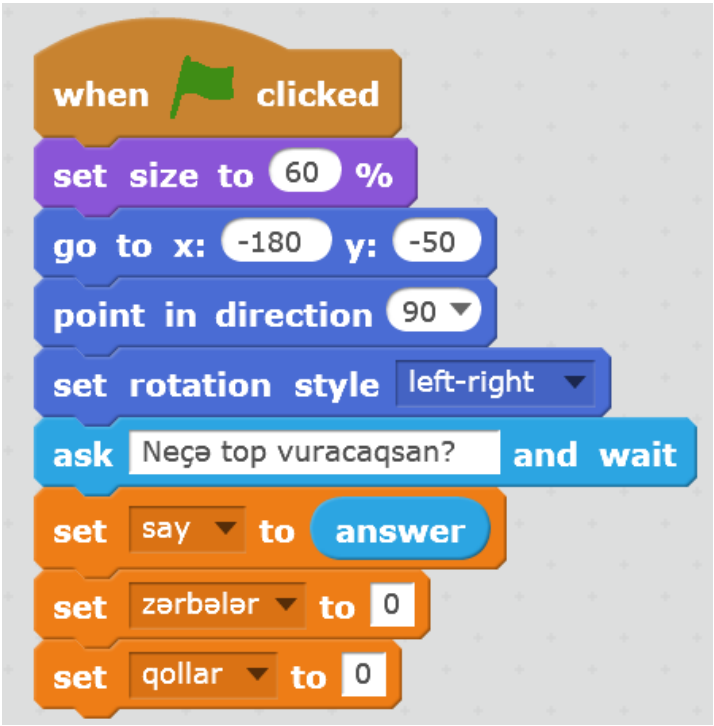
Proqramda vurulası zərbələrin maksimal sayını, hər dəfə vurulan zərbələrin sayını və qolların sayını göstərmək üçün bizə 3 dəyişən lazımdır. Bunlar aşağıdakılardır:

- say - vurulası zərbələrin maksimal sayı;
- zərbələr - vurulan zərbələrin sayı;
- qollar – vurulan qolların sayı.

Bunu **Data** qrupundan istifadə etməklə həyata keçirəcəyik.

zərbələr və **qollar** dəyişənlərinin qarşısındakı xanaları qeyd etməklə onları səhnədə görünən edirik. Görüntü aşağıdakı kimi olacaq:

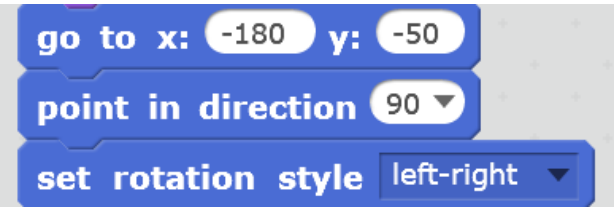
Əvvəlcə futbolçu üçün (QURBAN adlı sprayt) sağdakı əmrlərdən ibarət proqram yazaq:



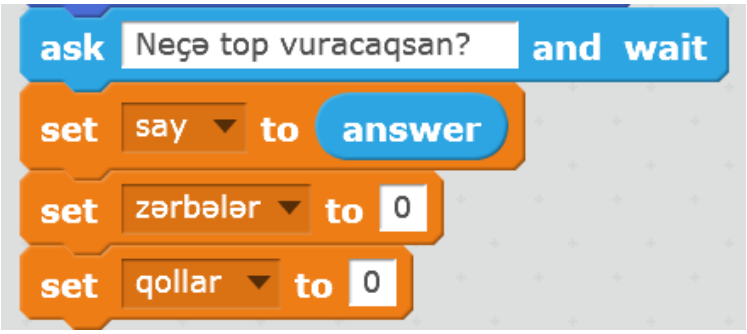
Yaşıl bayrağı sıx əmri proqramı başlatmaq üçündür.

Oyunçu spraytının ölçüsü səhnə üçün böyükdür. Onun orijinal ölçüsünün 60%-ni götürürük. Bunun üçün **set size to 60%** əmrindən istifadə edirik.

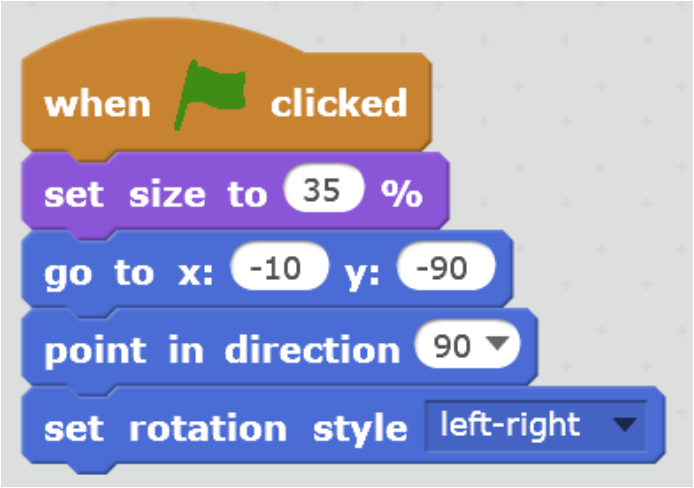
Növbəti 3 əmr futbolçunun yerini, istiqamətini və hərəkət stilini müəyyən edir.



Bundan sonra istifadəçidən neçə top vurmaq istəməsi soruşulur. Alınan cavab **say** adlı dəyişəninə mənimsədilir. Daha sonra **zərbələr** və **qollar** dəyişənlərinə “0” (sıfır) qiyməti mənimsədilir. Həmin əmrləri aşağıda görürüsünüz.



Oxşar əmrləri **Top** spraytı üçün də yazırıq:



Tam ekran rejiminə keçib **Yaşıl bayrağı sıx** əmri ilə proqramı başladırıq. Görürük ki, futbolçu və top ölçülərini dəyişdilər, ilkin olmaları gərək olan nöqtəyə keçdilər və istiqamətlərini sağa yönəltdilər.

Keçək futbolçu ilə top arasında əlaqəni qurmağa. Bunun üçün **Events** qrupundakı mesajlaşma əmrləridir. **broadcast (ötür)** və **when I receive (Mən alanda)** əmrləridir.

Bizim misalda bir əmr **"qol"** mesajını ötürür, digəri isə həmin mesajı alır.

Oyunçumuz **say** qədər zərbə vurmaq imkanına malikdir. Ona görə də hər zərbəni vurduqdan sonra zərbələrin sayı ümumi sayla müqayisə edilməli və əgər qol olubsa, qolların sayı hesablanmalıdır.

Klaviaturada **probel (space)** klavişi sıxılanda oyunçu ilkin vəziyyətini alır (zərbələr çox olduğundan başlanğıc anında aparılan yerdəyişmə sonrakı hallar üçün keçərli olmur).

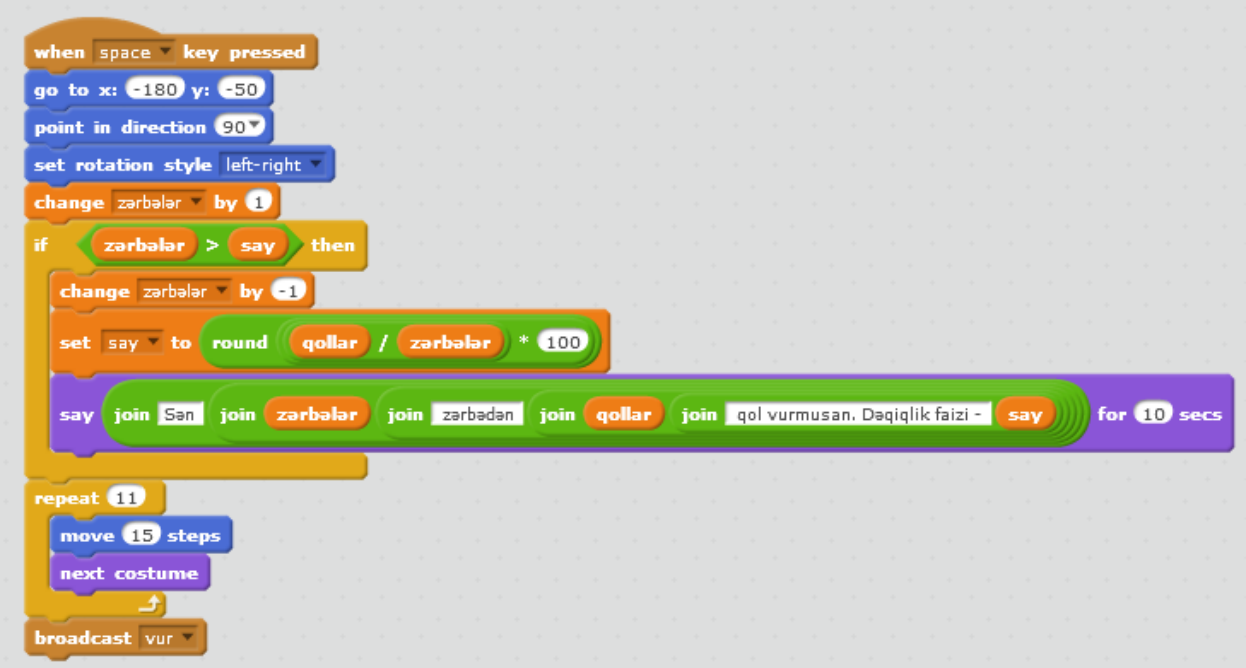
Daha sonra **zərbələr** dəyişəni bir vahid artırılır. Əgər zərbələrin sayı ümumi sayı **(say)** aşırsa, onda dəqiqlik faizi hesablanır və artıq proqramda istifadəsinə ehtiyac olmayan **say** dəyişəninə mənimsədilir. Sonda ekranda zərbələrin sayı, qolların sayı və dəqiqlik faizi çap edilir.

Yox əgər hələ zərbə vurmaq şansı varsa, onda **repeat** əmrinin köməyilə futbolçu 11 dəfə 15 addım irəli gedir. Onun qaçması effektini almaq üçün həm də kostyumunu növbə ilə dəyişirik.

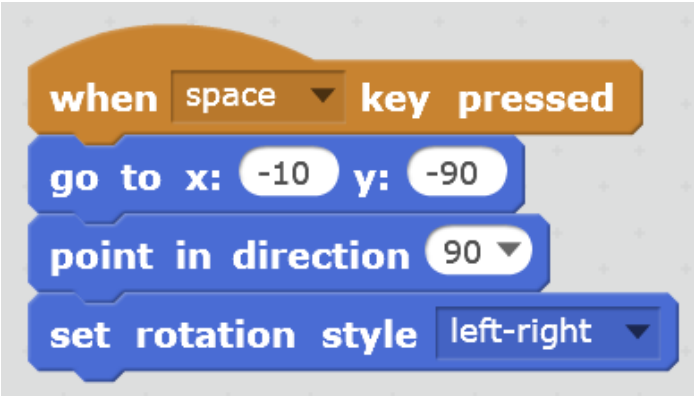
Lap sonda **broadcast** əmri ilə **"vur"** mesajını göndəririk.

Bəs **"vur"** mesajını harada qəbul edəcəyik. Yəqin bildiniz, **Top** spraytında.

Futbolçu üçün sadaladığımız əmrlər bütövlükdə aşağıda verilib:



Bəs **Top** spraytı üçün hansı əmrlər olmalıdır? Əvvəlcə o klaviaturada **probel (space)** klavişi sıxılanda ilkin vəziyyətini alır (zərbələr çox olduğundan başlanğıc anında aparılan yerdəyişmə sonrakı hallar üçün keçərli olmur).



İkinci əmrlər blokunda isə **“vur”** mesajını alanda yerinə yetirəcəyi əmrlər öz əksini tapır.

when I receive (Mən alanda) əmri **“vur”** mesajını alanda top hərəkətə başlamalıdır. Bildiyimz kimi futbolda qapıya zərbə endirəndə zərbələrin hamısı qapıya çatmır. Bu effekti almaq üçün **pic random in -40 to 40** əmrindən istifadə etməklə topun istiqamətini **turn** əmrilə ixtiyari döndəririk. Bundan sonra 26 dəfə 10 addım irəliləyirik.

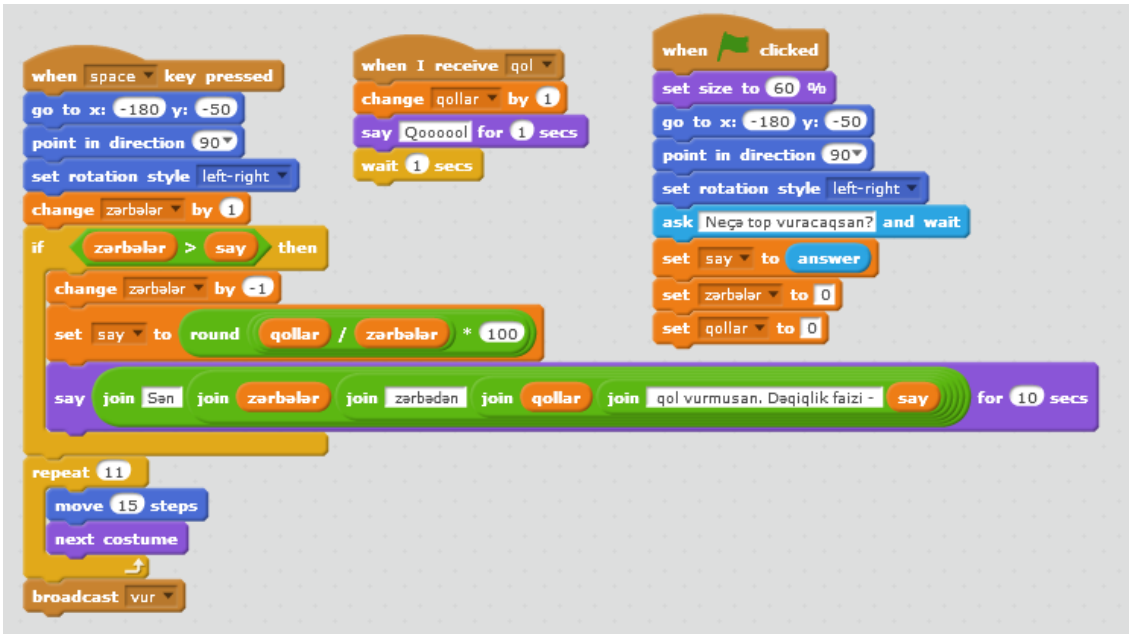
Bu əməliyyatdan sonra topun qapının arxasında yerləşən yaşıl rəngli sahəyə dəyib-dəymədiyini yoxlarıq. Əgər yaşıl sahəyə dəyibsə, onda top ilkin nöqtəyə gedir və **"qol"** mesajı göndərir. Əmrlər belədir:

Bəs **"qol"** mesajı harada alınacaq? Əlbəttə **"qol"** mesajını futbolçu alacaq. Bu halda **qollar** dəyişəni bir vahid artırılacaq və futbolçu **"Qoooool"** deyə səslənəcək (ekranda mətn görünəcək). Aşağıda həmin proqram kodu verilib.

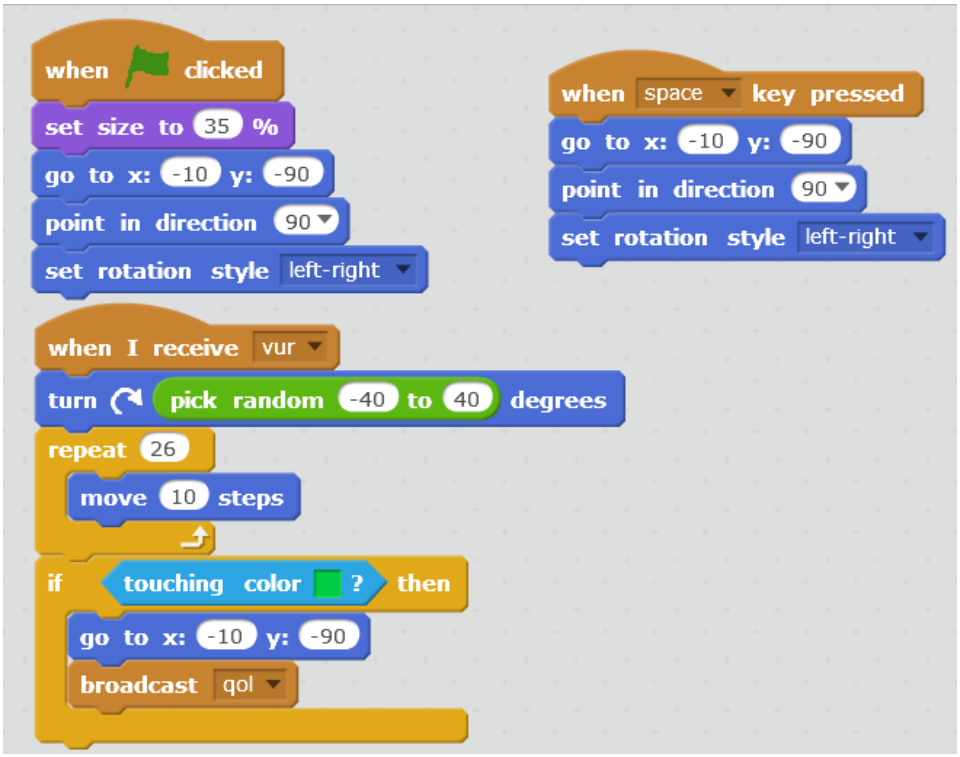
Artıq Siz tam ekran rejiminə keçib proqramı başlada bilərsiniz.

Görəcəksiniz ki, proqram bütün hesablamaları dəqiq aparır.

Oyunçu üçün son proqram kodları aşağıdakı kimidir:



Top üçün son proqram kodu isə belə olacaq:



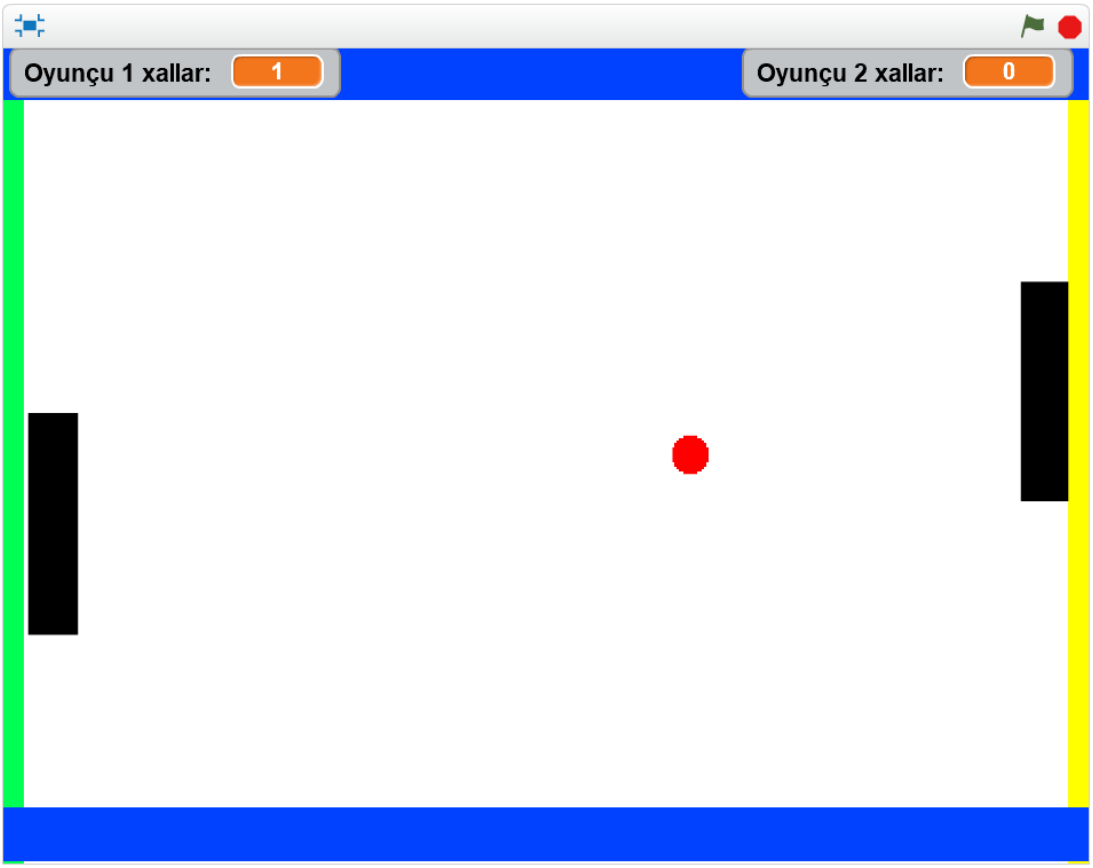
Yəqin ki, futbolsevərlər üçün maraqlı oldu.

7. TENNİS

İdmanla bağlı ikinci proqramımızı şərti olaraq TENNİS adlandırmışıq. Çünki bu oyunda tennis oyununun elementləri var.

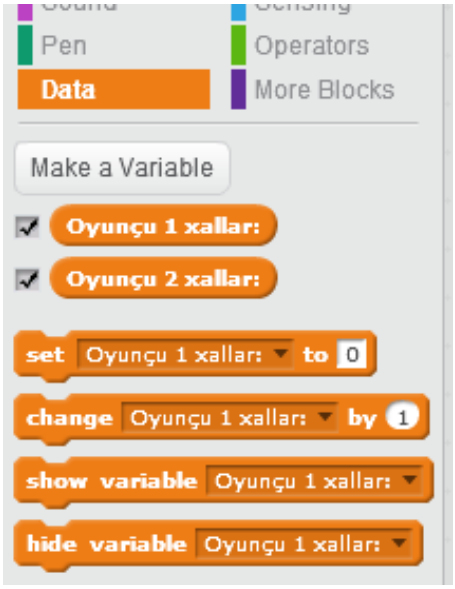
Oyunun qaydası belədir. Meydan 4 divarla məhdudlanır. İki divar (sol və sağ) oyunçular üçün ayrılıb. Yan divarlar neytraldır. Top yan divara dəyəndə geri sıçrayır və oyun davam edir. Oyunçuların qoruduğu divara dəyəndə isə həmin oyunçu bir xal itirmiş olur, yəni digər oyunçuya xal yazılır. Oyunçunun məqsədi ona aid olan divarı qorumaq və gələn topu vurmaqdır. Bunun üçün o divar boyu hərəkət edə bilir və əlindəki raketka ilə topu geri qaytarır. Hansısa oyunçu müəyyən sayda xal toplayanda oyun bitmiş hesab edilir.

Gəlin indi bu oyunun informasiya modelini quraq. Aşağıdakı şəkildə oyunun interfeysi göstərilib.

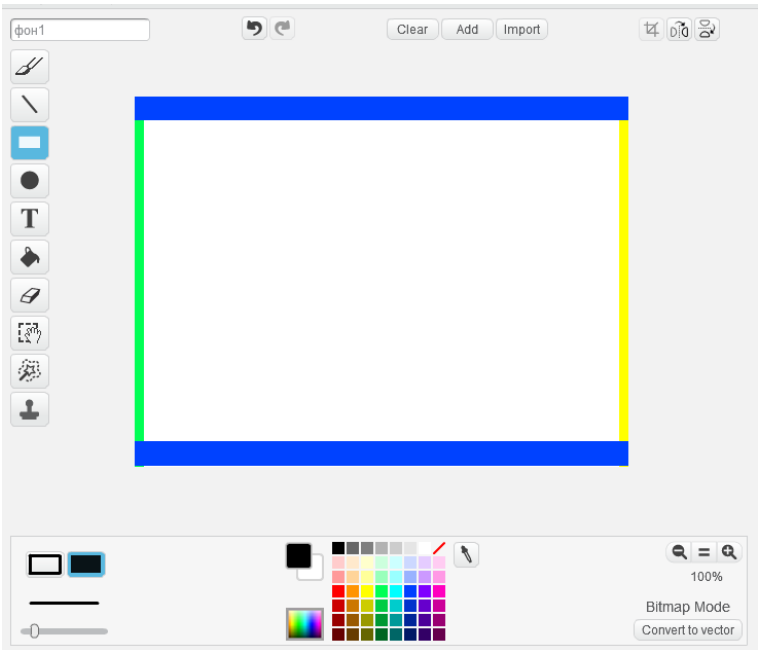


Yan (neytral) divarlar göy rəngdə verilib. Sol divar yaşıl (Oyunçu 1 üçün nəzərdə tutulmuş divar), sağ divar (Oyunçu 2 üçün nəzərdə tutulmuş divar) isə sarı rəngdə verilib. Qara rənglə 2 oyunçunun raketkası verilib. Soldakı qara düzbucaqlı yaşıl divarı, sağdakı qara düzbucaqlı isə sarı divarı qorumalıdır. Topun rəngi fərqli olmaq üçün qırmızı götürülüb. Ekranın yuxarı tərəfində Oyunçu 1 və Oyunçu 2 üçün xalları göstərən tablo verilib.

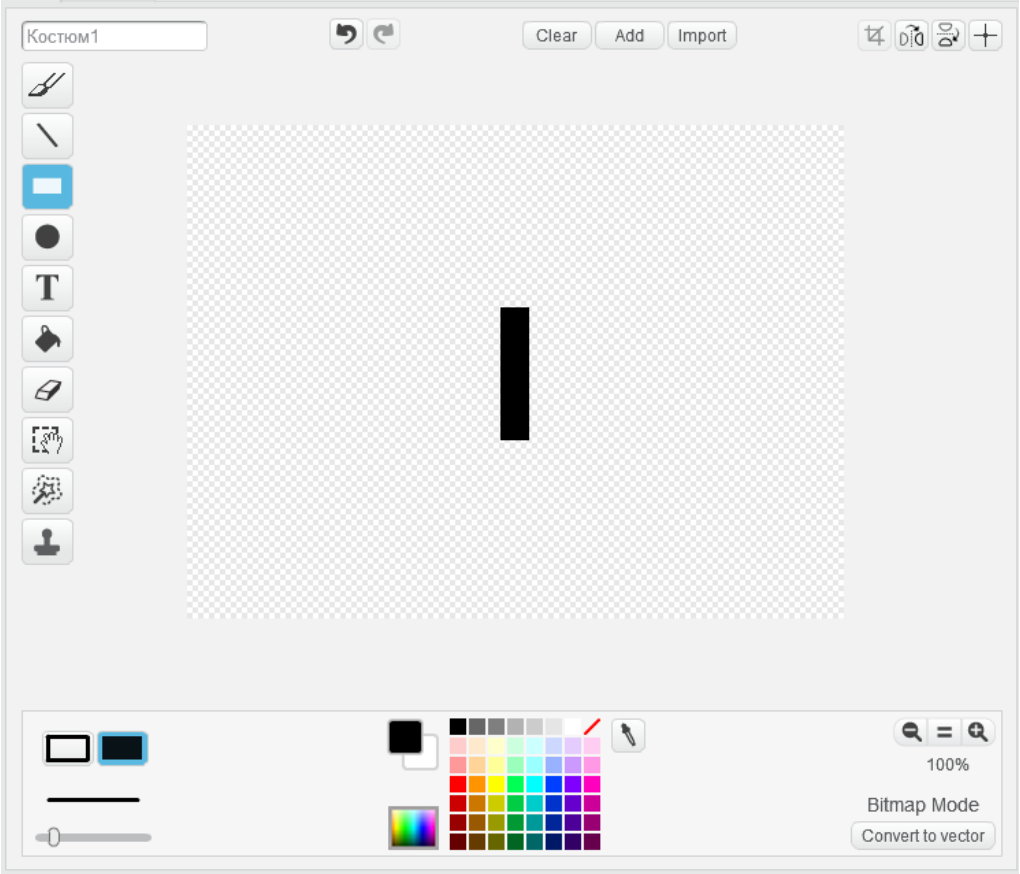
Oyunun modeli belə olacaq. Soldakı oyunçu yuxarı-aşağı uyğun olaraq ingilis hərflərini göstərən **“Q”** və **“A”** klavişləri ilə idarə edilir. Sağdakı oyunçu isə yuxarı-aşağı uyğun olaraq yuxarı və aşağı oxlarla idarə edilir.
Yaşıl bayraq düyməsi ilə oyunu başladanda qırmızı rəngli top meydança boyu xaotik hərəkət edir. Oyunçu top onun divarına yaxınlaşanda onu vurmalıdır. Əgər bunu edə bilməsə digər oyunçuya xal yazılır. Bu xallar yuxarıdakı tabloda öz əksini tapır.
Əvvəlcə **Data** qrupuna girib **Oyunçu 1 xallar:** və **Oyunçu 2 xallar:** adlı iki dəyişən yaradırıq. Onları görünən etmək üçün dəyişənlərin adlarının qarşısındakı xanaları qeyd edirik.



Daha sonra həmin dəyişənləri səhnədə yuxarıdakı şəkldə göründüyü formada yerləşdiririk.
İkinci addım meydançanı yaratmaqdır. Bunun üçün səhnəni seçirik. Daxili qrafik redaktora keçib səhnənin kənarlarını aşağıdakı formada haşiyələyirik.



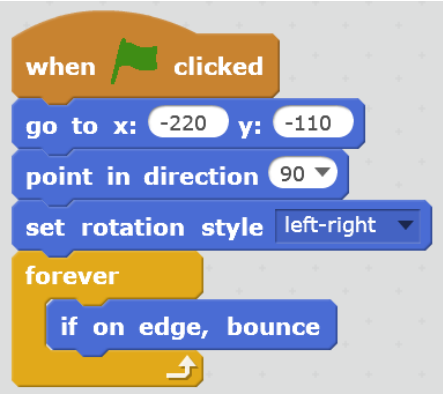
Qara rəngli raketkaları çəkmək üçün də daxili qrafik redaktordan istifadə edəcəyik. Aşağıdakı şəkildə gördüyünüz formada içi qara rənglə dolmuş düzbucaqlı çəkirik. Çalışın ki, düzbucaqlının eni çox dar olmasın.



Qaldı topun yaradılmasına. Bunun üçün də daxili qrafik redaktordan istifadə edəcəyik. Bir şeyi qeyd etmək lazımdır ki, topu çəkərkən rəngi qırmızı seçin və çalışın ki, onun radiusu qara düzbucalının enindən az olsun. Əks halda top divara toxunanda arzu olunmaz hərəkətlər baş verə bilər.

Proqramı yazmaq üçün artıq hər şey hazırdır. Əvvəlcə hər bir oyunçu üçün raketkaları idarə edən proqrqram yazaq.

Oyunçu 1 üçün bu proqram fraqmenti belə olacaq:

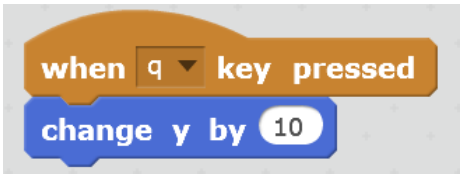


Yaşıl bayraq düyməsi sıxılan kimi növbəti 3 əmr raketkanın mövqeyini, istiqamətini və hərəkət stilini müəyyən edir.

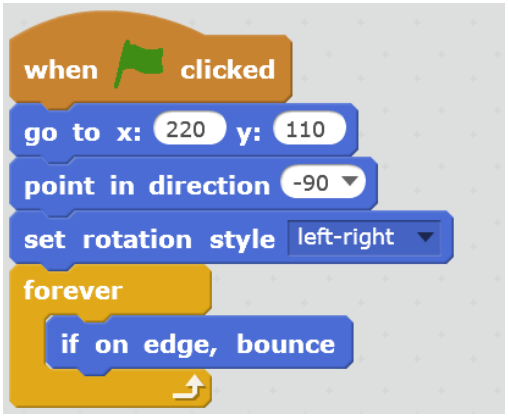
Əvvəlcə raketkanın mövqeyini müəyyən edirik. (**go to x: -220 y: -110**) əmri solda olan oyunçunun raketkasının mövqeyini səhnənin (-220; -110) nöqtəsində təyin edir.

İkinci əmrlə raketkanın istiqamətini, üçüncü əmrlə isə onun hərəkət stilini (sola-sağa) müəyyən edirik.

forever sonsuz dövr əmrinin daxilində olan **if on edge, bounce** əmri isə raketka kənara toxunanda geriyə təkan verəcək.
Oyunçu 1 üçün yuxarı-aşağı idarəetmə uyğun olaraq ingilis hərflərini göstərən **"Q"** və **"A"** klavişləri ilə idarə ediləcək. Bu əmrlər də belə olacaq:



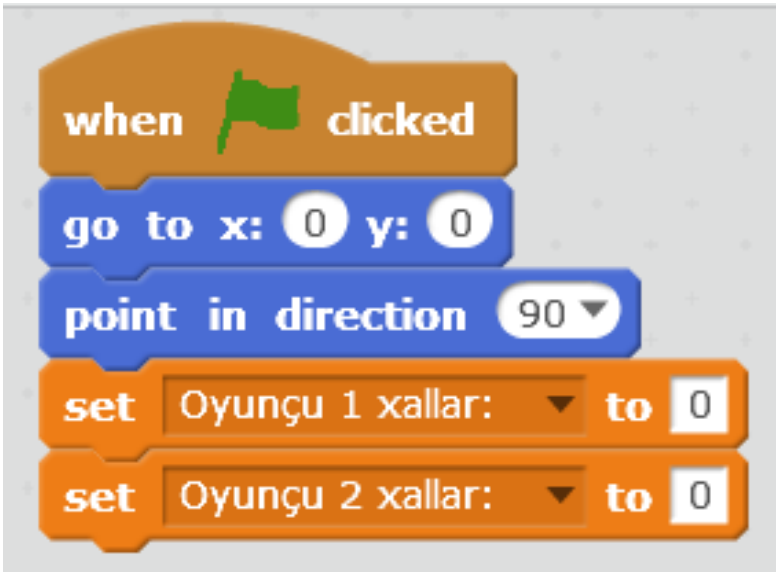
Oyunçu 2 üçün bu proqram fraqmenti belə olacaq:
Yaşıl bayraq düyməsi sıxılan kimi növbəti 3 əmr raketkanın mövqeyini, istiqamətini və hərəkət stilini müəyyən edir.



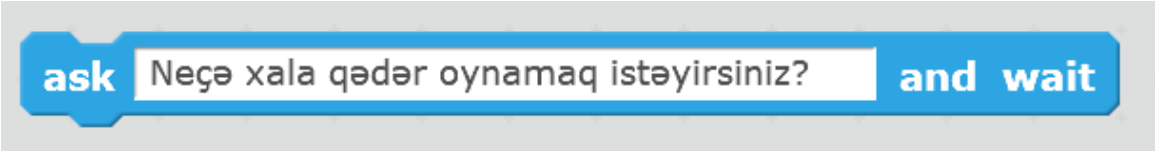
Əvvəlcə raketkanın mövqeyini müəyyən edirik. (**go to x: 220 y: 110**) əmri solda olan oyunçunun raketkasının mövqeyini səhnənin (220; 110) nöqtəsində təyin edir.
İkinci əmrlə raketkanın istiqamətini, üçüncü əmrlə isə onun hərəkət stilini (sola-sağa) müəyyən edirik.
forever sonsuz dövr əmrinin daxilində olan **if on edge, bounce** əmri isə raketka kənara toxunanda geriyə təkan verəcək.
Oyunçu 2 üçün yuxarı-aşağı idarəetmə uyğun olaraq **yuxarı** və **aşağı** ox klavişləri ilə idarə ediləcək. Bu əmrlər də belə olacaq:

Yoxlayın görək yazdığımız əmrlər işləyirmi?

İndi qaldı əsas obyektimiz olan top üçün proqram yazmaq. Top üçün başlağıc əmrlər aşağıda verilib. Bu əmrlər **Yaşıl bayraq**la proqramı başladır, topu səhnənin mərkəzinə gətirir, top üçün istiqamət seçir və **Oyunçu 1** və **Oyunçu 2** dəyişənlərinə sıfır qiyməti mənimsədir. Sonuncu ona görə lazımdır ki, oyunçular üçün xallar hesablananda real nəticə almaq olsun.

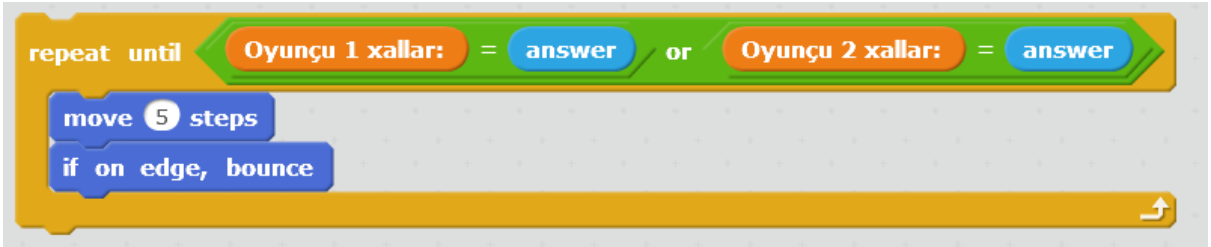


Bizim hazırladığımız tennis oyunu 2 nəfər oynamaq üçündür. Növbəti əmr oyunçulardan, onlardan biri neçə xal toplayanda, oyunu bitirməyi soruşur.



Bildiyimiz kimi bu əmr nəticəsində klaviaturadan daxil edilmiş informasiya **answer** sahəsinə (buferinə) yerləşdirilir. Bufer iki qurğu arasında körpü yolu kimi yaddaş rolunu oynayır.

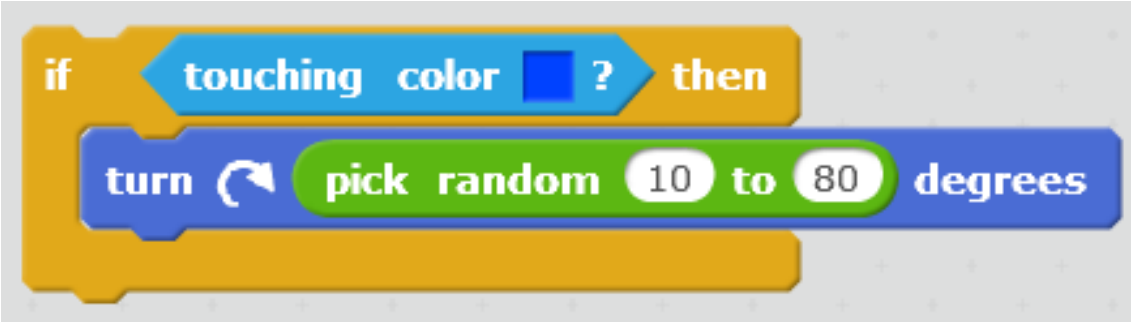
Artıq biz **repeat until** dövr əmrindən istifadə edərək dövr qura bilərik. Bu əmr şərtdə verilən məntiqi ifadə yerinə yetirilənə qədər dövrü təmin edəcək. Məntiqi ifadə isə belədir: **Oyunçu 1 xallar:** və **Oyunçu 2 xallar:** dəyişənlərindən biri maksimal xala bərabər olana qədər dövrün işi təmin edilir. Maksimal xal isə **answer** sahəsindədir (buferindədir).



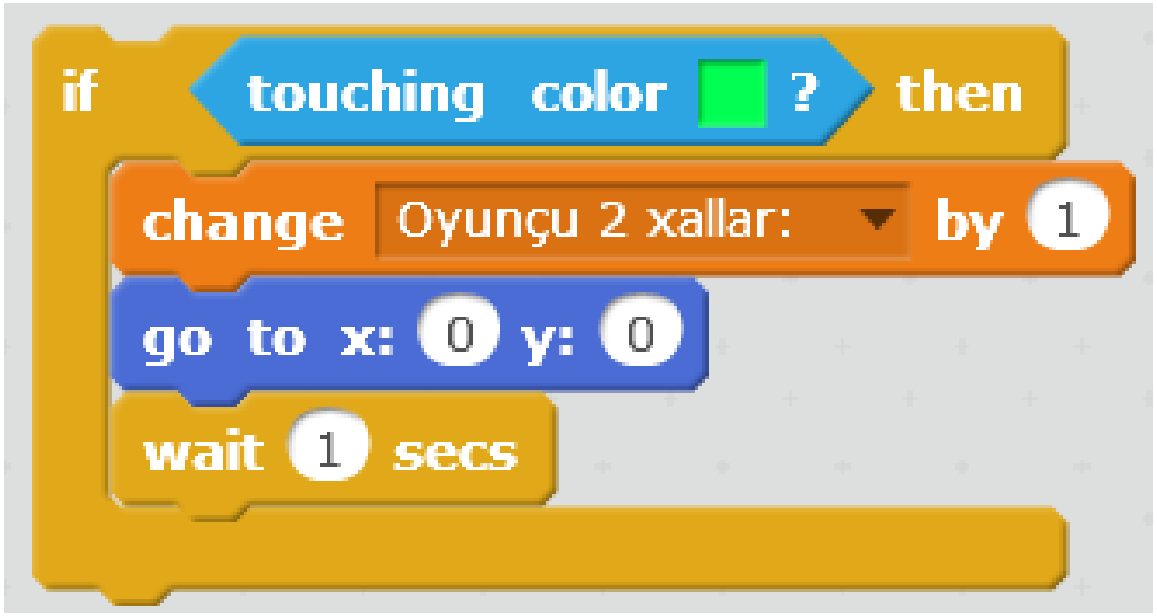
Fikir verirsinizsə, bu dövrün daxilinə ilkin olaraq 2 əmr daxil edilib: **5 addım getməli** və **kənara toxunduqda geriyə sıçramalı** əmrləridir. Başlanğıcda **Oyunçu 1 xallar:** və **Oyunçu 2 xallar:** dəyişənlərinin qiyməti sıfıra bərabər olduğundan dövr sona çatmayacaq. Deməli, biz dövrümüzü sona çatdırmaq üçün proqramımızı təkmilləşdirməliyik.

Birinci yeni salınası əmrlər bloku topun raketkaya toxunması nəticəsində nə baş verəcəyini idarə edir. Aşağıdakı əmrlərdən göründüyü kimi şərtə görə sprayt qara rəngə (raketkanın rəngi) toxunduqda onda sprayt müəyyən bucaq qədər sağa dönür. Bu bucağın qiyməti **pick random** əmri vasitəsilə 10 və 80 dərəcə arasında müəyyən edilir. İstəsəniz intervalı -80 ilə 80 arasında da müəyyən edə bilərsiniz.

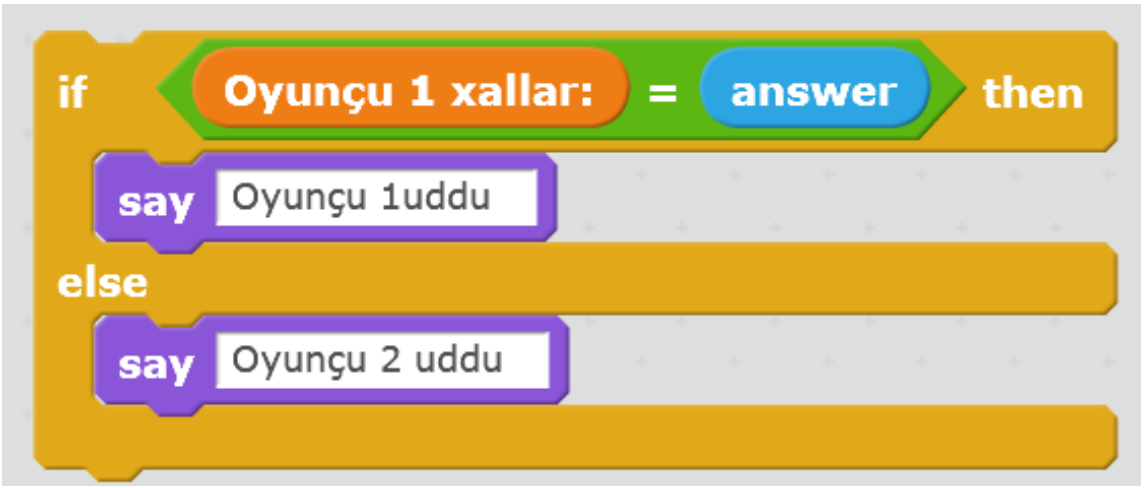
İkinci yeni salınası əmrlər bloku topun yan divara (göy rəng) toxunması nəticəsində nə baş verəcəyini idarə edir. Aşağıdakı əmrlərdən göründüyü kimi şərtə görə sprayt göy rəngə toxunduqda onda sprayt yenə müəyyən bucaq qədər sağa dönür. Bu bucağın qiyməti yenə də **pick random** əmri vasitəsilə 10 və 80 dərəcə arasında müəyyən edilir. İstəsəniz intervalı -80 ilə 80 arasında da müəyyən edə bilərsiniz.



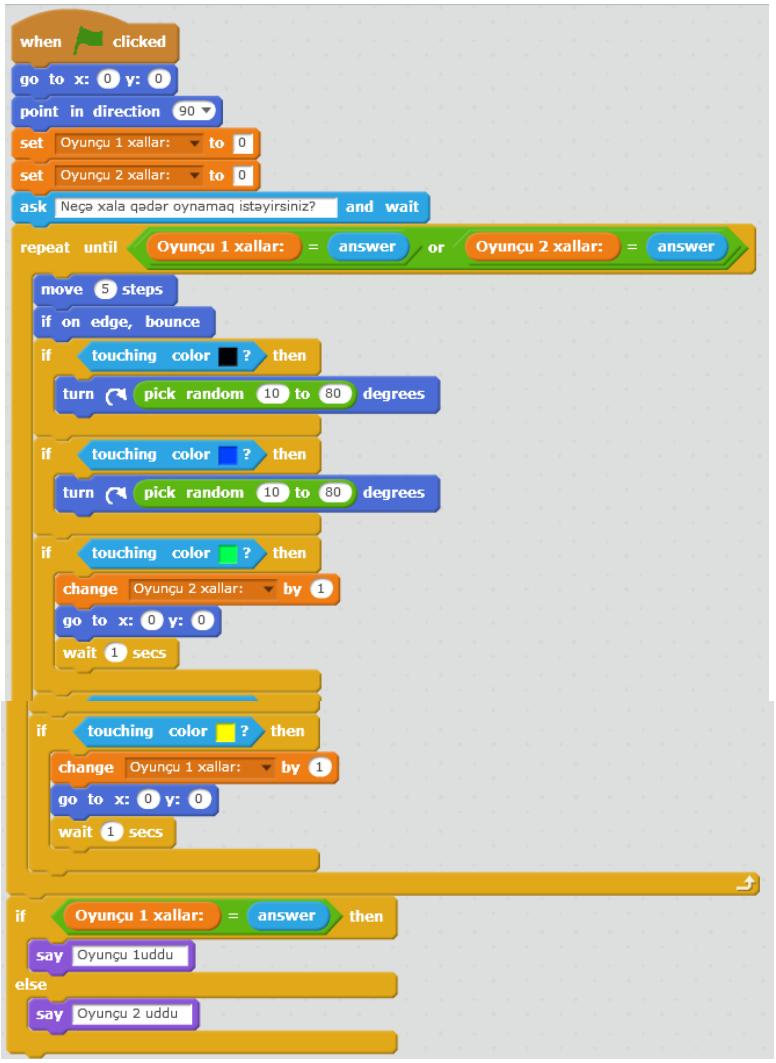
Növbəti yeni salınası 2 əmrlər bloku topun oyunçuların qoruduğu divara (yaşıl və sarı rəngdə) toxunması nəticəsində nə baş verəcəyini idarə edir. Şərtə görə sprayt yaşıl rəngə toxunduqda **Oyunçu 2 xallar:** dəyişəninin qiyməti 1 vahid artırılır. Top səhnənin mərkəzinə (0;0 koordinatına) aparılır v bir dəqiqə gözləmə baş verir.
Eyni hal sprayt sarı rəngə toxunduqda **Oyunçu 1 xallar:** dəyişəninin qiyməti 1 vahid artırılır. Top səhnənin mərkəzinə (0;0 koordinatına) aparılır və bir dəqiqə gözləmə baş verir.
Aşağıdakı əmrlərdə dediklərimiz öz əksini tapıb.



Dövr bitdikdən sonra proqram hansı oyunçunun qalib gəldiyini müəyyən edib ekranda uyğun informasiyanı çap etməlidir. Bunun üçün tam formalı **if** əmrindən istifadə edilir. Bizim halda birinci oyunçunun xallarının maksimal qiymətə malik olması yoxlanılır. Şərt ödənəndə birinci oyunçunun, əks halda ikinci oyunçunun qalib gəldiyi elan edilir. Biz burada yoxlamanı ikinci oyunçu üçün də edə bilərdik. Bu halda ikinci oyunçunun xallarının maksimal qiymətə malik olması yoxlanılır. Şərt ödənəndə birinci oyunçunun, əks halda ikinci oyunçunun qalib gəldiyi elan ediləcək. Bu əmrlər aşağıda verilib.



Proqramın tam mətni isə aşağıda göstərilib.

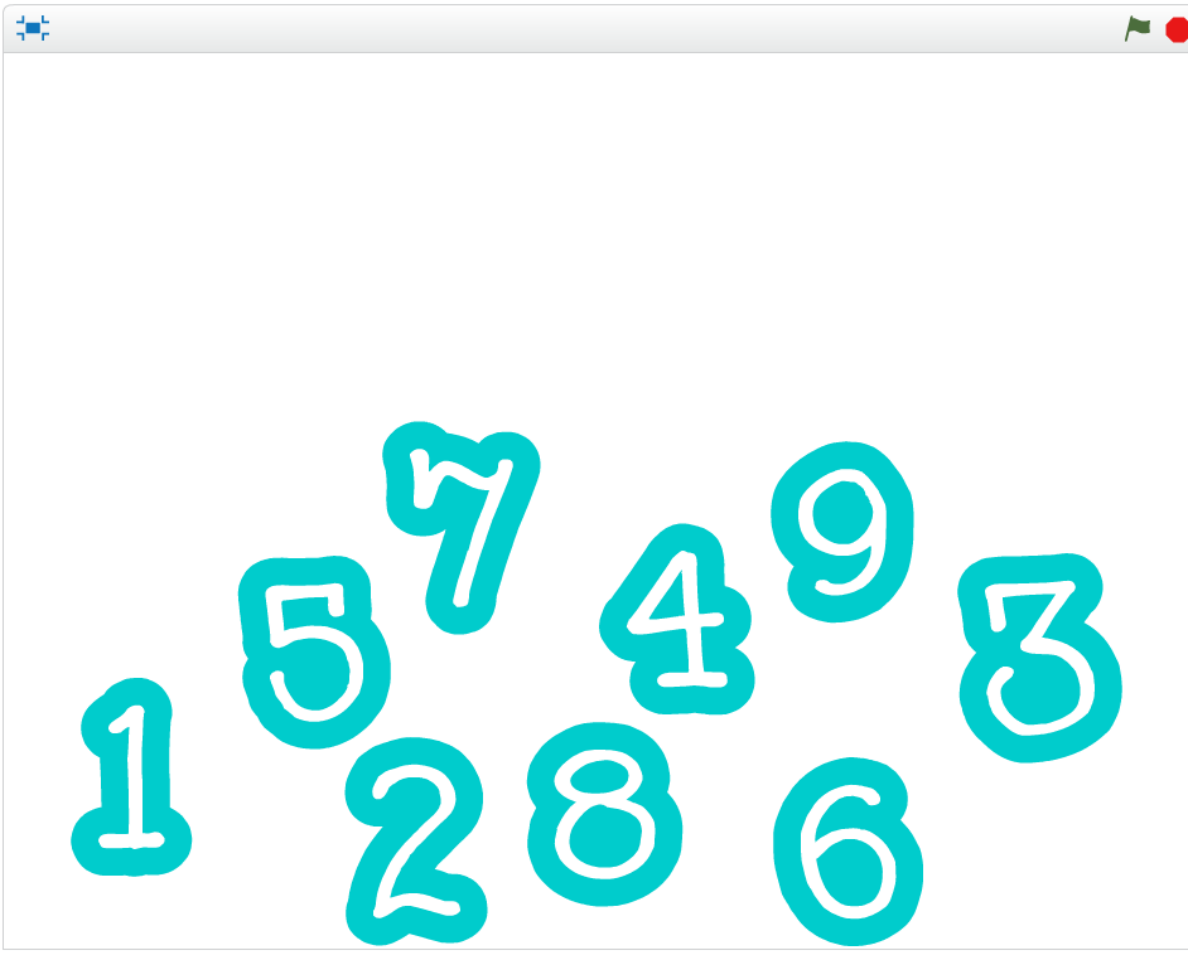


Buyurun, oynayın. Ən yaxşı tennisçini tapın.

8. TƏK-CÜT

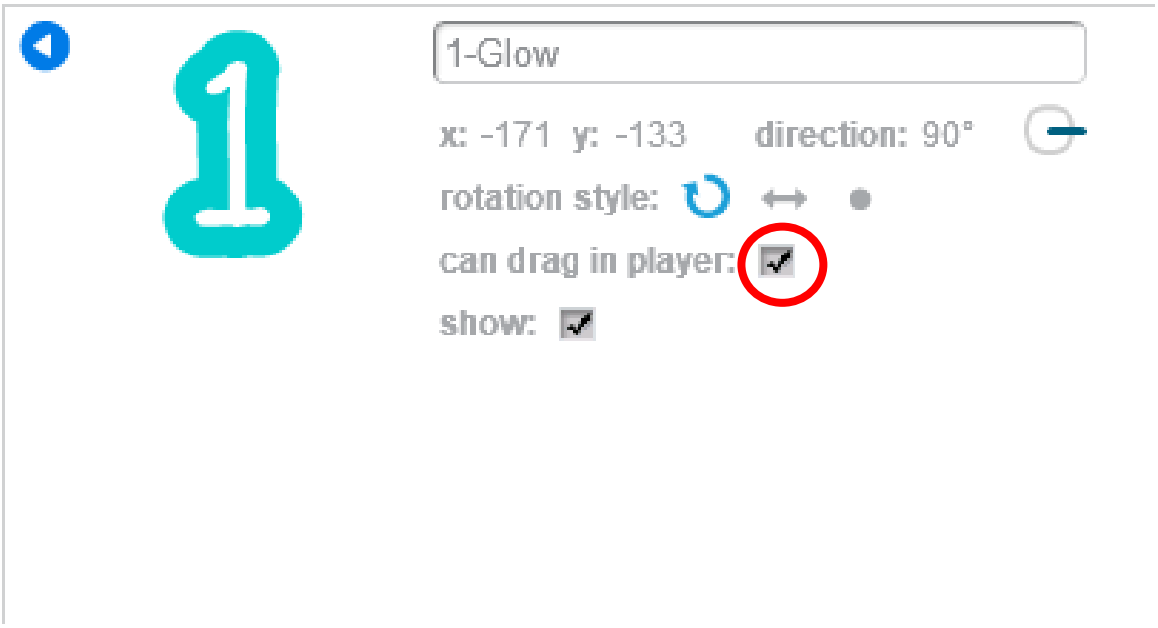
Birinci siniflərdə riyaziyyat dərsində çox vaxt şagirdlərə tək və cüt ədədləri ayırmaq üçün tapşırıqlar verirlər. Gəlin biz də tək və cüt ədədləri tapmaq üçün proqram yazaq.

Boş fayl yaradırıq. Pişik spraytını silirik. Scratch proqramının spraytlar kitabxanasının **Letter** qrupundan **1-Glow, 2-Glow, 3-Glow, 4-Glow, 5-Glow, 6-Glow, 7-Glow, 8-Glow** və **9-Glow** adlı spraytları yükləyirik. Bu spraytları səhifənin aşağısında ixtiyari yerləşdiririk.



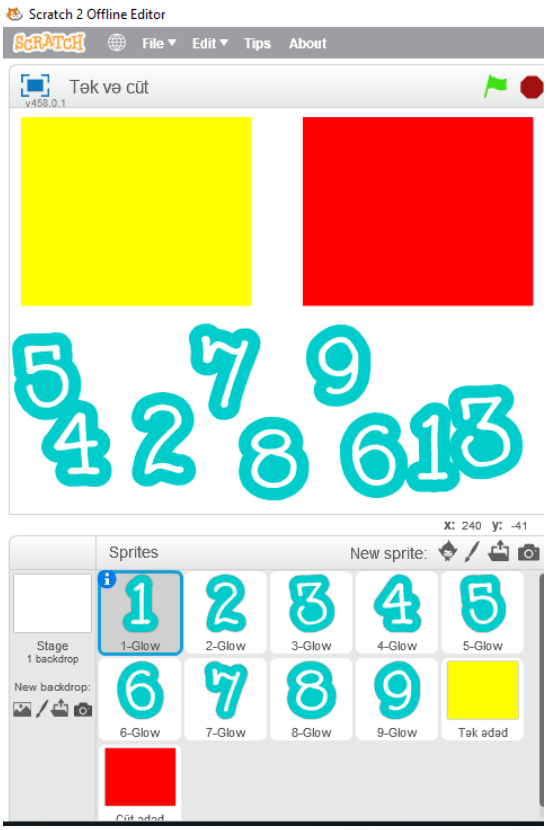
Bundan sonra bu 9 spraytın hamısı üçün aşağıdakı prosedurları yerinə yetirin:

- Spraytı seçin.
- Solda aşağıda spraytlar sahəsində spraytın adının solunda mavi dairənin içində olan i düyməsini sıxın.
- Açılan pəncərədə **can drag in player** (**iş zamanı daşıya bilmək**) sətrinin sağındakı kvadratı qeyd edin.
- Solda yuxarıda mavi dairənin içində olan sola baxan üçbucaq düyməsini sıxın.



Bu əməliyyatlar ona görədir ki, tam ekran rejimində mouse qurğusunun köməyilə personajları daşımaq olsun.

İndi tək və cüt ədədləri bir yerə yığmaq üçün müxtəlif rəngli iki düzbucaqlı formalı sprayt yaradaq. Bunun üçün aşağıda solda spraytlar sahəsində fırça düyməsini sıxıb daxili qrafik redaktora keçirik və dediyimiz düzbucaqlıları çəkirik.

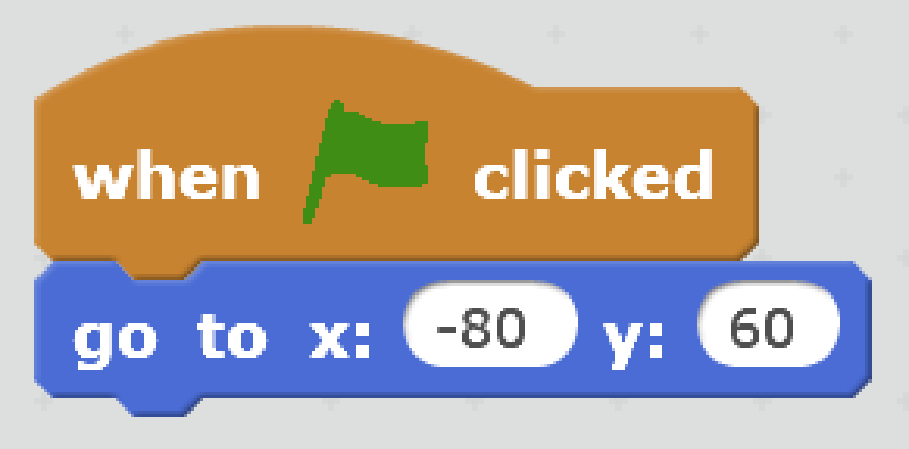


Sarı rəngli düzbucaqlı tək ədədlər üçün, qırmızı rəngli düzbucaqlı cüt ədədlər üçün nəzərdə tutulsun.

Sağdakı şəkil təxmini həmin görüntünü verir.

Artıq bizə lazım olan əsas personajlar hazırdır və onlar üçün proqram yazmaq olar.

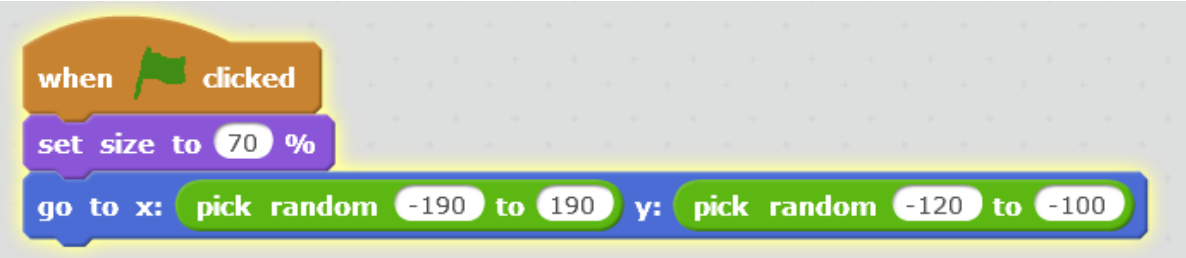
Əvvəlcə sarı və qırmızı düzbucaqlıları səhnədə konkret yerlərdə yerləşdirək. Bunun üçün sarı düzbucaqlıya aşağıdakı əmrləri verək:



Eyni qaydada qırmızı düzbucaqlı üçün də aşağıdakı əmrləri verək:

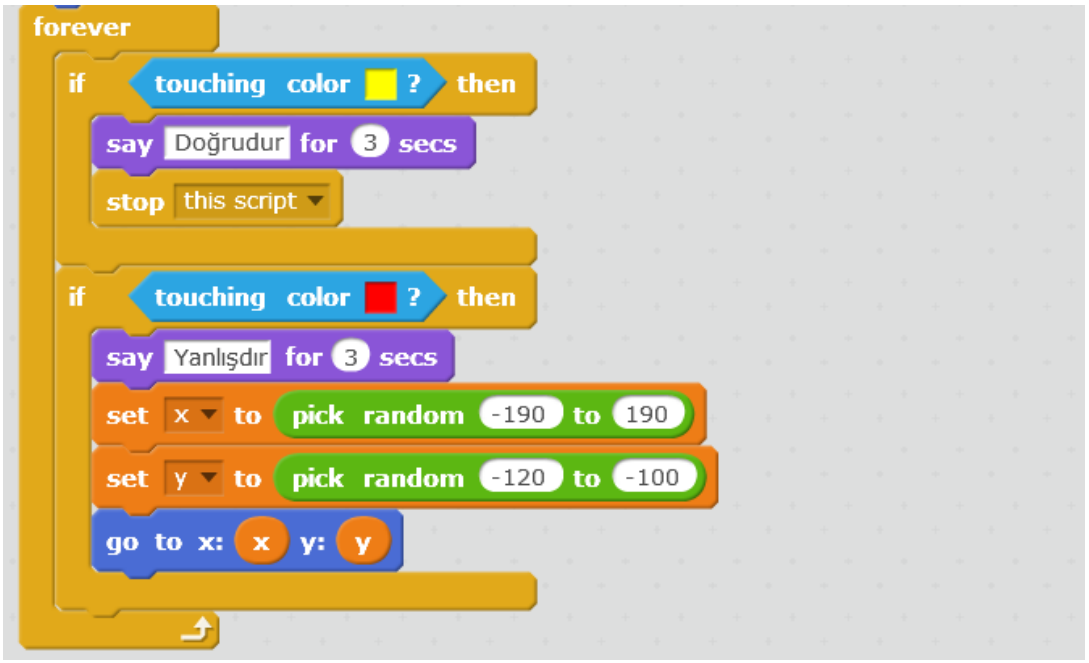
Əmrlərdən görürük ki, sarı düzbucaqlının mərkəzi səhnənin (-80;60) koordinatlı nöqtəsinə, qırmızı düzbucaqlının mərkəzi isə səhnənin (170;60) koordinatlı nöqtəsinə yerləşdirilir.

İkinci addım kimi rəqəmləri göstərən spraytların ölçüsünü nisbətən kiçildək. Təxminən orijal ölçünün 70%-ni götürsək kifayət edər. Dediyimizi tək ədədlərdən olan "**1**" rəqəmini göstərən **1 Glow** adlı sprayt üzərində göstərək. Bu sprayt üçün əmrləri Yaşıl bayraq düyməsini sıxmaqla başladacağıq. Daha sonra ölçünün 70%-ni götürəcəyik. Gəlin bura bir əmr də əlavə edək. Bu əmr spraytı **x** koordinatı -190 ilə 190 arasında, **y** koordinatı isə -120 ilə -100 arasında dəyişən səhnənin ixtiyarı nöqtəsində yerləşdirək. Əmrlər aşağıdakı kimi olacaq:



Üçüncü əmrdən göründüyü kimi **x** və **y** koordinatları üçün qiymət təsadüfi ədəd seçən **pick random___to___** əmri vasitəsilə tapılır.

Gəlin ssenarimizi belə quraq. Əgər daşıdığımız personaj tək ədəddirsə və o sarı düzbucaqlıya daşınıbsa, onda ekranda 3 saniyə müddətində "**Doğrudur**" sözü çıxsın və personaj daşındığı yerdə qalsın. Yox əgər daşıdığımız personaj tək ədəddirsə və o qırmızı düzbucaqlıya daşınıbsa, onda ekranda 3 saniyə müddətində "**Yanlışdır**" sözü çıxsın və personaj səhnənin aşağısında ixtiyarı nöqtədə yerləşsin. Bunun üçün aşağıdakı əmrləri proqrama əlavə etmək lazımdır:



Göründüyü kimi **forever** əmrinin daxilində 2 **if** əmri var.

Bunlardan birincisi ədəd sarı rəngə (tək ədədlər üçün rəng) toxunanda ekranda "**Doğrudur**" sözü çap edilir.

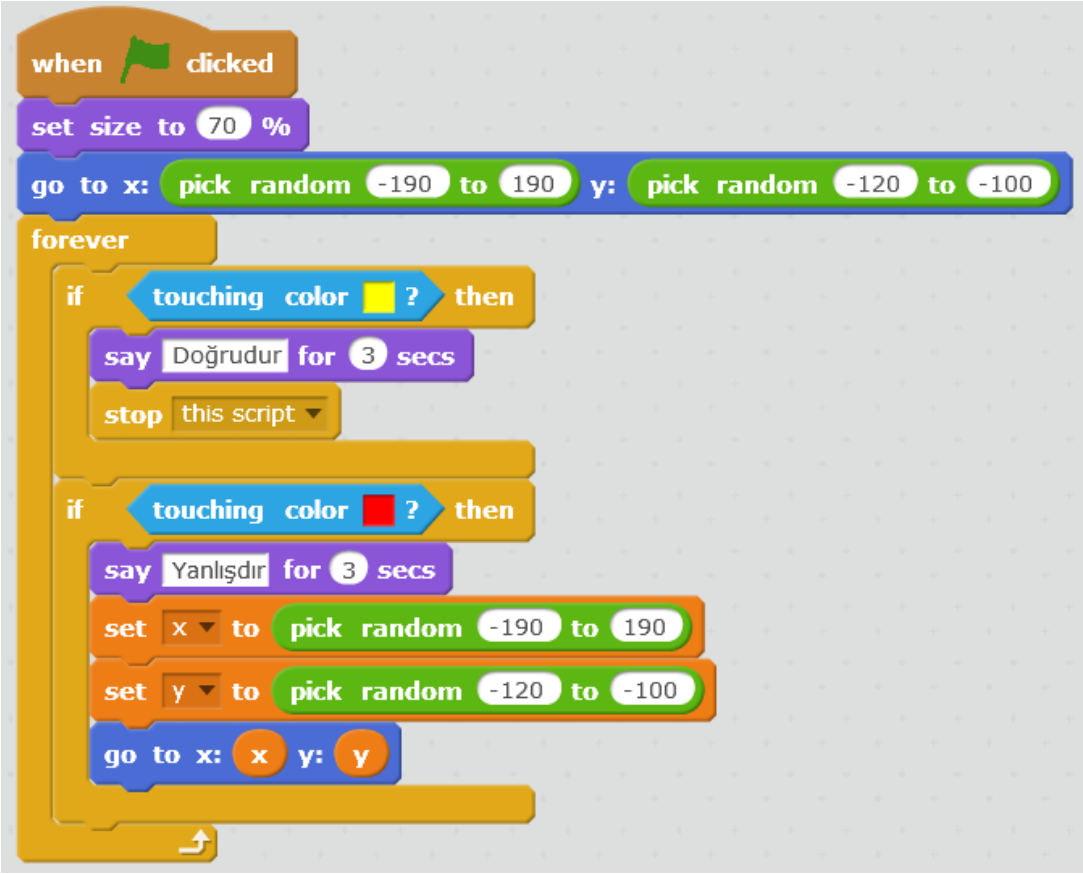
Əmrlərə diqqət eləsiniz görərsiniz ki, burada "**Doğrudur**" sözü çap edilən əmrdən sonra aşağıdakı əmr durur.

Tək ədəd sarı sahəyə toxunanda "**Doğrudur**" sözü çap edilməlidir. Çap əmri **forever** sonsuz dövrünün içinə salındığından personaj daim sarı sahənin üzərində olma səbəbindən "**Doğrudur**" sözü sonsuz sayda çap ediləcək. Bu əmr ona görədir ki, "**Doğrudur**" sözü bir dəfə çap ediləndən sonra **if** əmrinin işi durdurulsun.

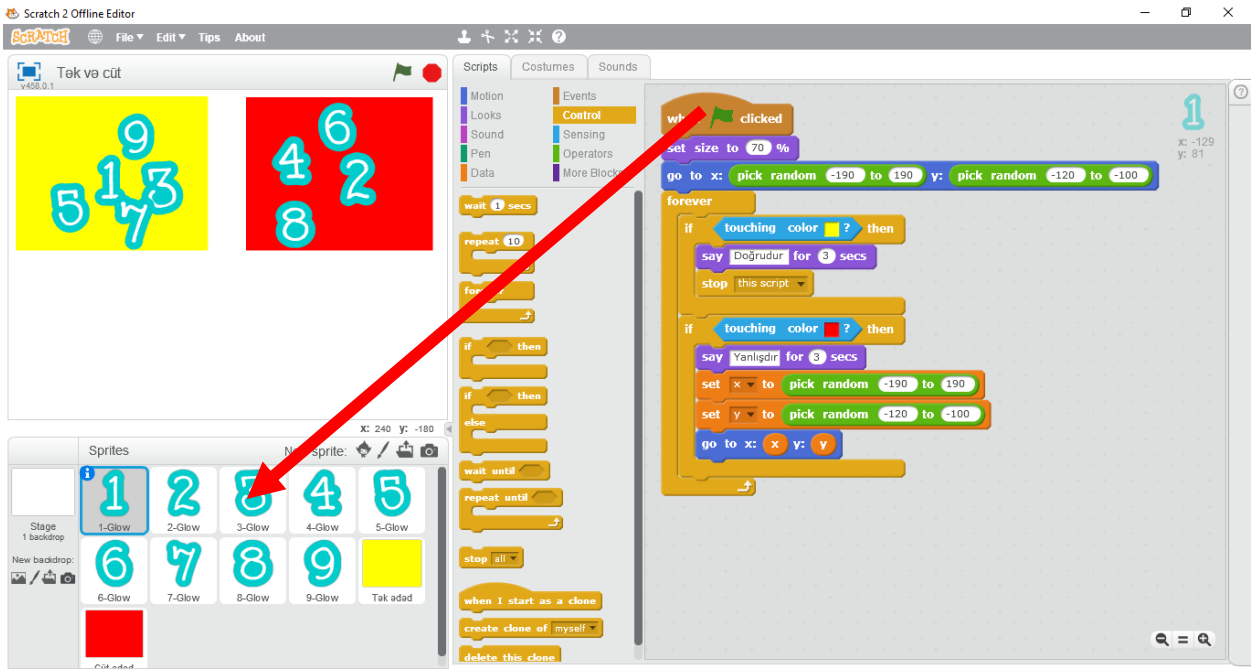
İkinci **if** əmrinə daxil edilən əmrlərin sayı bir qədər çoxdur.

Ədəd qırmızı rəngə (tək ədədlər üçün rəng) toxunanda ekranda "**Yanlışdır**" sözü çap edilir. Daha sonra **x** və **y** dəyişənləri üçün ixtiyarı qiymətlər müəyyənləşir və personaj həmin koordinata yerləşdirilir.

Bu əmrlər proqrama əlavə edildikdən sonra tək ədədləri göstərən personaj üçün əmrlər bloku belə olacaq:

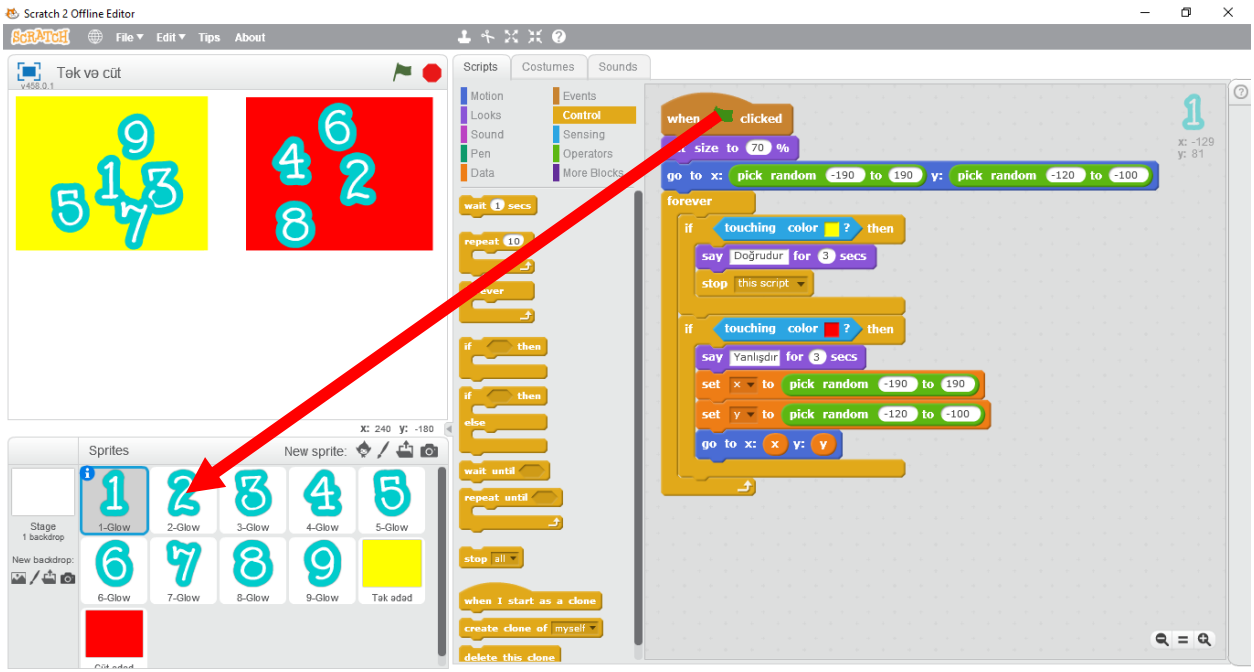


Prinsipcə bütün tək ədədlərə uyğun personajlar üçün proqramlar eyni olmalıdır. Ona görə də biz “**1**” ədədinə uyğun personaj üçün yazılmış proqramı “**3**”, “**5**”, “**7**” və “**9**” ədədlərinə uyğun personajlara köçürə bilərik. Bunun üçün “**1**” ədədinə uyğun personaj üçün yazılmış proqramı açırıq. Mouse qurğusu ilə onun birinci əmrini “**3**”, ədədinə uyğun personajın üzərinə gətirib bir neçə saniyə gözləyib buraxırıq. İndi “**1**” ədədinə uyğun personajı seçin və dediklərimizə əmin olun. Dediklərimizi “**5**”, “**7**” və “**9**” ədədlərinə uyğun personajlar üçün də edin.



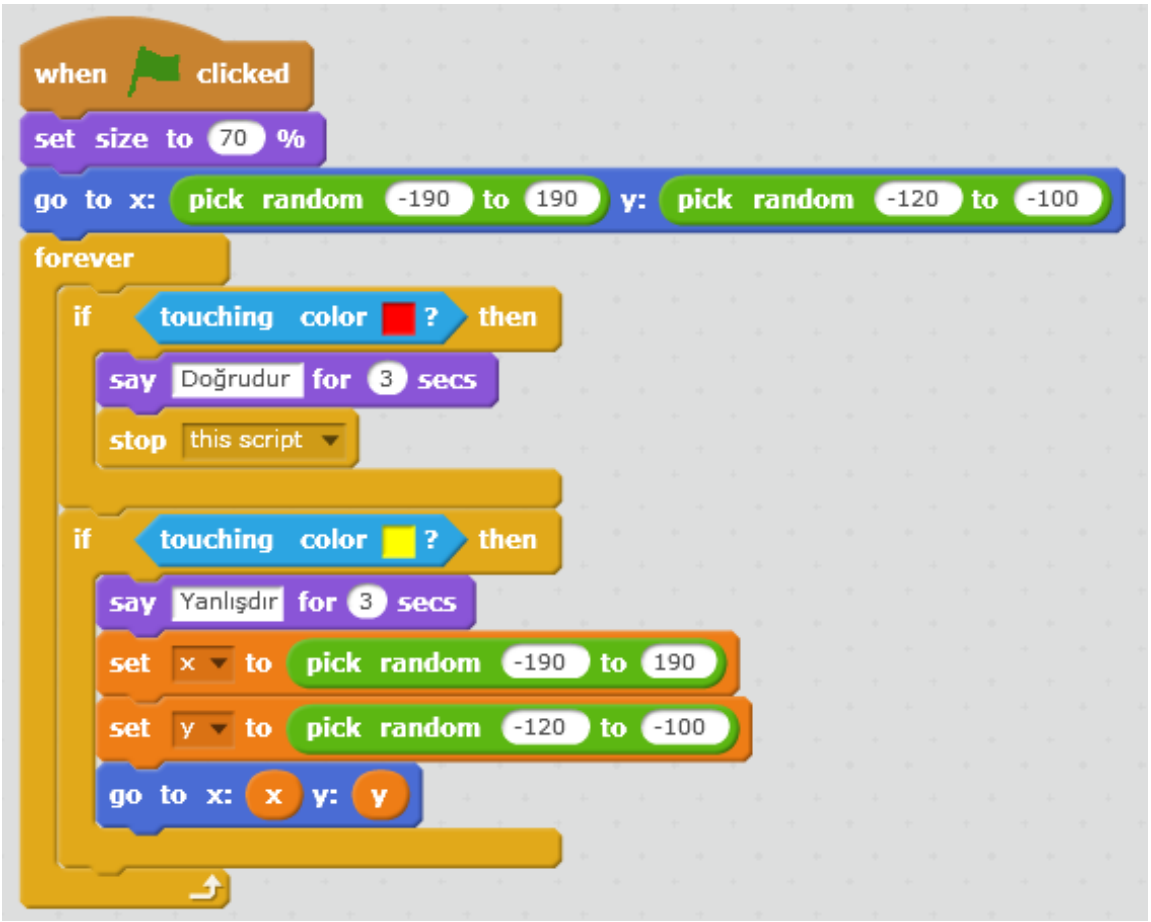
İndi isə cüt ədədlər üçün uyğun proqramı yazaq. Prinsipcə proqram burada da eyni olmalıdır. Sadəcə olaraq birinci və ikinci **if** əmrində verilmiş rəngləri uyğun olaraq sarıdan qırmızıya və qırmızıdan sarıya dəyişirik.

Bunun üçün biz “**1**” ədədinə uyğun personaj üçün yazılmış proqramı “**2**” ədədinə uyğun personaja köçürürük.



Daha sonra rəngləri dəyişirik (sarını qırmızıya və qırmızını sarıya).

Nəticədə aşağıdakı əmrlər blokunu alırıq:

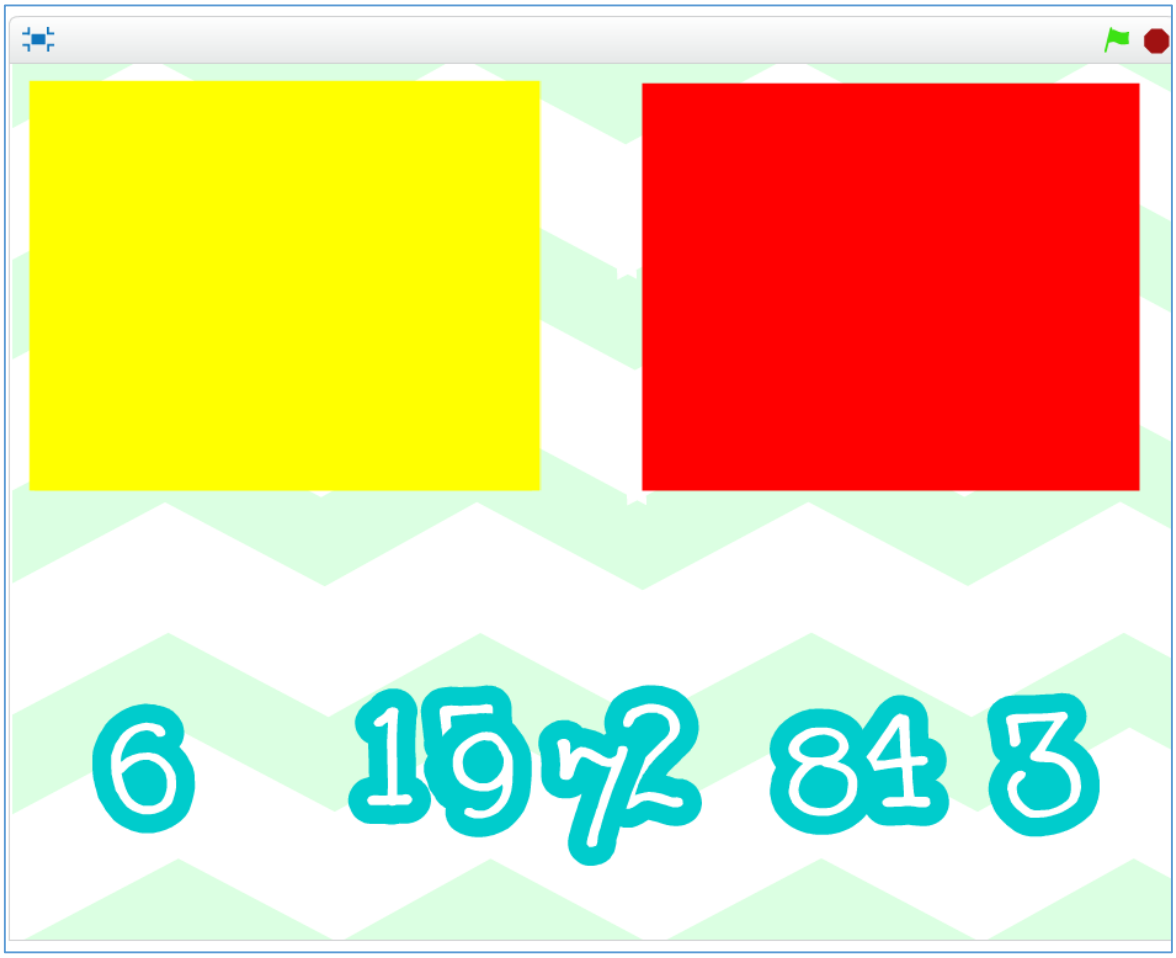


İndi isə “**2**” ədədinə uyğun personajı seçin və onun əmrlərini “**4**”, “**6**” və “**8**” ədədlərinə uyğun personajlara köçürün.

Artıq proqramımız iş üçün hazırdır. Tam ekran rejiminə keçin və **Yaşıl bayraq** düyməsini sıxıb proqramı başladın.

Amma görürük ki, baş verən hadisələr ağ fonda o qədər də gözəl görünmür. Gəlin proqramımıza gözəgəlimli fon verək. Bunun üçün səhnəni seçin. Onun üçün fonlar kitabxanasından **Holiday** qrupundan **stripes** fonunu seçin. Əlbəttə ki, Siz başqa bir fonu və ya İnternetdən yüklədiyiniz şəkli səhnəyə qoya bilərsiniz.

Bizim halda səhnənin görünüşü təxminən belə ola bilər:



İndi isə **TƏK-CÜT** öyrədici oyununu oynaya bilərsiniz.

9. SPİNNER

XXI əsrin ikinci onilliyinin ən dəbdə olan oyuncaqlarından biri də **Spinner** adlanan oyuncaqdır. İstehsalıçılar bu oyuncağın insanın psixologiyasına müsbət təsir etdiyini bəyan edirlər. Uşaqlar və yeniyetmələr arasında böyük populyarlıq qazanan bu oyun üç və daha çox ləçəkli bir fiqurdur. Fiqurun ortasında xırda təkərcik var. Həmin fıqur bu təkərcik ətrafında fırlanır. İstehsalçıların dediyinə görə əllə bu fiquru fırlatdıqda fırlanma hadisəsi insanın əsəblərini sakitləşdirir.

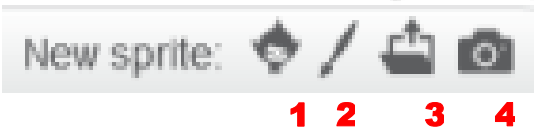
Əlbəttə ki, bu fiqurun psixologiyaya necə təsir etməsi bizi maraqlandlrmır. Sadəcə olaraq gəlin birlikdə bu oyuncağın kompüter variantı (modeli) üçün proqram quraq.

İşimizi pişik spraytını silməklə başlayaq.

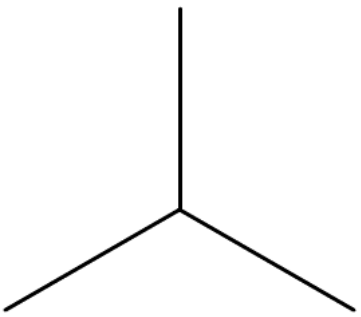
Əvvəlcə bizə spinnerin şəkli lazımdır ki, onun əsasında sprayt yaradaq. Biz bunu özümüz də yarada bilər, ya da internetdən 4-5 variantdan **gif** formatda fonu olmayan spinner şəkli tapıb onu kömpüterimizə yükləyə bilərik. Əgər fonsuz spinner şəkli tapmasanız, onda yüklədiyimiz şəkillərin fonunu təmizləmək olar.

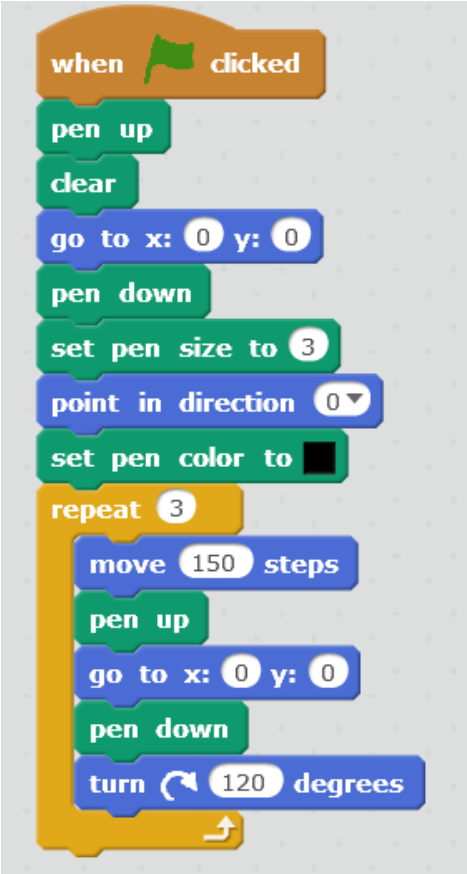
Gəlin biz Scratch 2.0 proqramının qrafik redaktorundan istifadə edərək öz spinnerimizi yaradaq. Əvvəlcə üçləçəkli spinner yaradaq.

İnterfeysin **4** nömrəli sahəsinin sağ hissəsində Yuxarı sağ küncdə **New sprite:** bölməsindəki yeni sprayt əlavə etmək üçün ikinci düyməni sıxıb Scratch proqramının daxili Paint proqramına keçək. Daxili Paint proqramı ilə iş bu vəsaitə əlavədə təsvir edilib.



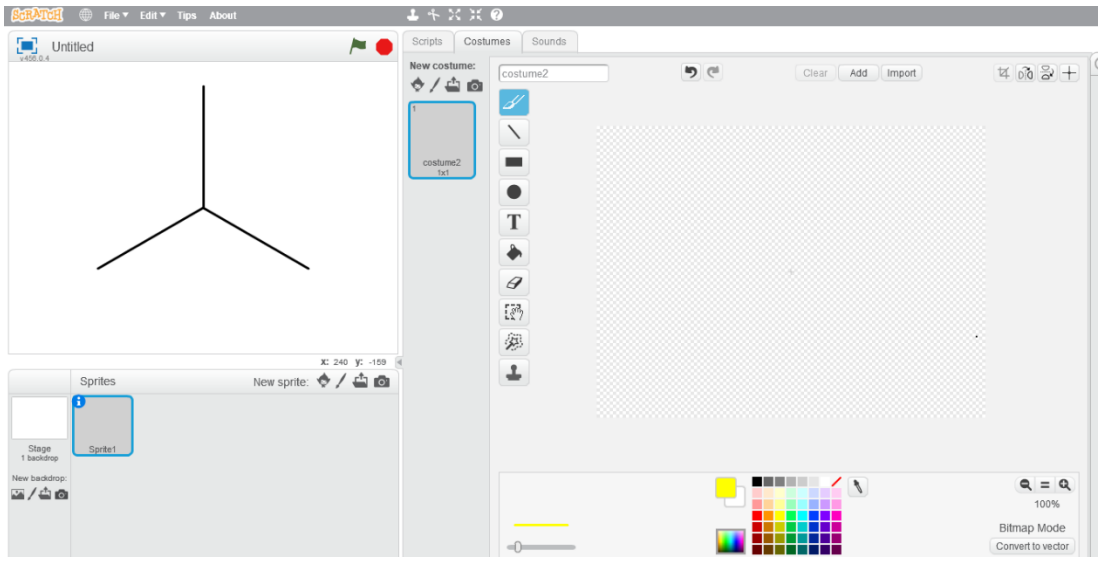
Ləçəklər arasındakı məsafənin bərabər olması üçün belə bir üsuldan istifadə etmək olar. Əvvəlki addımda boş sprayt yaratdıqdan sonra əsas interfeysdə **Scripts** menyusunu seçin. Bu sprayt üçün aşağıdakı proqram kodunu başladanda səhnədə sağda gördüyünüz fiqur alınacaq.



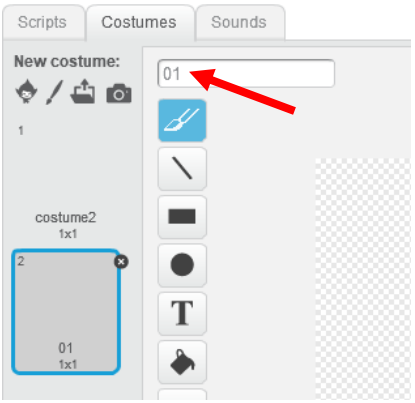


Gördüyünüz kimi proqram kodu üçləçəkli spinner üçündür. Dörd və daha çox ləçəkli spinner üçün proqram kodunda **repeat** əmrində ləçəklərin sayı göstərilməlidir. **turn ↻ ___ degress** əmrində isə dönmə bucağı **360/ləçəklərin sayı** düsturu ilə hesablanacaq. Məsələn: dördləçəkli spinner üçün düstur belə olacaq: 360/4=90.

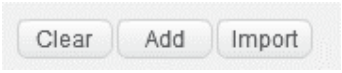
Yenidən əsas interfeysdə **Costumes** menyusu seçib Paint proqramına qayıdaq. Ekranda aşağıdakı görünüş olacaq:



Daxili Paint proqramının **New costumes** bölməsində fırça işarəsi olan düyməni sıxdıqda biz yeni boş kostyum alacağıq. Qırmızı oxla göstərilən hissədən kostyumun adını dəyişib **01** qoyun. Yəqin bu bizim **01** adlı kostyumumuzdur. Növbəti yaradacağımız kostyumlar üçün də oxşar qaydada ad qoyun.

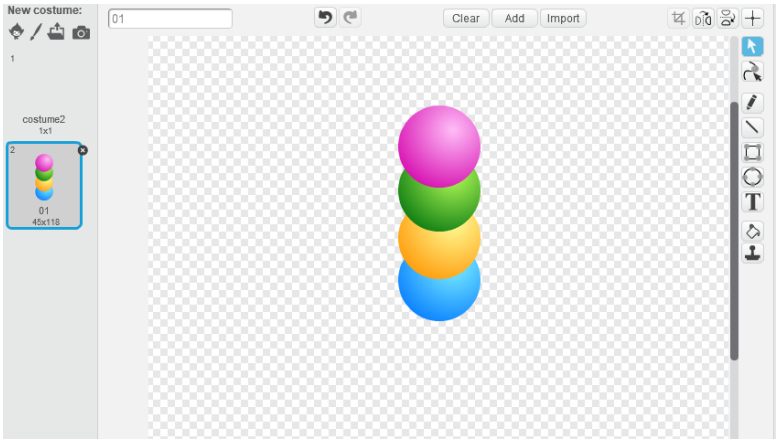


Daha sonra daxili Paint proqramının yuxarı sətrindəki düymələrdən

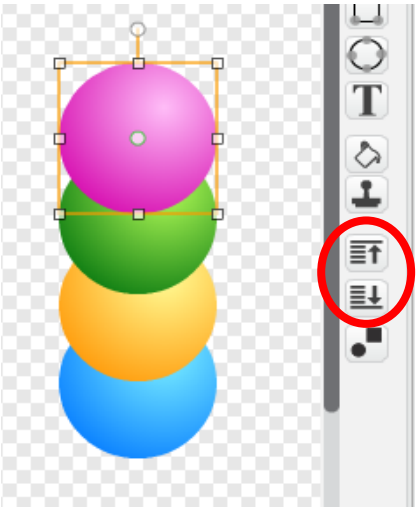


Add düyməsini sıxın və açılan kostyumlar kitabxanasından **Things** qrupundan növbə ilə **ball-a**, **ball-b**, **ball-c**, **ball-d** kostyumlarını seçin.

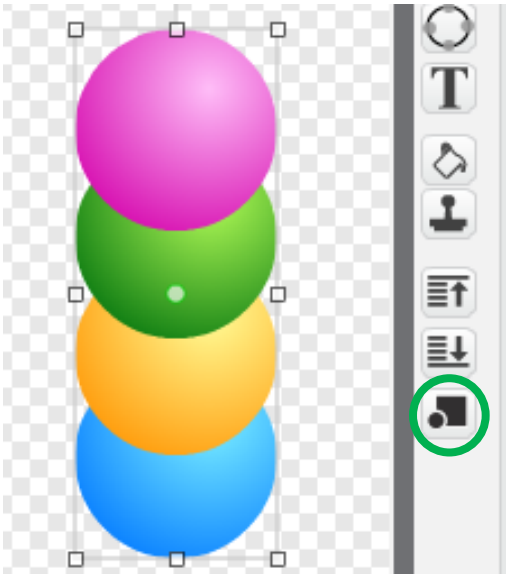
Həmin fiqurları aşağıda gördüyünüz formada düzün.



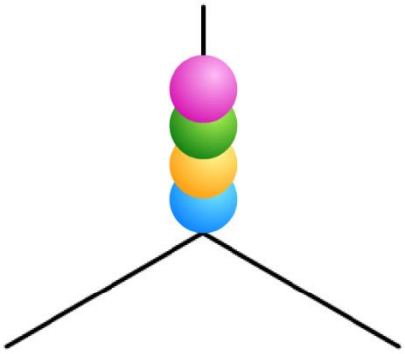
Kürəcikləri laylara görə düzmək üçün fiquru seçib sağ tərəfdə alınan 2 düymədən (həmin düymələr qırmızı halqaya alınıb) birini seçməklə fiqurun yerini üst və ya alt qata dəyişmək olar.

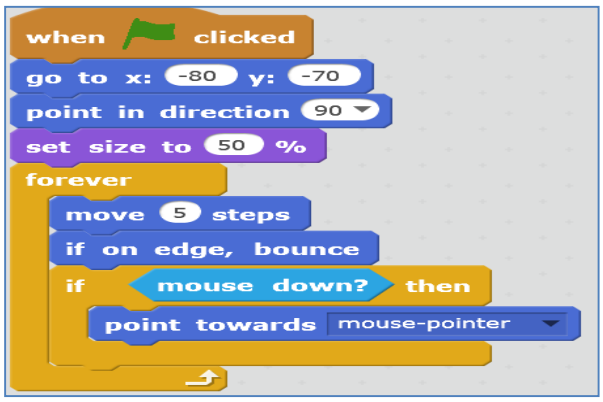


Növbəti addımda dörd fiquru birgə seçirik və yaşıl halqaya alınmış düyməni sıxmaqla onları qrulaşdırırıq.

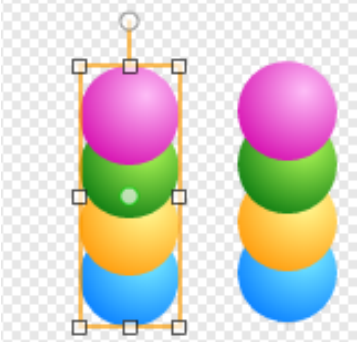


Birləşmiş fiquru mouse qurğusu və oxlarla elə hərəkət etdiririk ki, səhnədə aşağıdakı şəkildə gördüyünüz vəziyyət alınsın.



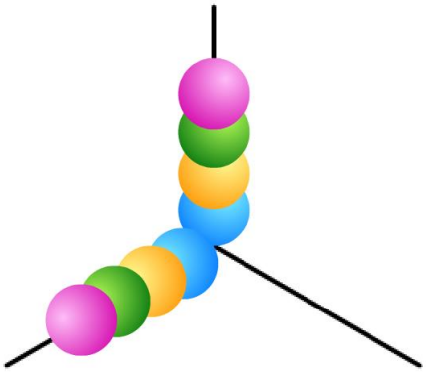


Alınmış yeni fiquru seçib onun surətini (dublikatını) çıxardırıq. Bunun üçün Scratch proqramının interfeysində **2** nömrəli sahədən **1** nömrəli **(Dublicate)** aləti seçib kursoru surətini çıxarmaq istədiyimiz fiqurun üzərinə gətirməli və onun üzərinə sıxmalıyıq. Əməliyyatdan sonra həmin fiqurun dublikatı alınacaq.

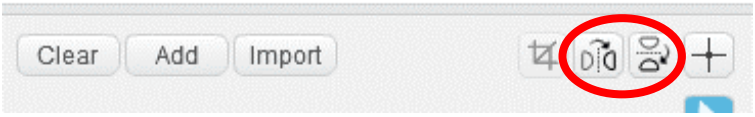


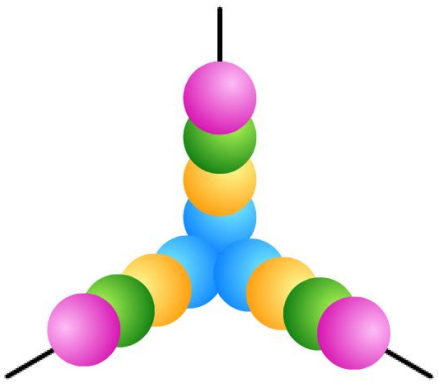
Seçilmiş fiqurun ətrafında 8 ədəd xırda kvadrat və yuxarısında isə xırda dairə əmələ gəlir. Kvadratlar ölçünü dəyişmək, dairə isə fırlatmaq üçündür.

Fırlanma əməli və klavişlər vasitəsilə ikinci fiquru şəkildə gördüyüniz vəziyyətə gətiririk.

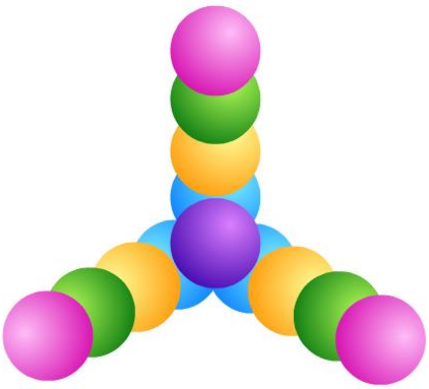


Üçüncü ləçəyi yaratmaq üçün ya indicə dediyimiz üsuldan istifadə edirik, ya da ikinci fiqurun (ləçəyin) dublikatını alırıq, Paint proqramının aşağıda görünən hissəsindən qırmızı haşiyəyə alınan hissədən üçüncü fiquru (ləçəyi) üfiqi istiqamətdə dəyişirik və klavişlərlə idarə edib aşağıdakı görüntünü alırıq.

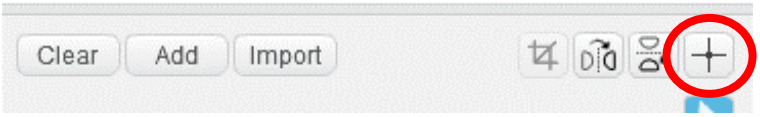




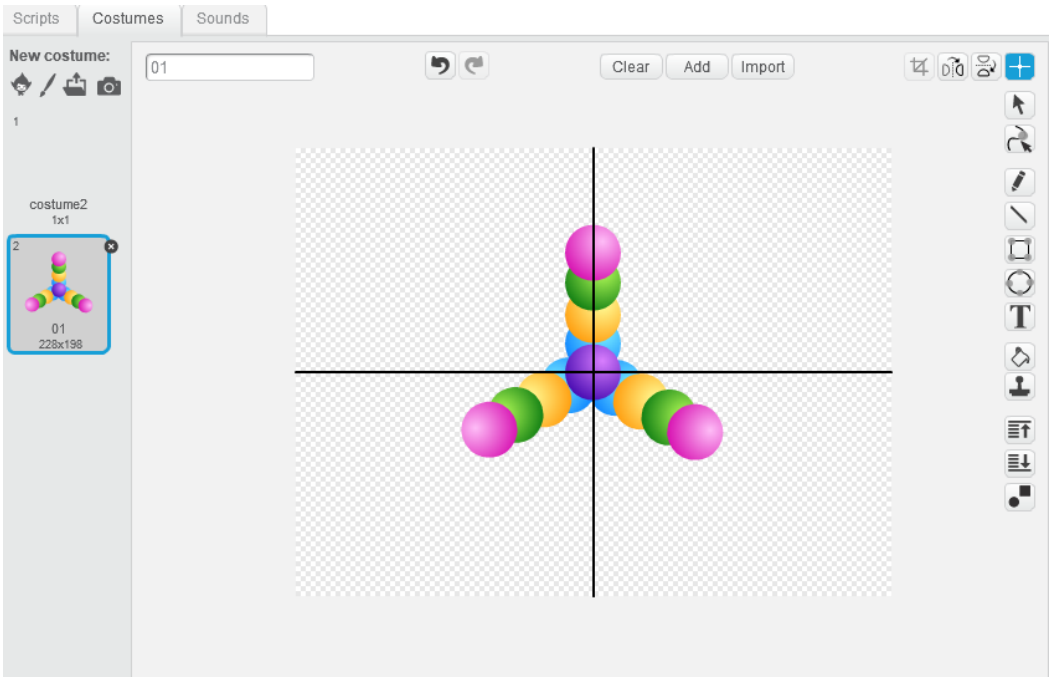
Add düyməsini sıxın və açılan kostyumlar kitabxanasından **Things** qrupundan daha bir **ball-e** kostyumunu seçin və o kostyumu qurduğumuz fiqurun mərkəzinə qoyun. Fiquru tam seçin və onu qruplaşdırın. Nəticədə gördüyünüz fiqur alınacaq:



Bu fiqurla bağlı daha bir işimiz qaldı. O da fiqurun mərkəzini seçmək və onu səhnənin mərkəzinə gətirmək. Bunun üçün daxili Paint proqramının yuxarı sağ küncündə olan haşiyəyə alınmış **kostyumun mərkəzini seçin** düyməsini sıxın.

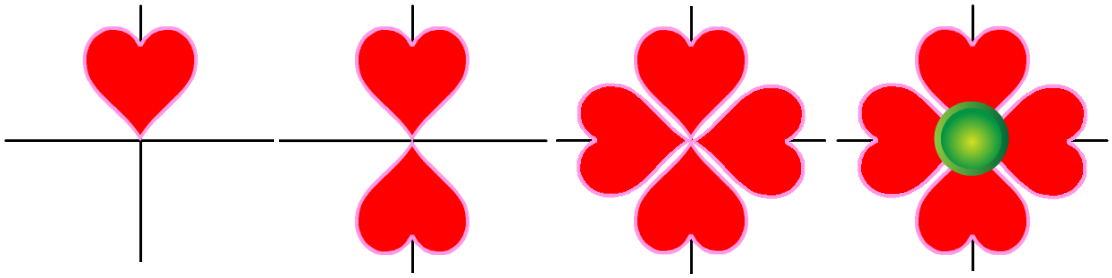


Alınmış nişanı şəkildə göründüyü kimi fiqurun mərkəzinə gətirin və mouse qurğusunun sol düyməsini buraxın.

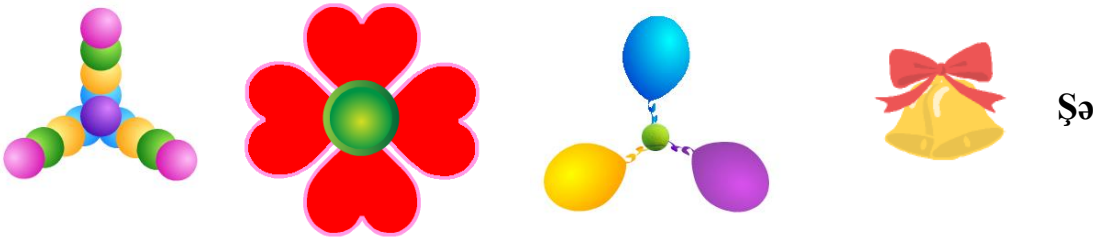


Nəticədə fiqurumuzun mərkəzi səhnənin mərkəzində duracaq.
Yuxarıda dediyimiz qaydada daha üç ədəd spinner yaradın. Dördləçəkli spinner yaradanda bunu 3 addıma etmək olar. Əvvəlcə bir şaquli (və ya üfüqi) ləçək yaradılır. İkinci addımda onun dublikatı alınır və şaquli (üfüqi) istiqamətdə döndərilir. Ləçək lazım olan yerə sürüşdürülür. İki ləçək birgə seçilib qruplaşdırılır. Üçüncü addımda birləşmiş fiqurun dulikatı çıxarılır və dublikat fırlanma aləti ilə 90 dərəcə fırladılır və yeri mərkəzə dəyişdirilir. Tam fiqur seçilir və qruplaşdırılır.

Sonda fiqurun mərkəzinə yenə bir dairə qoyulur və yenə birləşdirmə aparılır. Ardıcıllıq aşağıda verilib.

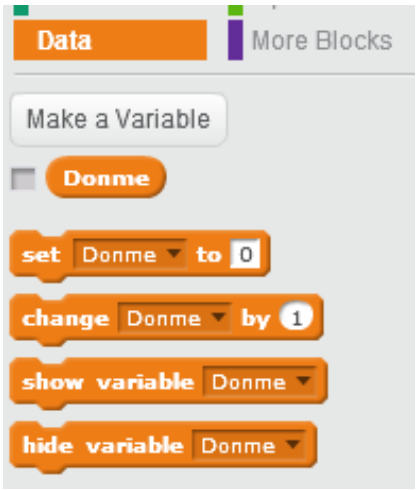


Fiquru səhnənin mərkəzinə qoymaq prosesi təkrar edilir.
Sonda spraytların adının dəyişilməsini bir də yoxlayın və boş olan birinci kostyumu silin. **Scripts** menyusuna qayıdın. Ləçəkləri düzgün çəkmək üçün istifadə etdiyimiz xətləri çəkən proqram kodunu silin.
Bizim proqram aşağıdakı kostyumlarla işləyəcək:



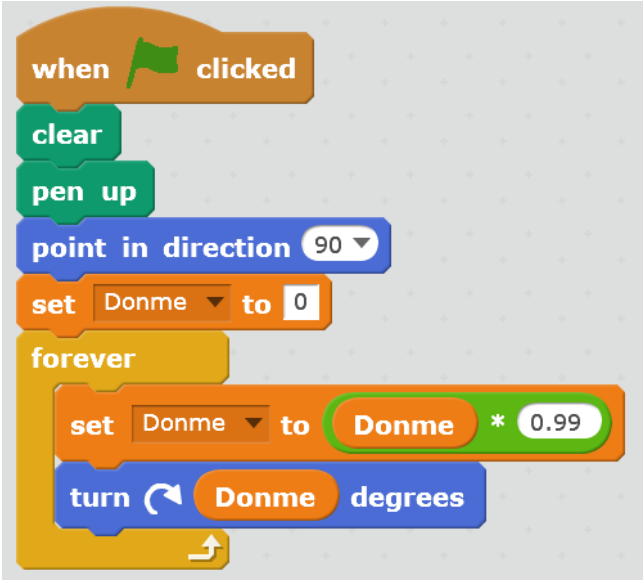
Şə

Kostyumların adı uyğun olaraq **01, 02, 03 və 04** olmalıdır.



Artıq biz proqramı yaza bilərik. Əvvəlcə **Data** qrpunda **Donme** adlı dəyişən yaradaq. Bu dəyişən fırlanma zamanı dönmə bucağını idarə etmək üçündür.

Spinnerin fırlanmasını təmin edən proqram kodu belə olacaq:



when ____ clicked yaşıl bayrağı sıxanda proqramı başladır.
clear ekrandakı mövcud qrafik görüntüləri təmizləyir.
pen qələmi yuxarı qaldırır.
point in direction 90 başlanğıc istiqamət kimi sağı götürür.
set Donme to 0 əmri **Donme** dəyişəninə başlanğıcda sıfır mənimsədir. Bu o deməkdir k, biz spinneri fırlatmamışıq.
Növbəti əmrlər spinnerin fırlanmasını təmin edir.

forever əmri sonsuz dövrü təmin edir.

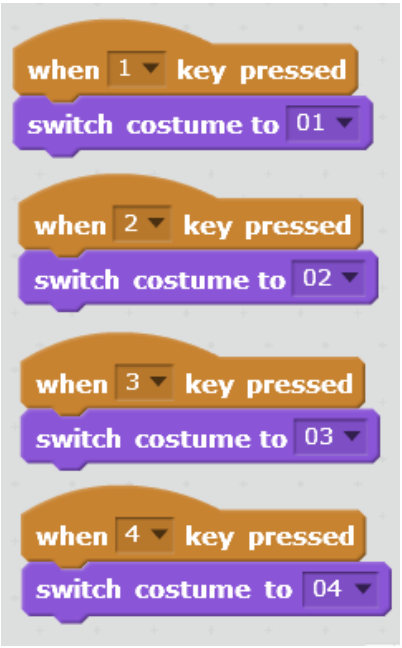
set Donme to Donme *0.99 əmri hər dəfə dönmə bucağının **0.99** hissəsini götürür. Bu ona görədir ki, əgər spinnerin fırlanma sürəti dəyişilmirsə, onda dönmə bucağının qiyməti tədricən azalır, sürət aşağı düşür və sonda spinner sükunət halına gəlir.

turn ↻ Donme degress əmri fırlanmanı **Donme** dəyişəninin qiyməti qədər təmin edir.
Əgər tam səhnə rejiminə keçib yaşıl bayrağı sıxsaq görəcəyik ki, spinner fırlanmır. Səbəb **Donme** dəyişəninin qiymətinin sıfıra nərabər olmasıdır.

Deməli biz spinnerin fırlanma sürətini və istiqamətini dəyişməliyik. Bunun üçün ← və → oxlarından istifadə edərək sürətin qiymətini və istiqamətini dəyişəcəyik. **Events** qrupunda **when ____key pressed** əmri var. Bu əmri iki variantda yazacağıq.

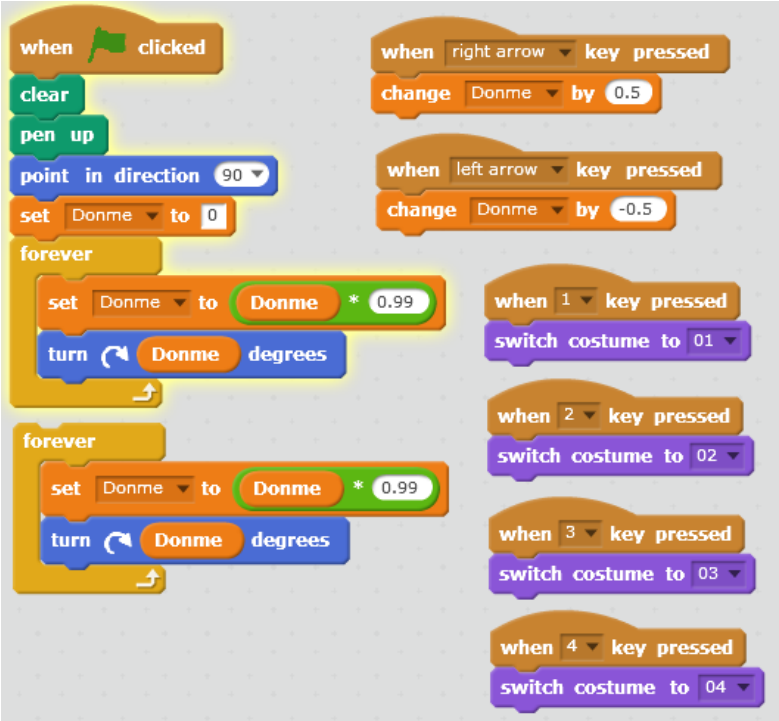
Birinci əmr sağ oxu sıxanda **Donme** dəyişənini 0.5 vahid artırır. İkinci əmr isə sol oxu sıxanda **Donme** dəyişənini 0.5 vahid azaldır (başqa sözlə istiqaməti dəyişir). İndi proqramı başladın və sol, sağ oxları ilə spinnerin hərəkətini idarə edin.

Bəs görəsən fırlanma zamanı spinnerin görünüşünü (kostyumunu) dəyişmək olarmı?

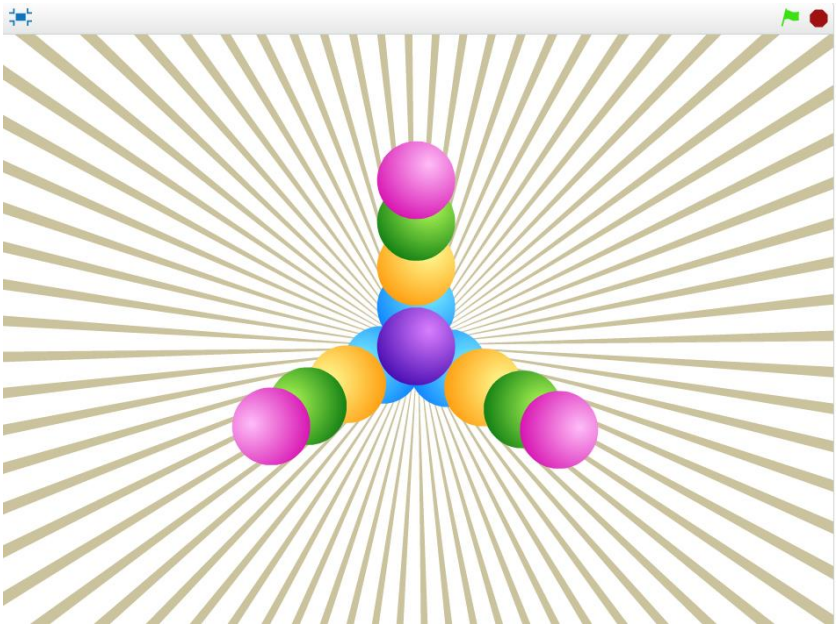


Yenə də **Events** qrupundakı **when ____key pressed** əmrindən istifadə edərək istədiyimizə nail oa bilərik. 1, 2, 3 və 4 klavişlərini sıxmaqla uyğun gələn adlı spinneri göstərə bilərik.

Proqramın son görünüşü belə olacaq:



Tam ekran rejiminə keçin və proqramı başladın. Artıq hər şey istədiyimiz kimi işləyir. Amma oyun darıxdırıcı olmasın deyə maraqlı bir fon seçilsə və musiqi ilə müşayiət olunsa daha maraqlı olardı. Səhnənin fonu üçün proqramın fonlar kitabxanasındakı **Other** qrupundan **rays** adlı fonu seçin. Ağ fonu isə silin. Onda görüntü belə olacaq:

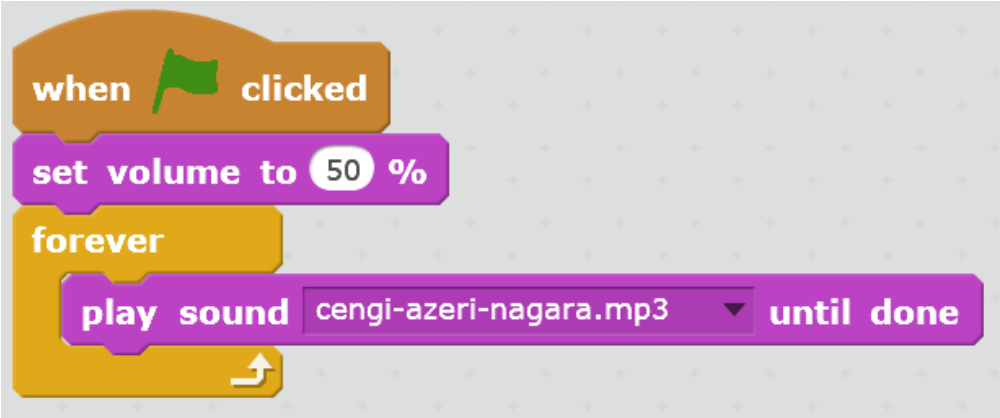


Əlbəttə ki, Siz başqa fon da seçə bilərsiniz. Bunu özünüz yoxlayın.

Spinner oyunu ritmli oyun olduğundan bizə uyğun musiqi seçmək lazımdır. Bu məqsədlə ritmik bir Azərbaycan musiqisi seçək və onu oyun oynanılan zaman oyuna

arxa səs fonu kimi proqrama əlavə edək. Bu məqsədlə nağara ifasında **"Cəngi"** adlı musiqini **mp3** formatda internetdən yükləyək. Daha sonra səhnəni seçək. **Sound** rejiminə keçək. Həmin mahnını əlavə fayl kimi səhnəyə əlavə edək. Faylın adını **cengi-azeri-nagara.mp3** qoyaq.

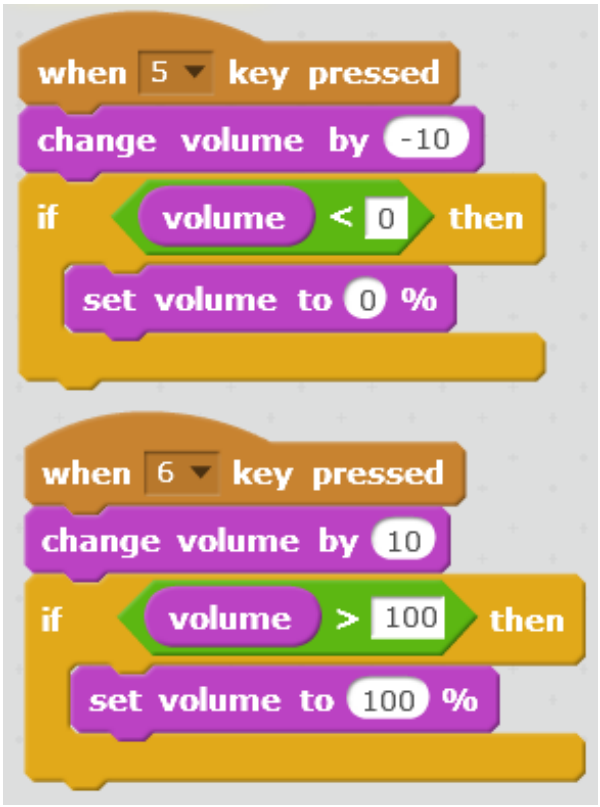
Musiqi proqrama daxil edildikdən sonra səhnə üçün olan aşağıdakı proqramı yazaq:



Görürük ki, səsin ucalığını başlanğıcda 50% götürmüşük.

Artıq proqramı başladanda bu gözəl musiqi səslənir.

Bəs görəsən musiqinin səsini oyunun gedişi zamanı artırıb-azalda bilərikmi? "**Novruz payı**" adlı oyunun proqramında biz artıq bunu etmişik. Bunun üçün yenə də klavişlə idarəetmə əmrindən istifadə etmək olar. Gəlin bu proqramda "5" klavişini səsi azaltma, "6" klavişini səsi ucaltma üçün istifadə edək. Əmrlər aşağıdakı kimi olacaq:



Yuxarıda dediyimiz kimi, başlanğıcda səsin ucalığını 50% götürürük. Oyun prosesində “1” və “2” klavişlərini sıxmaqla səsin ucalığını artırıb-azaldırıq. Amma burada bir halı nəzərə almaq lazımdır ki, səsin ucalığı mənfi ola bilməz. Həmçinin 100%-dən də çox ola bilməz.

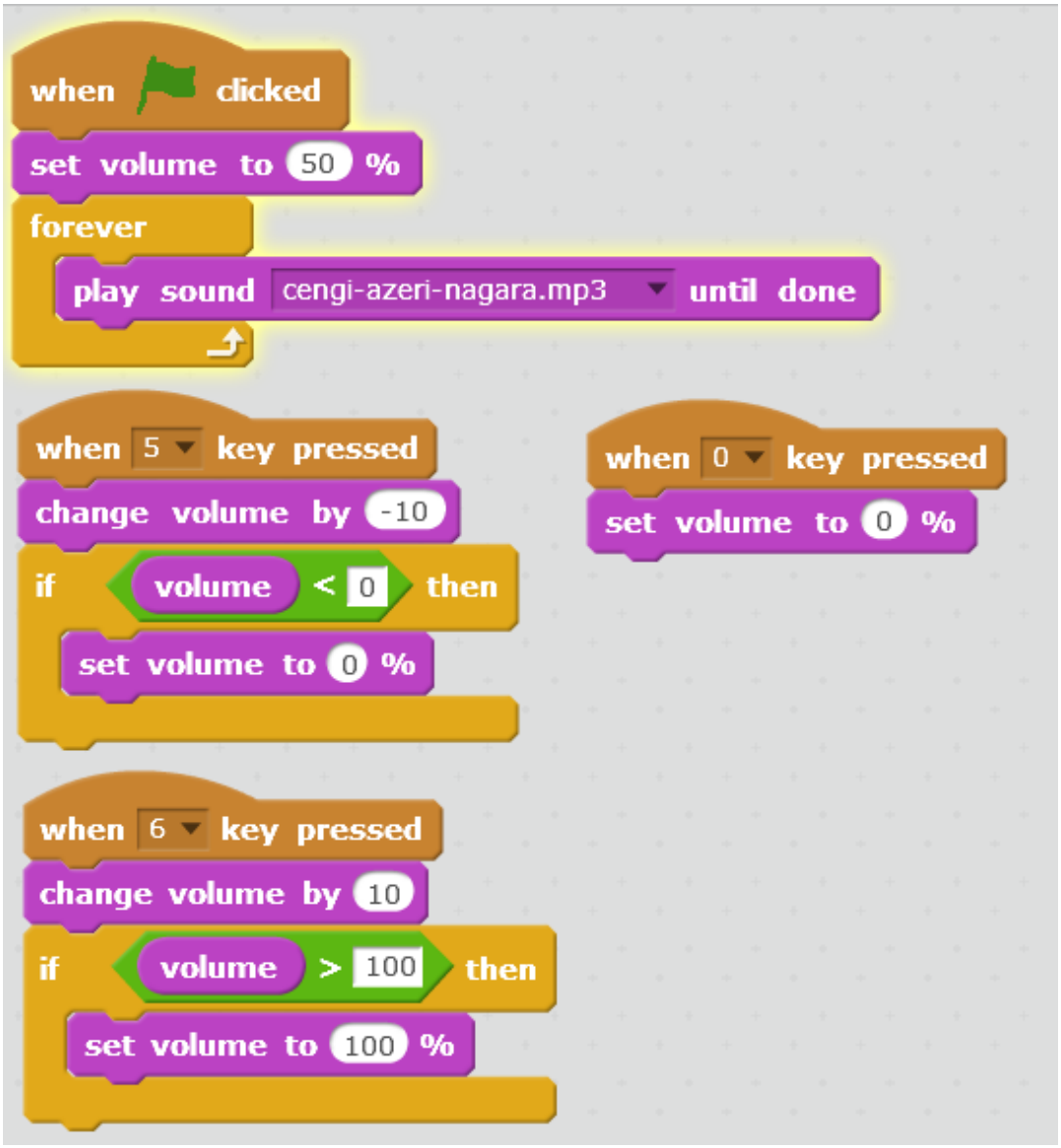
Sonda yenidən tam ekran rejiminə keçin, proqramı başladın, bahar əhval-ruhiyyəsini hətta qışda da yaşayın və oyundan həzz alın. İstədiyiniz vaxt səsin yüksəkliyini dəyişin.

Bu proqramda “3” klavişini sıxdıqda səsin tamamilə kəsilməsi (**mute** rejimi) üçün əmrlər bloku yazın.

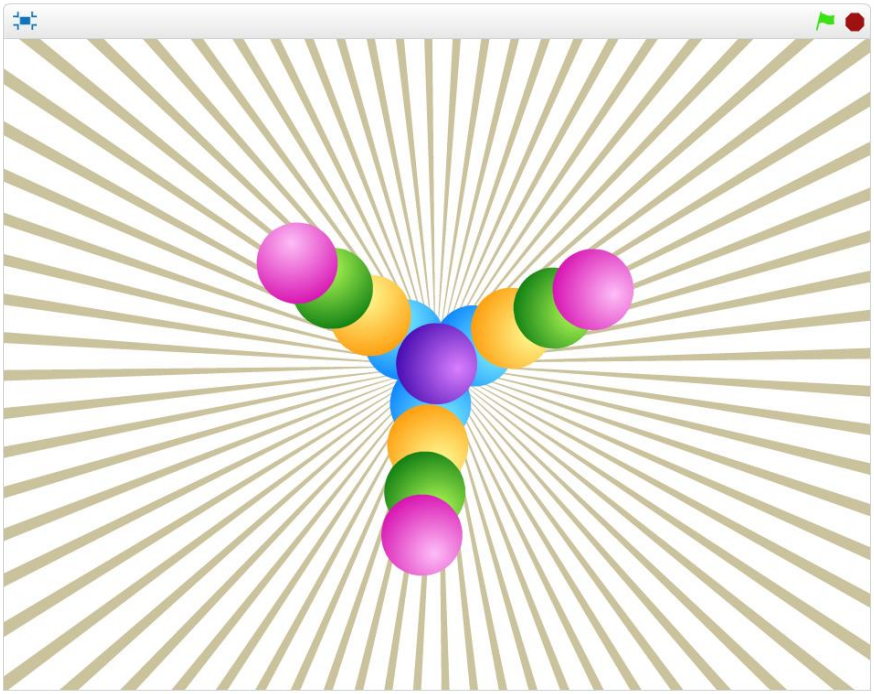
Əgər oyun zamanı səs Sizi bezdirirsə, səsi tamailə kəsmək olar. Bunun üçün “5” klavişini sıxmaqla səsi tədricən azaldıb kəsə bilərik. Bəs görəsən səsi birbaşa kəsmək olarmı? Bunun üçün səhnəyə aşağıdakı əmrlər blokunu əlavə etmək olar:

Əmrlərdən görürük ki, “0” klavişini sıxanda səs tamamilə kəsilir (0% qiymət alır).

Səhnə üçün yazılan əmrlər aşağıdakı kimi olacaq:

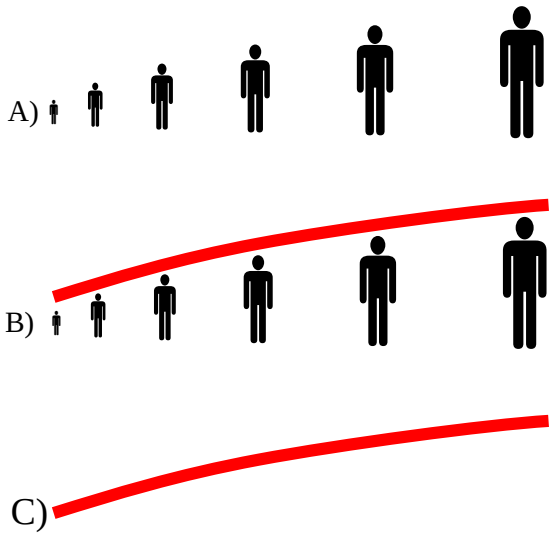


İndi sevimli spinner oyununuzu oynayın və həzz alın.
U Ğ U R L A R !!!



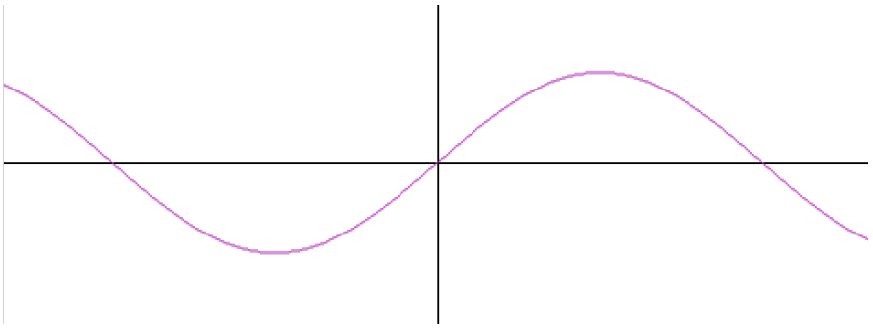
10.ƏYRİLƏR

Siz aşağı siniflərdə qrafik haqqında eşitmisiniz. Təsəvvür edin ki, Siz hansısa uşağın boyunu 1, 3, 5, 7, 9 və 11 yaşlarında ölçüb divarda yan-yana qeyd eləmisiniz. Həmin nöqtələri bir-biri ilə birləşdirsək, aşağıda gördüyünüz mənzərəni alarsınız.



A) variantı yaşa görə nöqtələri seçməyi, B) variantı həmin nöqtələri birləşdirməyi, C) variantı isə sonda alınan əyrini göstərir.

Ya da tibb aparatlarında tez-tez aşağıdakı görüntülər alınır:

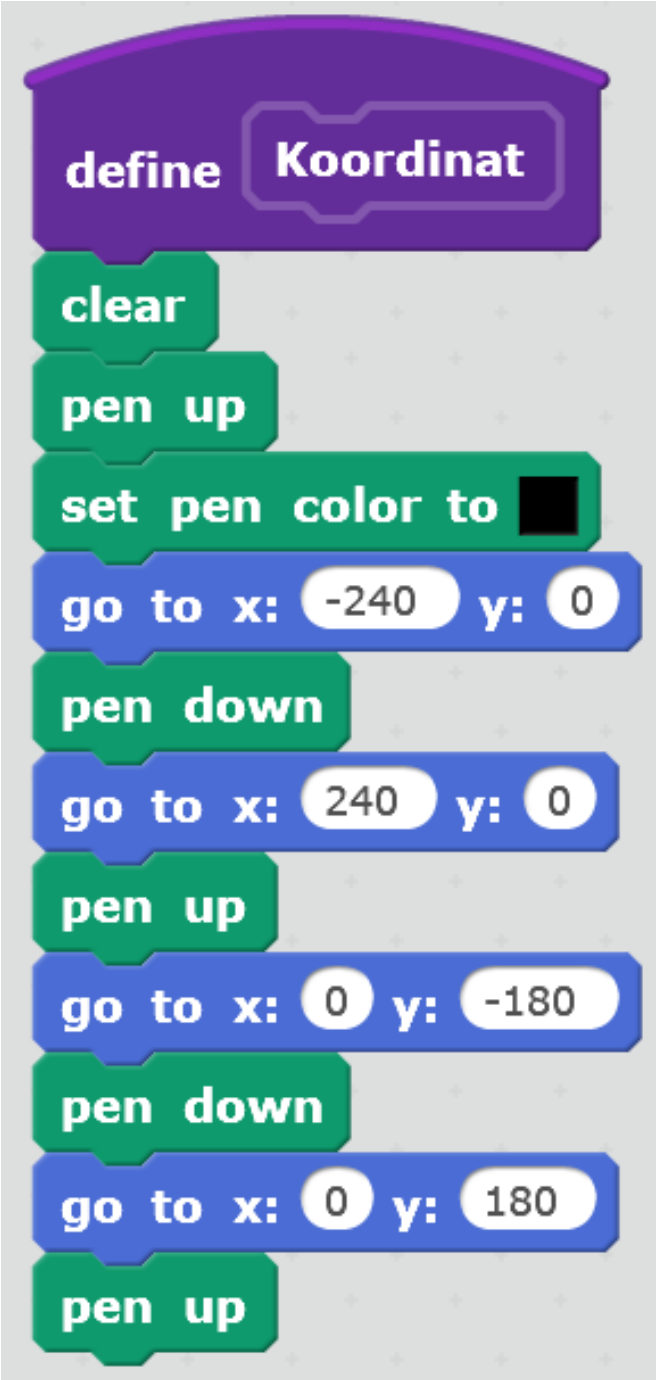


Belə görüntülər qrafik adlanır. Sonuncu görüntü isə sinusoid adlanır. Bəs görəsən sinusoidi Scratch proqramında necə çəkmək olar?

10.1. KOORDİNAT SİSTEMİ

Öncə gəlin koordinat sistemini çəkən proqram yazaq. Bunun üçün **Digər bloklar** əmrlər qrupuna keçək. **Make a block** düyməsini sıxaraq **Koordinat** adlı blok yaradaq.

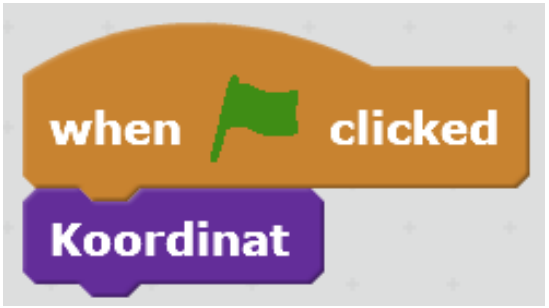
Sağda əmrlər sahəsində **define Koordinat** əmrinə aşağıda göstərilən kimi əmrlər əlavə edək.



Əmrlərə nəzər yetirsək görərik ki, o qədər də çətin proqram deyil.

- Eran təmizlənir.
- Qələm qaldırılır.
- Qələmin rəngi qara seçilir.
- Qələm (-240, 0) nöqtəsinə aparılır. Qələm qalxmış vəziyyətdə olduğundan yalnız yerdəyişmə baş verir.
- Qələm endirilir.
- Qələm (240, 0) nöqtəsinə aparılır. Qələm enmiş vəziyyətdə olduğundan yerdəyişmə iz qoymaqla (başqa sözlə üfüqi düz xətt çəkməklə) baş verir.
- Qələm qaldırılır.
- Qələm (0, -180) nöqtəsinə aparılır. Qələm qalxmış vəziyyətdə olduğundan yalnız yerdəyişmə baş verir.
- Qələm endirilir.
- Qələm (0, 180) nöqtəsinə aparılır. Qələm enmiş vəziyyətdə olduğundan yerdəyişmə iz qoymaqla (başqa sözlə şaquli düz xətt çəkməklə) baş verir.
- Qələm qaldırılır.

Bu bizim altproqramımızın mətnidir. Bu altproqramı yerinə yetirmək üçün aşağıdakı əmrləri proqrama əlavə etmək lazımdır:



Proqramın ümumi görünüşü belə olacaq:

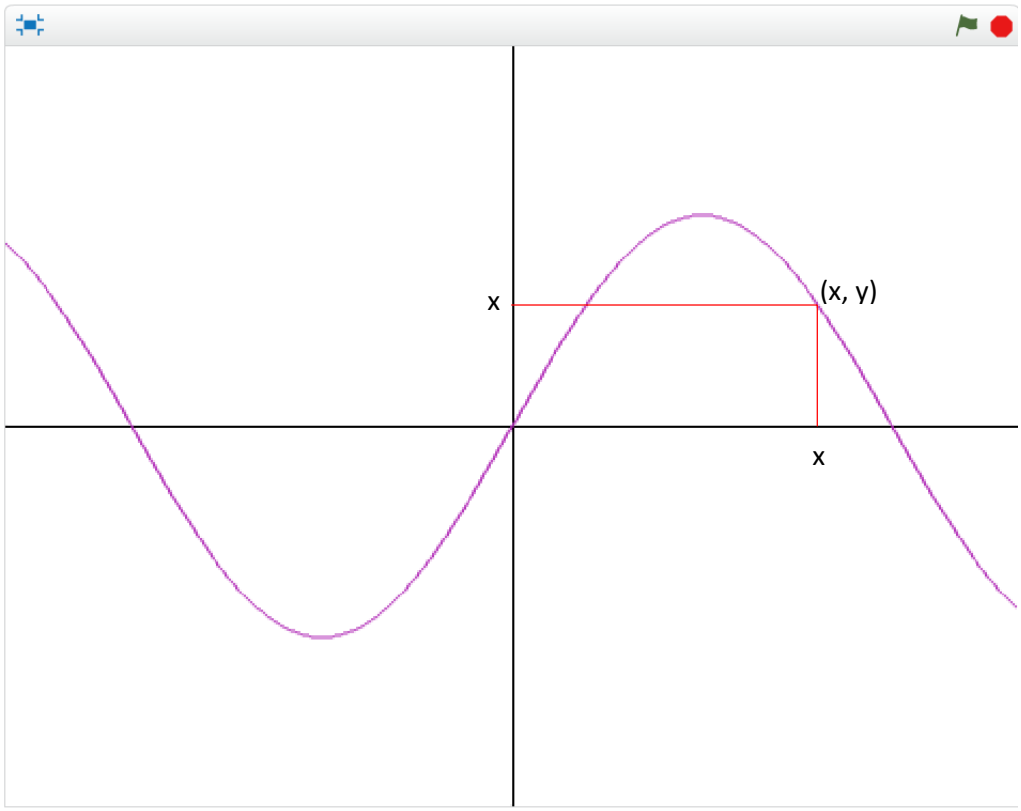
Gəlin proqramın işini yoxlayaq. Tam ekran rejiminə keçin. **Yaşıl bayraq** düyməsini sıxıb proqramı başladın.

Nəticədə ekranda aşağıdakı görüntü alınacaq:

Biz istədiyimiz, yəni koordinat sistemini göstərən üfüqi və şaquli düz xətləri aldıq.

10.2. SİNUSOİD

Bəs görəsən aşağıdakı şəkildə görünən sinusoidi neçə çəkmək olar?



Əvvəlcə bir qədər riyazı biliklərimizi yada salaq. Koordinat müstəvisində istənilən nöqtənin yeri onun koordinatını göstərən iki ədədlə ifadə edilir. Bu ədədlərdən biri onun üfüqi olan **x** oxunda koordinatını, ikinci ədəd isə **y** oxunda onun koordinatını bildirir. Deməli bizə **x** və **y** adlı iki dəyişən lazımdır. **x** adlı dəyişən üfüqi ox boyu qiymətlər alacaq və qiyməti **-240** ilə **240** arasında dəyişəcək. **y** adlı dəyişən isə şaquli xətt boyu qiymət alacaq. Onun qiymətinin **-240** ilə **240** arasında dəyişməsi mümkündür. Amma bizə şəkildə gördüyümüz əyrini almaq üçün **y** dəyişəninin qiymətini Scratch dilinin **Operators** bölməsində 

əmrində **sin**

(sinus) funksiyasının köməyilə tapacağıq.

Soldakı şəkildə **sin** (sinus) funksiyasını seçmək qaydası verilib.

y dəyişəninin qiyməti **y=sin(x)** düsturu ilə hesablanacaq. Funksiyanın adı **sinus** olduğundan alınan əyri də **sinusoid** adlanır.

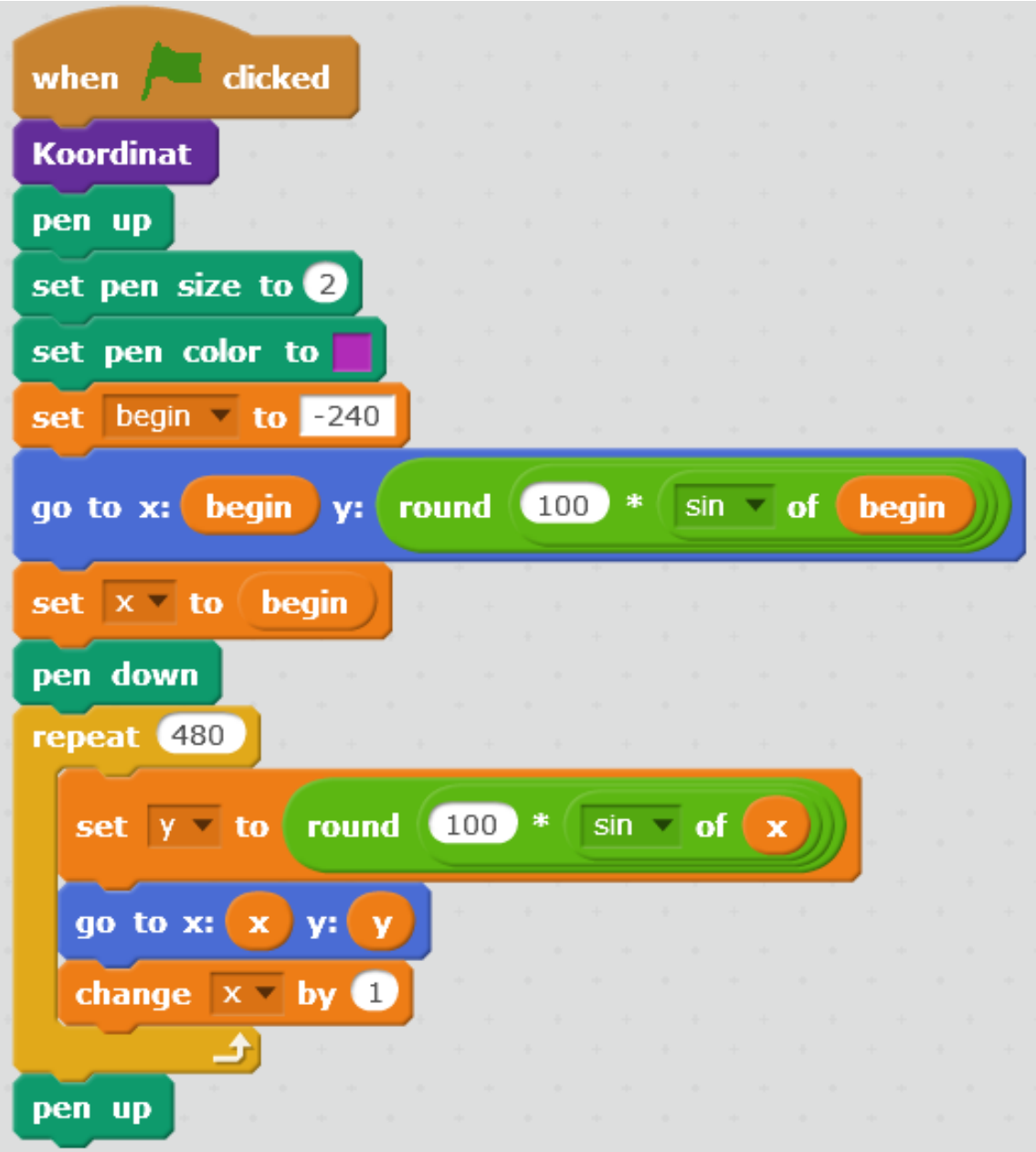
Amma burada bir şeyi nəzərə almaq lazımdır ki, **sin** funksiyası **-1** ilə **1** arasında (intervalında) qiymət alır. Ona görə də onun aldığı qiymət kəsr ədəd

olacaq. Belə halda qrafik formada biz bu xətti görməyə bilərik. Problemi həll etmək üçün düsturu **y=100*sin(x)** kimi yazacağıq. Bu halda **y** dəyişəninin qiyməti şaquli xəttdə 100 dəfə böyümüş götürüləcək və xətt istədiyimiz kimi görünəcək.

Bizə **begin** daha bir dəyişən lazımdır. Bu dəyişən üfüqi xətdə başlama nöqtəsini göstərəcək.

Dəyişənlərimiz yaradıldıqdan sonra **Data** qrupunun görünüşü belə olacaq:

Proqrama əlavə ediləcək əmrləri proqramın mətnində yerləşdirəndən sonra görüntü belə olacaq:



Yuxarıda dediyimiz kimi, birinci 2 əmr koordinat sistemini çəkmək üçündür.

Sonrakı 3 əmr qələmi qaldırır, qələmin qalınlığını müəyyən edir və onun rəngini seçir.

Set begin to -240 əmri ilə **begin** dəyişəninə səhnənin ən son sol nöqtəsi olan **-240** qiyməti mənimsədilir.

7-ci **go to** əmri qrafikin səhnədə başlama nöqtəsinə keçir. **x** koordinatı kimi **begin** dəyişəni götürülüb. Scratch proqramında koordinatlar yalnız tam ədəd ola bilər. Buna görə də **y=round(100*sin(begin))** düsturunda **y** dəyişəninin qiyməti tam ədəd kimi yuvarlaqlaşdırılır.

Bundan sonra **x** dəyişəninə başlanğıc qiymət (**begin**) mənimsədilir.

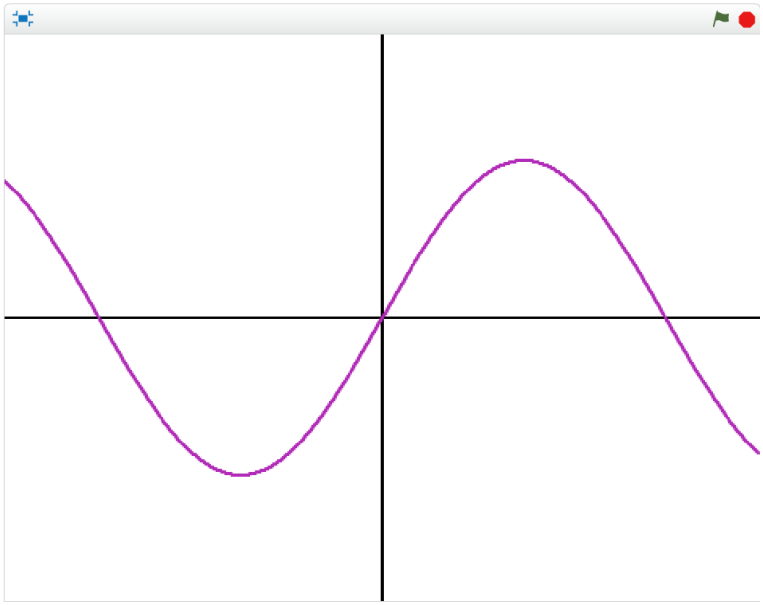
Daha bir əmr qələmi endirir.

Bildiyimiz kimi **-240** ilə **240** arasında 480 tam ədəd var. Ona görə də **repeat** əmrinin köməyi ilə **480** dəfə təkrarlanan dövr qurulur. Bu dövrün içərisinə aşağıdakı əmrlər daxil edilib:

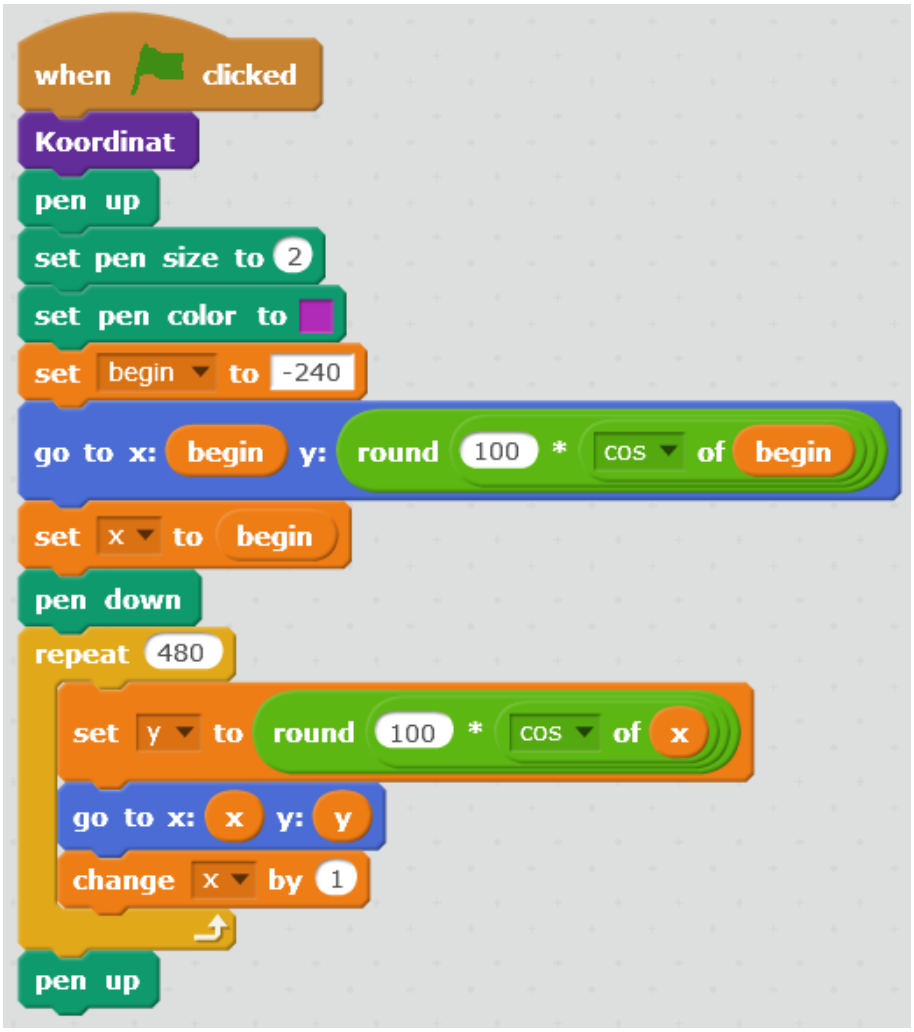
- Yuxarıda dediyimiz düstur əsasında **x** dəyişəninin cari qiyməti üçün **y** dəyişəninin qiymətini müəyyənləşdirmək.
- **(x,y)** koordinatlı nöqtəyə keçmək.
- **x** dəyişəninin qiymətini bir vahid artırmaq.

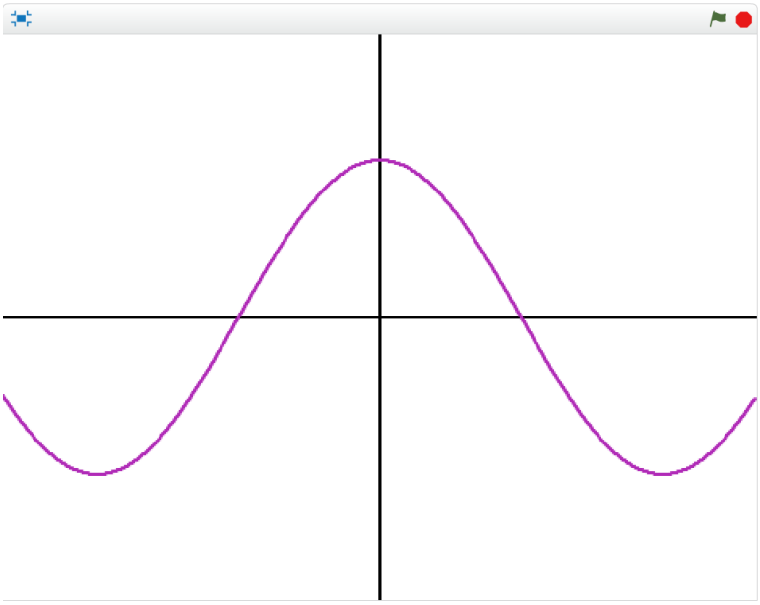
repeat dövrü **480** dəfə təkrarlanandan sonra qələm qaldırılır.

Gəlin indi tam ektan rejiminə keçib proqramı başladaq. Siz aşağıdakı görüntünün şahidi olacaqsınız:

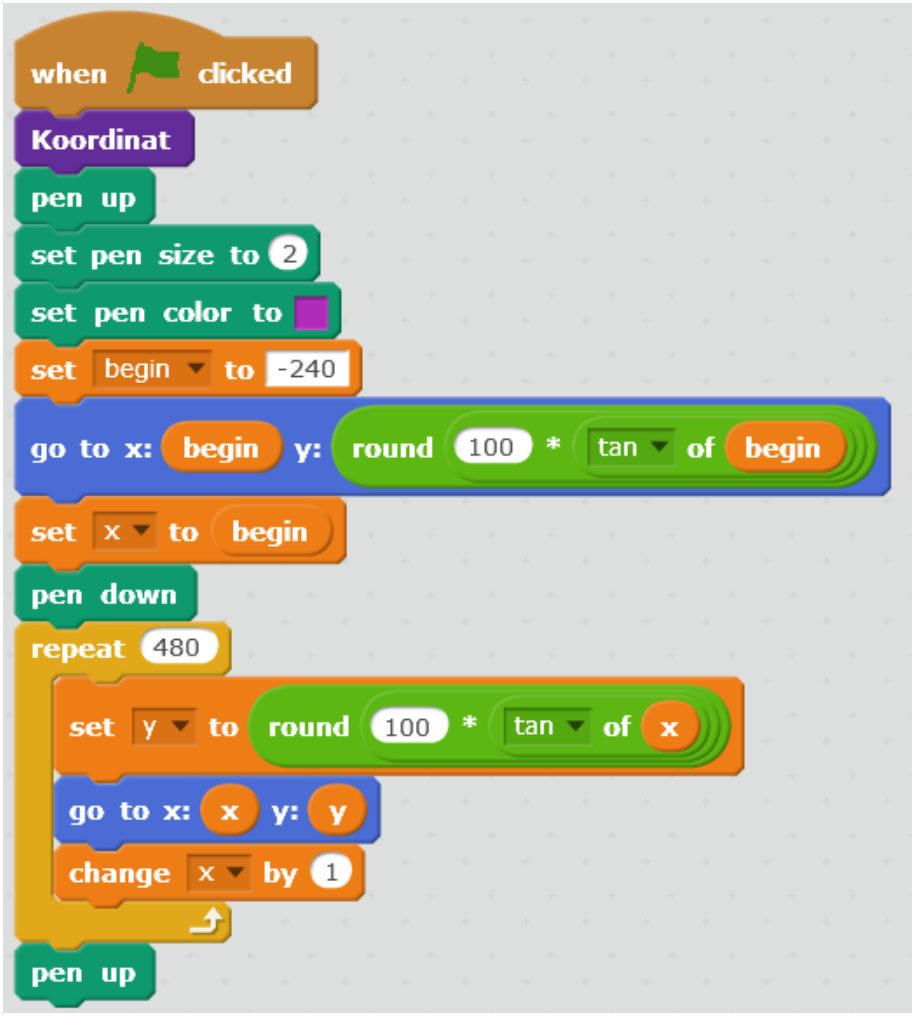


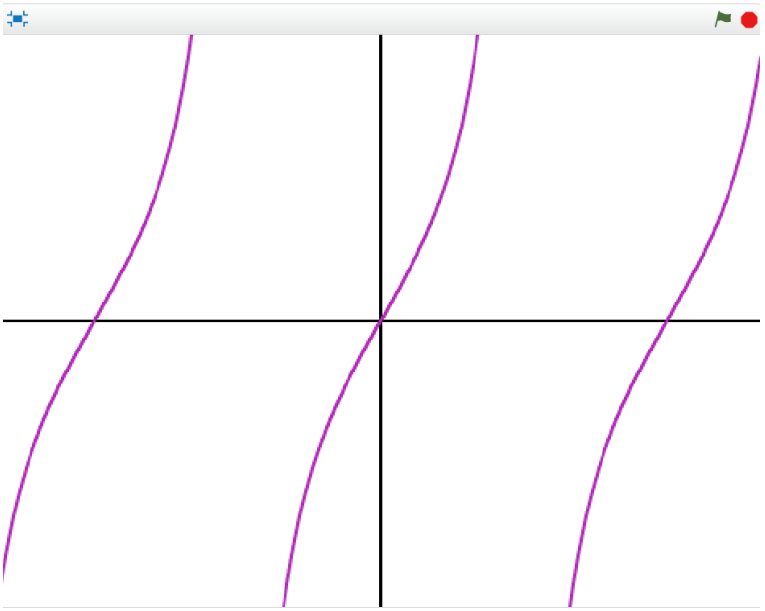
Bəs görəsən **y** dəyişəninin qiymətini hesablayan düsturda **sin** funksiyasının yerinə **cos (kosinus)** funksiyasını götürsək qrafik necə olar? Aşağıda proqramın həmin variantı və işinin nəticəsi görünür:



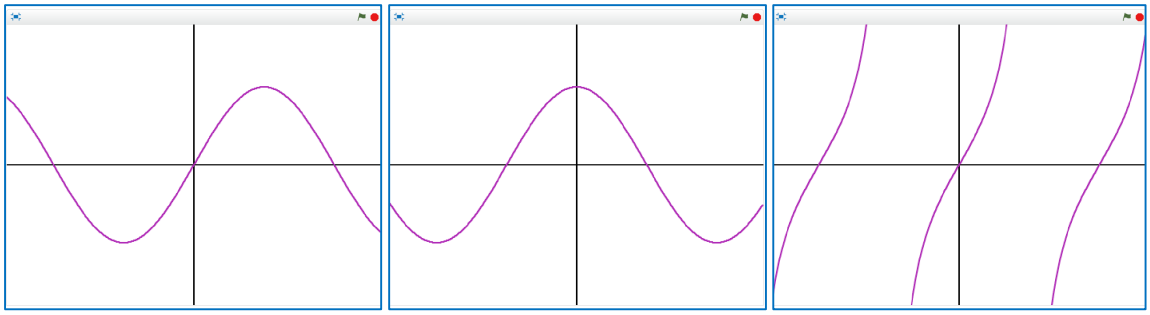


Gəlin daha bir eksperiment edək, bu dəfə funksiya kimi **tan (tangens)** götürək. Proqram və işin nəticəsi belə olacaq:





Bu üç qrafiki yan-yana qoysaq görərik ki, bu görüntülər tam fərqli formadadırlar. Deməli, biz funksiyaları dəyişməklə tamam fərqli görüntülər ala bilərik.

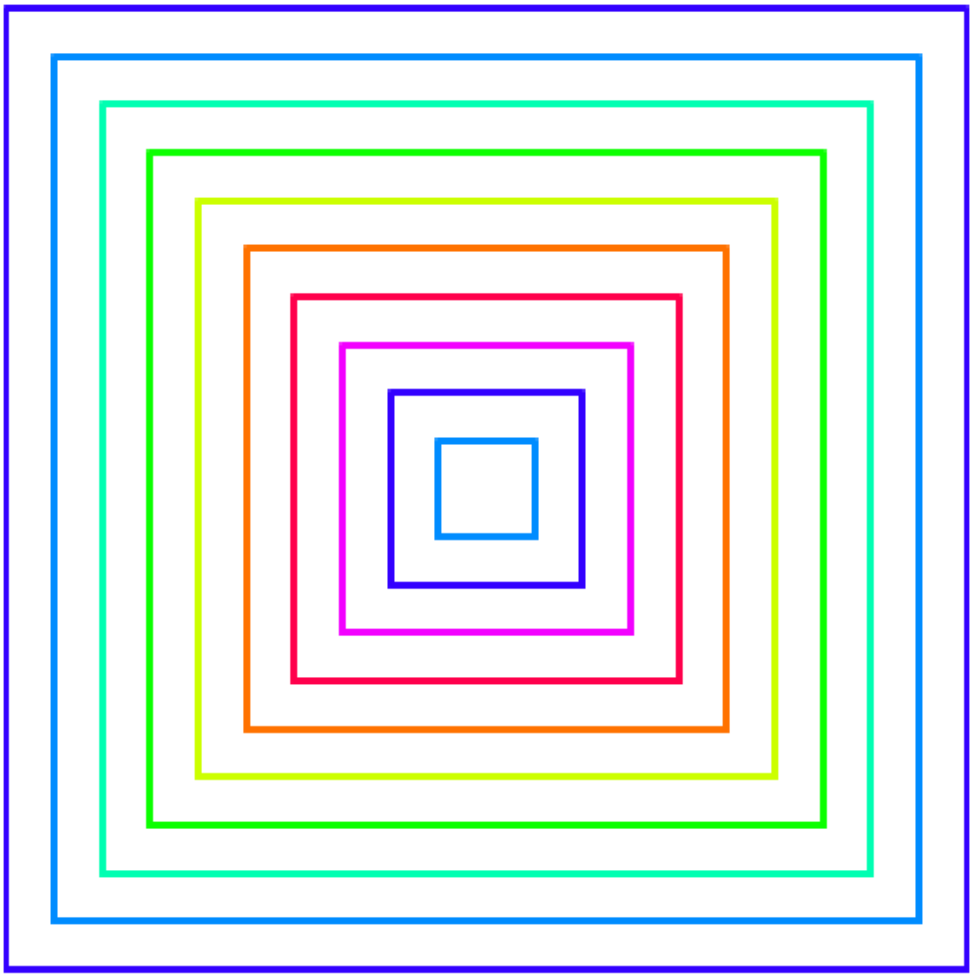


Sinus Kosinus Tangens

11. DAXİLİ KVADRATLAR

Bu kitabın əvvəli olan "SCRATCH 2.0 proqramlaşdırma dili. I hissə" adlı kitabın "5.1.2. Kvadratın çəkilməsi" bölməsində biz Scratch 2.0 proqramlaşdırma dilində kvadratın necə çəkilməsini öyrənmişdik. Bəs görəsən kvadratla bağlı daha çətin fiqurları necə çəkmək olar?

Bu dərsdə biz bir-birinə daxil olan kvadratların çəkilməsini öyrənəcəyik. Aşağıdakı şəkildə bu nümunələrdən biri verilib:



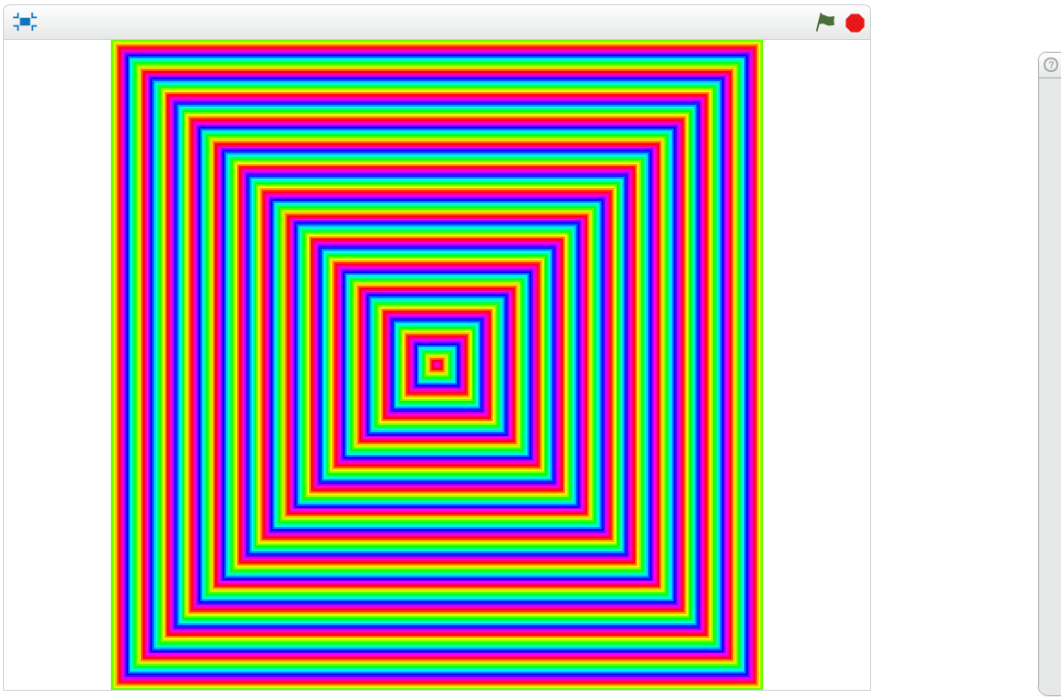
Şəklə diqqət etsəniz görərsiniz ki,

- kvadratların hamısının mərkəzi **(0;0)** koordinatındadır.
- onların ölçüləri fərqlidir.
- onların rəngləri fərqlidir.
- Kvadratların tərəfləri arasındakı məsafə bərabərdir.

Bildiyimiz kimi səhnədəki nöqtələrin sayı **480×360** ilə ölçülür. Yəni sütunların sayı 480, sətirlərin sayı isə 360-dır. Səhnənin mərkəzinin koordinatı **(0;0)** olduğundan onda üfüqi ox boyu interval **(-240;240)** arası, şaquli ox boyu interval isə **(-180;180)** arası dəyişir.

Şəkildən göründüyü kimi biz yalnız kvadratları çəkən proqram yazmalıyıq. Ona görə də üfüqi ox boyunca da intervalı **(-180;180)** arası götürməliyik. Buradan görünür ki, ən böyük kvadratın ölçüsü **360×360** olacaq. Kvadratın təpələrinin koordinatları isə **(-180;-180)**; **(-180;180)**; **(180;180)** və **(180;-180)** olacaq.

Kvadratların maksimal sayı **180** ola bilər. Amma kvadratların sayı maksimal olanda səhnənin ortasında aşağıdakına bənzər fiqur alınacaq:

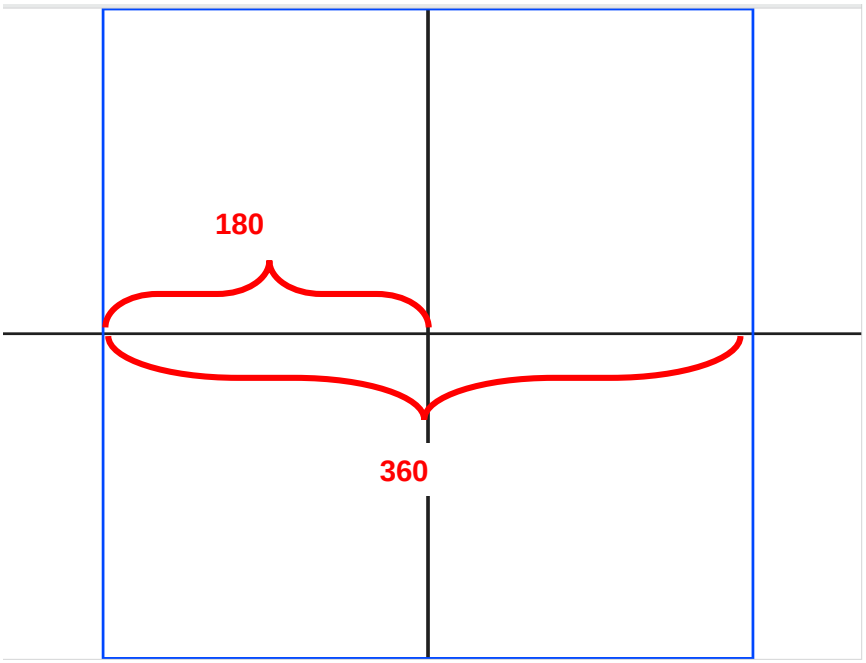


Ona görə də bizim yazacağımız proqram kvadratların sayının **180**-dan çox olmamasını yoxlamalıdır.

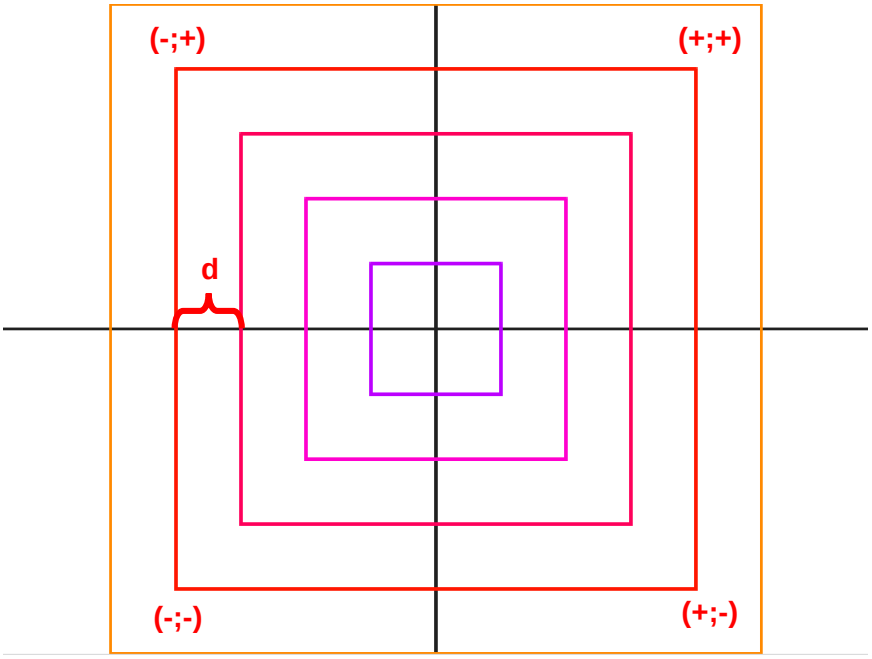
Deməli bizim alqoritmimiz təxminən belə olmalıdır.

- Kvadratların sayı daxil edilməlidir.
- Əgər kvadratların sayı **180**-dan böyükdürsə, onda proqram bu haqda məlumat verməli və təkrar kvadratların sayını soruşmalıdır.
- Kvadratların sayı maksimum **180** olanda proqram işini davam etməlidir.
- Kvadratların sayı **0** olanda səhnədə heç nə çəkilməyəcək. Kvadratların sayı **1** olanda isə səhnədə **360×360** ölçülü bir ədəd kvadrat çəkiləcək.
- Növbəti kvadratlararası məsafə hesanlanmalıdır. Bəs kvadratlararası məsafə nə qədər olmalıdır? Bunun üçün **d=180/say** düsturundan istifadə edəcəyik. Nə üçün düstur belə olmalıdır? Yuxarıda dediyimiz kimi səhnədə **360×360** ölçülü

sahədə kvadratlar çəkilməlidir. Ən böyük kvadratın tərəfi **360** nöqtədən ibarət ola bilər. Səhnənin mərkəzində üfüqi və şaquli xətt çəksək, onda kvadratın tərəfi **360** nöqtə olduğundan tərəfin yarısı **180** nöqtə olar.



Deməli, səhnənin mərkəzindən maksimal ölçülü kvadratın tərəflərinədək məsafə **180** nöqtədir. Beləliklə biz **180** ədədini kvadratların sayına bölsək, kvadratlararası məsafəni **(d)** alarıq.



Qaldı hər kvadratın çəkilməsinin başlama nöqtəsi. Bizim 4 variantımız var. Bunlar:

- Yuxarı sağ təpə - **(+ ; +)**
- Yuxarı sol təpə - **(- ; +)**

- Aşağı sol təpə - **(- ; -)**
- Aşağı sağ təpə - **(+ ; -)**

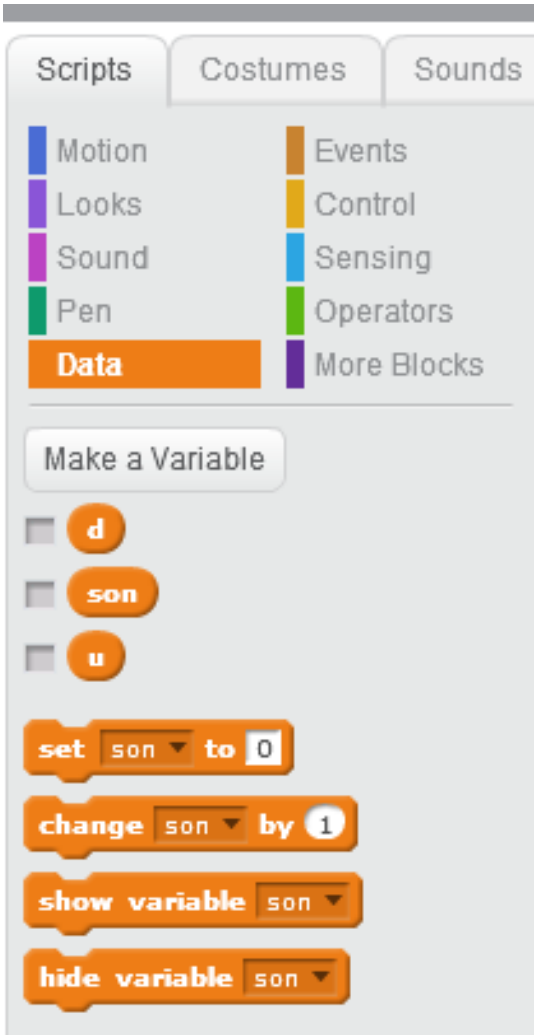
Gəlin biz kvadratın çəkilməsinin başlama nöqtəsi kimi yuxarı sağ təpəni **(+ ; +)** götürək.

Sonda bizə kvadratların tərəflərini hesablamaq üçün düsturu tapmaq lazımdır. Əgər biz **d** ilə kvadratın tərəfləri arasındakı məsafəni, **u** ilə isə kvadratın tərəfinin yarısının uzunluğunu işarələsək, onda növbəti kvadratın tərəfinin uzunluğunun yarısı **u=u+d** düsturu ilə tapılar. Tam tərəfin uzunluğu isə **2×d** düsturu ilə tapılar.

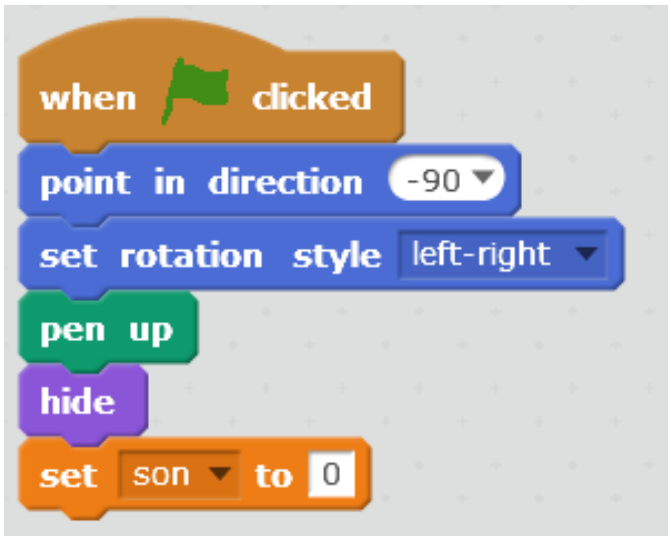
Artıq proqramımızı yaza bilərik.

Öncə 2 dəyişən yaradaq:

- **d** – kvadratın tərəfləriarası məsafə.
- **u** – kvadratın tərəfinin yarısının uzunluğu.
- **son** – dövrün sonunu bildirən əlamət. Əgər **son=0**, onda kvadratların sayı **180**-dan çoxdur. **son=1** olanda isə kvaratları çəkmək olar. Burada **son** dəyişəni başlanğıcda “**0**” qiyməti alır.



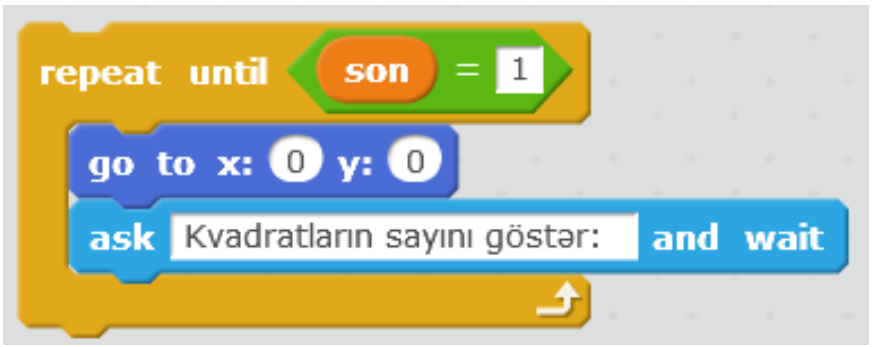
İlk əmrlər belədir:



Bu əmrlərdən əvvəllər istifadə etmişik deyə onları şərh etməyəcəyik. Yalnız sonuncu əmr, yuxarıda dediyimiz kimi, **son** dəyişəninin başlanğıcda "**0**" qiyməti almasını təmin edir.

Növbəti əmrlər proqrama əlavə ediləndən sonra görünüş belə olacaq:

Əlavə edilən yeni əmrləri şərh edək.



repeat until son=1 əmrində **son=1** şərti ödənənə qədər dövr davam edəcək və daxili əmrlər yerinə yetiriləcək. Bunlar **(0;0)** koordinatına keçid və "Kvadratların sayının daxil edilməsini" istəyən **ask** əmrləridir.

Daha sonra proqrama əlavə edilən əmrlər klaviaturadan daxil edilən ədədin **180**-dan böyük olmasını yoxlayır. Bildiyimiz kimi klaviaturdan daxil edilən məlumat **answer** adlı buferə yerləşdirilir. Biz klaviaturada cəmi bir ədəd daxil etdiyimizdən, **answer** adlı buferdən dəyişən kimi istifadə edə bilərik.

Kvadratların sayını daxil etdikdən sonra onun düzgünlüyünü yoxlamaq lazımdır. Bunun üçün aşağıda verilmiş tam formatlı **if** əmrindən istifadə etməliyik:

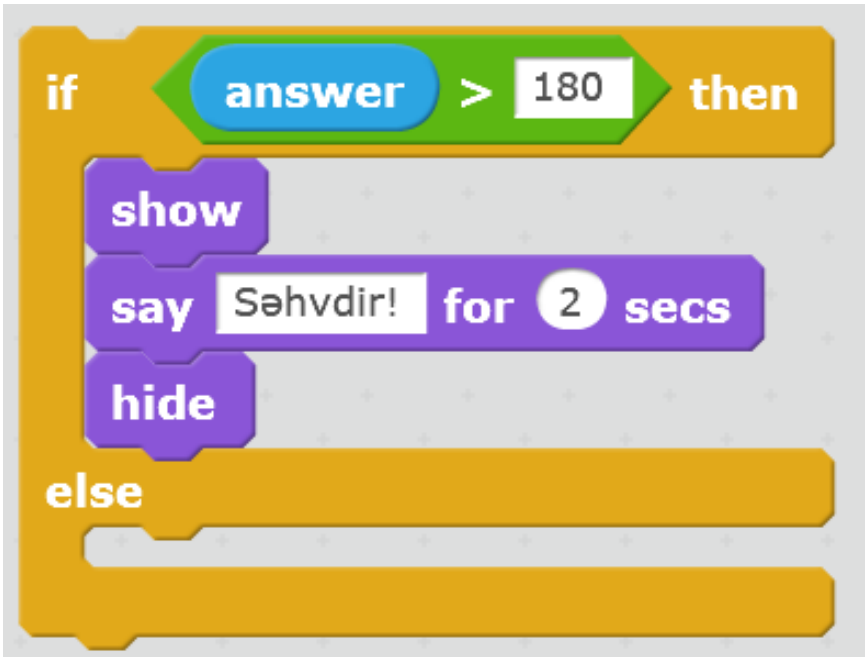
Şərt kimi **answer** buferinin qiymətini yoxlamalıyıq. Burada 2 variant ola bilər:

1. Şərt kimi 

götürərik və şərt ödənən halda əmrin **then** sahəsində ekrana səhv haqqında məlumat verib yenidən kvadratların sayını daxil etməyi təklif edərik. Əmrin **else** sahəsində isə proqramımızın kvadratları çəkən əsas hissəsinin əmrlərini yerləşdirərik.
2. Şərt kimi 

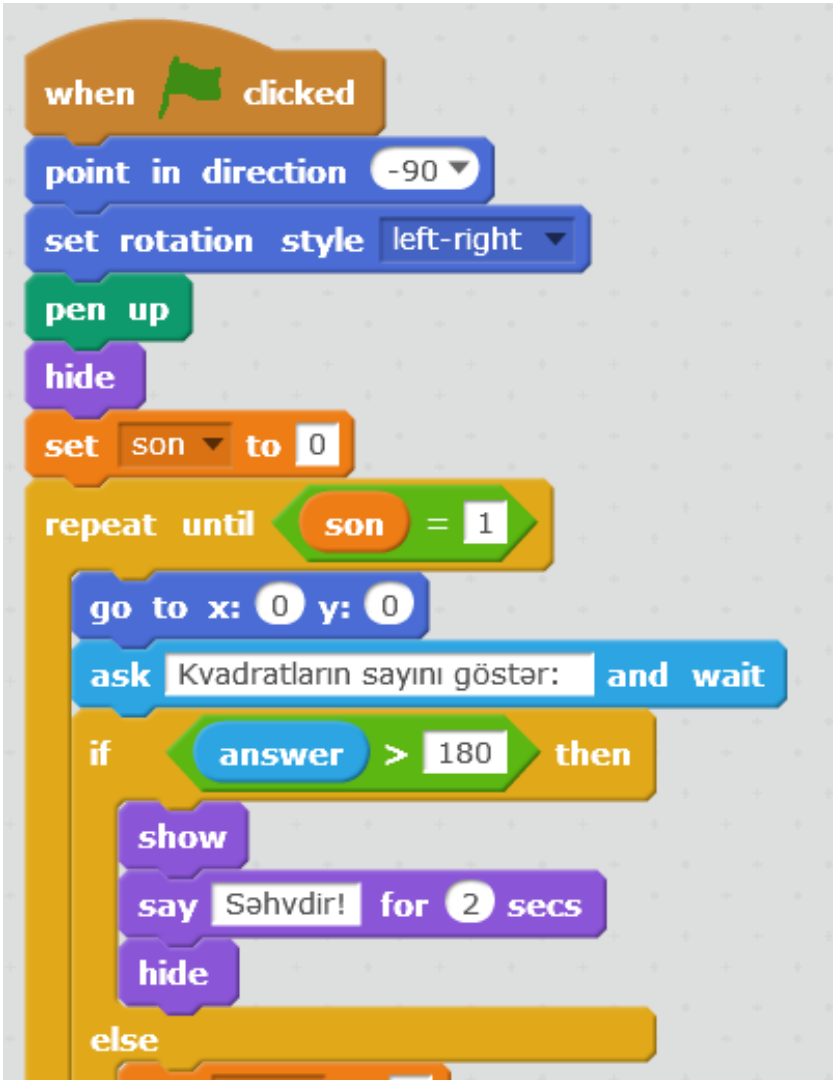
götürərik və şərt ödənən halda əmrin **then** sahəsində proqramımızın kvadratları çəkən əsas hissəsinin əmrlərini yerləşdirərik. Əmrin **else** sahəsində isə ekrana səhv haqqında məlumat verib yenidən kvadratların sayını daxil etməyi təklif edərik.

Biz birinci variantı seçəcəyik.



Əmrlərdən görürük ki, proqramın başlanğıcında gizlədilmiş bizim personajımız (pişik) görünən edilir. Ekrana 2 saniyə müddətində "Səhvdir!" sözü çıxır. Sonda personajımız gizlədilir və proqram işini davam edir.

Bu halda proqramımız belə görünəcək:



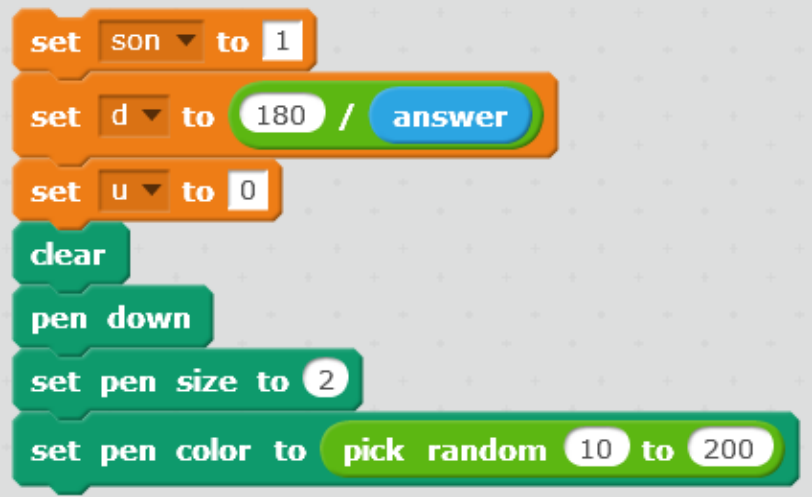
İndi isə proqramımızın **if** əmrindəki **else** sahəsinə əlavə ediləcək əmrlərini yazaq:

İlk əmr olaraq **son** adlı dəyişənə "**1**" mənimsədilir. Bu ona görə edilir ki, **else** sahəsindəki əmrlər yerinə yetirildikdən sonra proqram işini bitirsin.

d=180/say düsturu ilə kvadratlararası məsafə tapılır. Bizim halda say kimi **answer** dəyişəni götürülür.

Kvadratın tərəfinin uzunluğunun yarısını hesablamaq üçün **u** adlı dəyişəndən istifadə edirik. Başlanğıcda bu dəyişənə "**0**" mənimsədirik.

Başlanğıc əmrlər aşağıda verilib:



Şəkildən göründüyü kimi, daha sonra aşağıdakı işlər görülür:

- ekran təmizlənir,
- qələm aşağı endirilir,
- qələmin ölçüsü "**2**" götürülür,
- qələmin rəngi **pick random** əmri ilə təsadüfi seçilir.

Kvadratların çəkilməsinə hazırlıq məqsədilə proqrama əlavə edilən bu əmrlərdən sonra artıq daxil edilən say qədər kvadratların çəkilməsi üçün əmrlər bloku yazılır. Bunun üçün **repeat** əmri **answer** dəyişəni qədər təkrar edilir. **repeat** əmrinin daxilində isə aşağıdakı əmrlər olacaq.

- Qələm qaldırılır.
- **u** dəyişəninin qiyməti **d** qədər artırılır.
- **x** və **y** koordinatlarına **u** dəyişəninin qiyməti mənimsədilir.
- Qələm endirilir.
- Qələmin rəngi "15" vahid artırılır.
- Cari kvadratı çəkmək üçün əmrlər qrupu. Bu qrup 4 dəfə təkrarlanan **repeat** əmrindən və ona daxil ola 2 ədəd hərəkət etmək və sola 90 dərəcə dönmək əmrlərindən ibarətdir. Bu əmrlərdən biri olan 

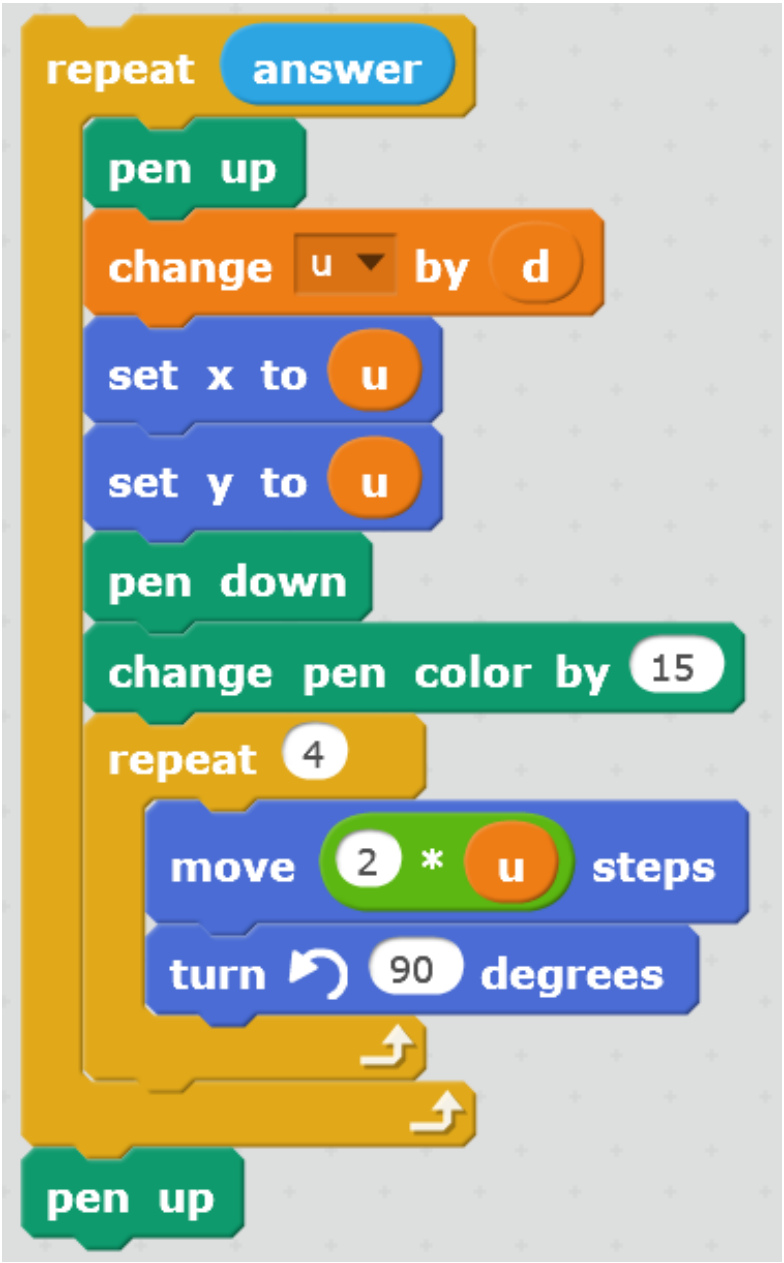
əmri cari kvadratın tərəfinin uzunluğunu tapır. **u** dəyişəninin qiyməti cari kvadratın tərəfinin uzunluğunun yarısına bərabərdir. **2×u** düsturu isə tərəfin

tam uzunluğunu verəcək. 

əmri isə kvadrat çəkmək üçün lazım olan olan 90 dərəcə dönmə bucağını verir.

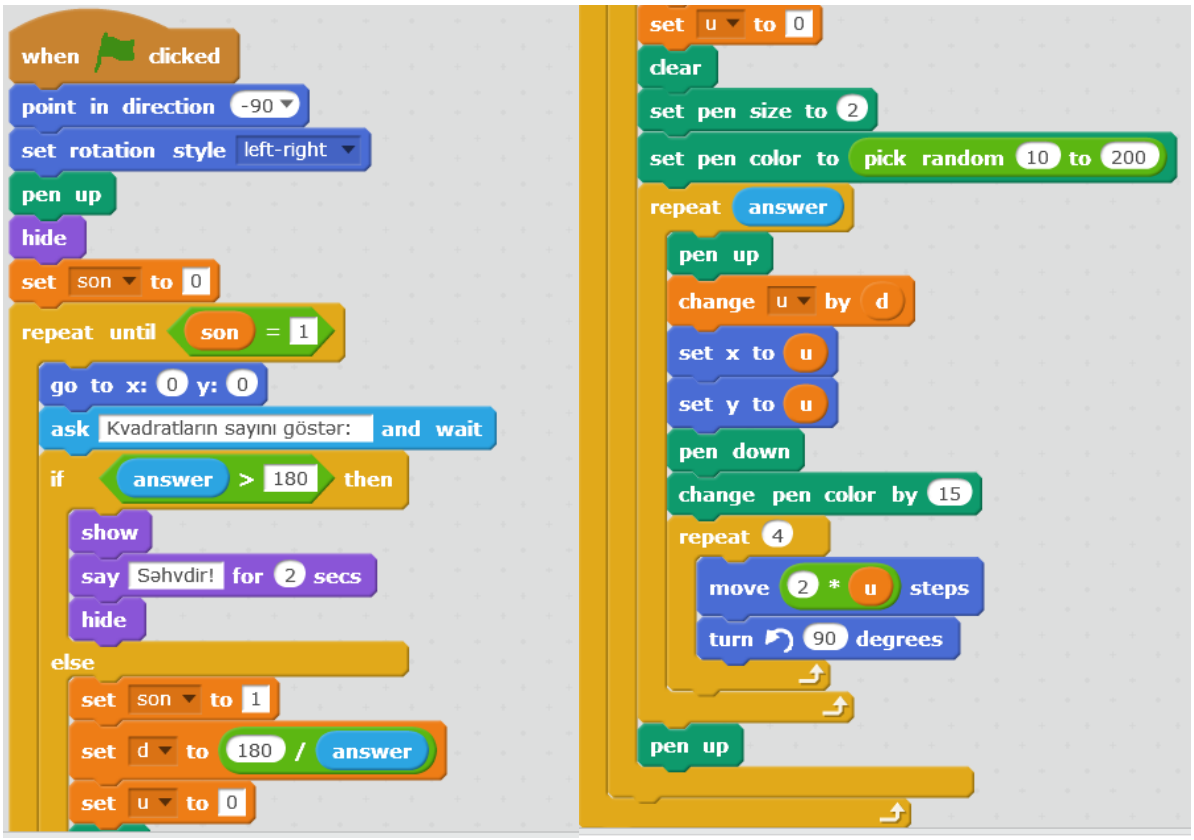
- Sonda qələm qaldırılır.

if əmrinin **else** sahəsinə əlavə edilən yeni əmrlər aşağıdakılardır:



Beləliklə, biz bir-birinə daxil olan kvadratları çəkmək üçün proqramı yazıb qurtardıq. Əlbəttə Siz **Digər bloklar** əmrlər qrupunun imkanlarından istifadə edib proqramınızı daha asan oxunaqlı edə bilərsiniz. Bu artıq Sizin ixtiyarınıza buraxılır.

Aşağıda proqramın tam mətni verilib. Mətn böyük olduğundan biz onu 2 yerə bölüb yan-yana vermişik.



Sonda səhnənin tam ekran rejiminə keçin. **Yaşıl bayraq** düyməsini sıxaraq proqramı başladın. Əgər dediklərimizə riayət etmisinizsə, onda bir-birinə daxil olan kvadratlar üçün istədiyimiz görüntüləri ala biləcəksiniz.

12.Son söz əvəzi

Əziz oxucumuz, **SCRATCH 2.0** üzrə bu ikinci kitabda artıq **SCRATCH** proqramlaşdırma dilində yazılmış nisbətən çətin proqramlarla tanış oldunuz. Bu son deyil. Sən artıq **SCRATCH** aləminin daha dərinliklərinə baş vura bilərsən. **SCRATCH 2.0** proqramlaşdırma dili üzrə yazılan bu 2 kitab **SCRATCH** adlı aysberqin görünən üst tərəfidir. Aysberqin su altında qalan hissəsi qat-qat böyük olduğu kimi, **SCRATCH** proqramlaşdırma dilinin imkanları da demək olar ki, həddindən çoxdur.

Bu kitabı oxuyub sona çatanda Sən hiss edəcəksən ki, proqramlaşdırmada doğrudan da ilk uğurlu addımlar atmısan. Bizim məqsədimiz də proqramlaşdırmanın Sənin həyatında xüsusi yer tutmasına nail olmaq idi.

Bu kitabdakı proqramlarla tanış olandan sonra Sən yenə də seçim qarşısında qalırsan. Bu halda ortaya yenə eyni sual çıxır:

GÖRƏSƏN BU QƏDƏR KİFAYƏT EDƏRMİ?

Sən artıq proqramlaşdırmada ilk addımlarını atmısan və bundan sonra növbəti addım atmağa hazırsansa, onda Sənə gələcəkdə növbəti bir kitabımızıda təklif edəcəyik. Bu kitab "**Scratch 2.0 proqramlaşdırma dili. Peşəkar proqramçılar üçün alqoritmlər.**" adlanacaq. Bu kitabda həyatını peşəkar proqramlaşdırmaya həsr etmək istəyənlər üçün **Scratch 2** proqramlaşdırma dilində yazılmış və daha çətin alqoritmlərə malik maraqlı proqramlar təqdim ediləcək.

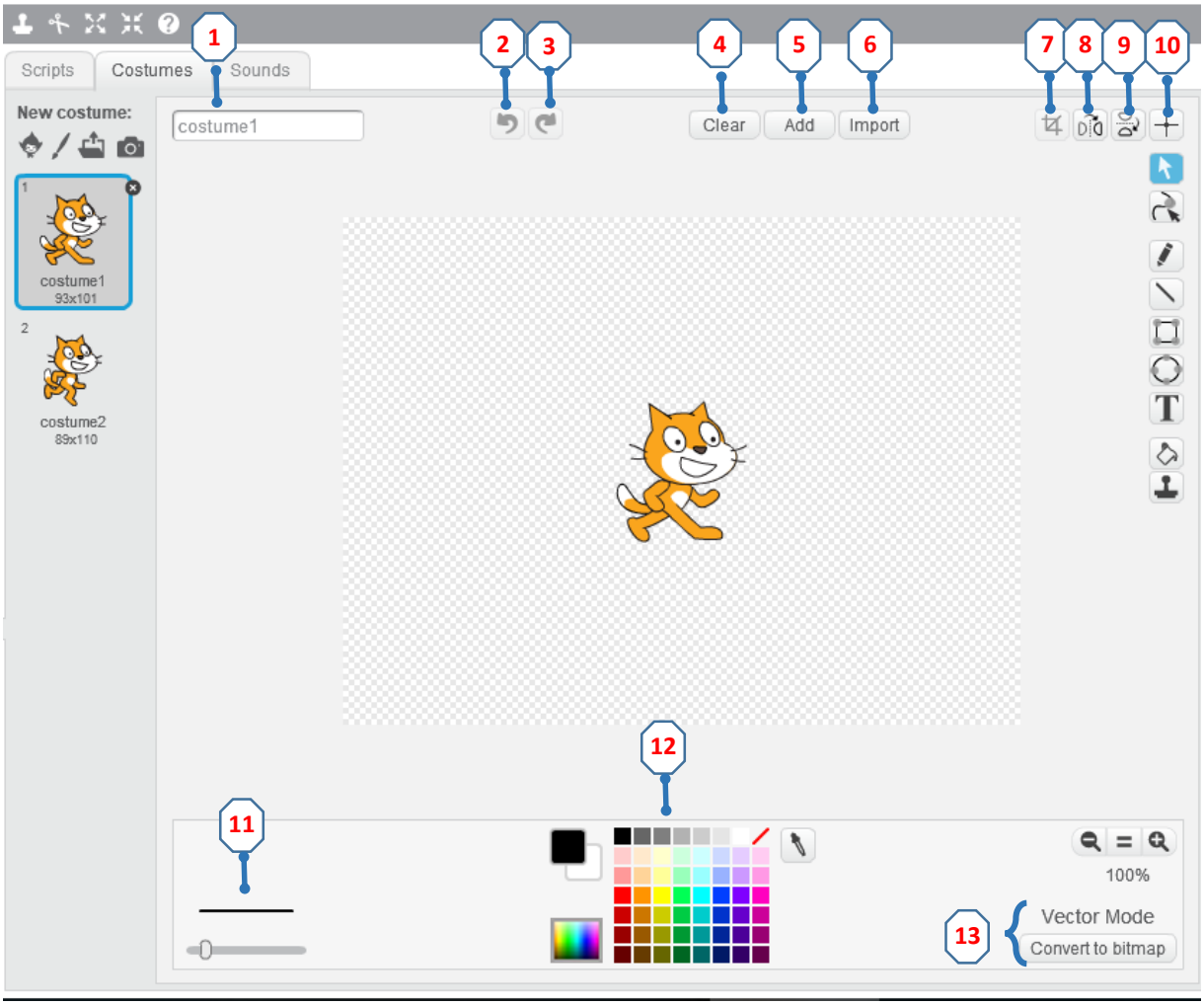
e-mail: abdullaqehreman@gmail.com **Telefon:(+99450) 3503920**

13. Ə L A V Ə L Ə R

13.1. Daxili qrafik redaktor (Paint)

SCRATCH 2.0 proqramının daxili **Paint** proqramı mövcuddur. Aşağıda həmin Paint proqramının interfeysi verilibdir.

Əsas interfeysdə **Costumes** menyusu seçib Paint proqramına qayıdaq. Ekranda aşağıdakı görünüş olacaq:



1 - **Şəklin (Personajın) adı (Character Name).** Şəklin (Personajın) adını redaktə edin.

2 - **Ləğv edin(Undo).** Əgər iş zamanı səhv etmişsinizsə, onda bu düyməni sıxdıqda axırıncı hərəkəti ləğv edəcək.

3 - **Təkrar edin (Redo).** Əgər Siz ləğv etdiyiniz sonuncu addımı yenidən bərpa etmək istəyirsinizsə, onda bu düyməni sıxın.

4 - **Sil (Clear).** Sahəni (ekranı) təmizlə.

5 - **Əlavə et (Add).** Sahəyə kitabxanadan şəkil daxil et.

6 - **İmport (Import).** Sahəyə kənar fayldan şəkil əlavə et.

7 - **Seçilmişi kəs (Crop to selection).** Seçilmiş fraqmentdən kənarda olanı kəsir.

8 - **Sola-sağa fırla (Flip left-righat).** Seçilmiş fraqmenti sola-sağa döndərir.

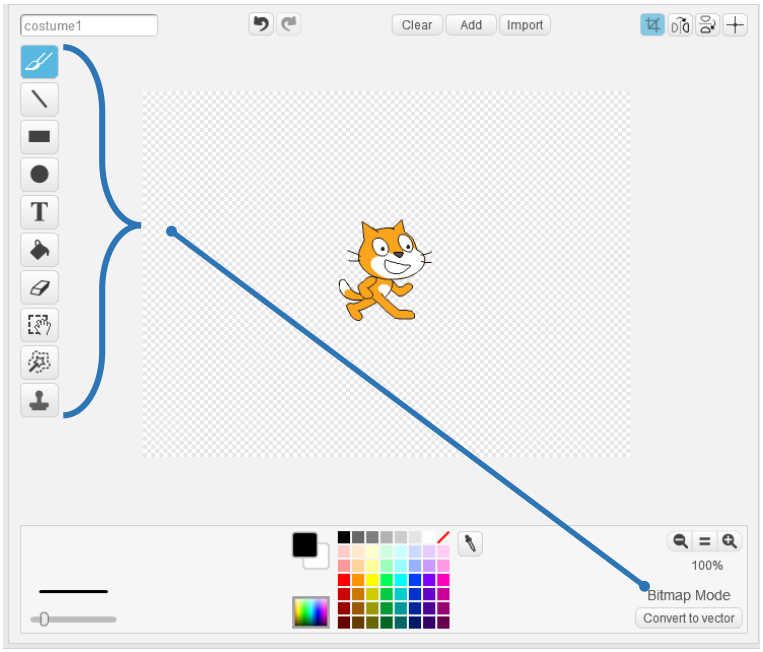
9 - **Yuxarıya – aşağıya fırla (Flip up-down).** Seçilmiş fraqmenti yuxarıya-aşağıya döndərir.

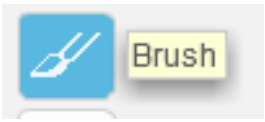
10 - **Kostyumun mərkəzini seç (Set costume center).** Seçilmiş fraqmentdən kənarda olanı kəsir.

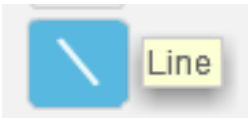
11 - **Xəttin qalınlığı (Line width).** Xəttin qalınlığını dəyişmək olar.

12 - **Rəng (Color).** Şəkil və ya forma çəkmək üçün yeni rəng seçmək olar.

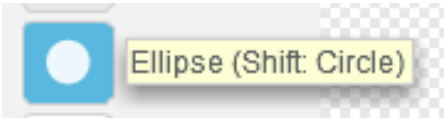
13 - **Rastr və ya Vektor rejimi (Bitmap or Vector mode).** Rastr və ya vektor qrafikaya keçid. Rastr qrafika rejimi seçəndə işçi sahənin sol tərəfində, qrafik rejimdə isə işçi sahənin sağ tərəfində qrafika ilə işləmək üçün alətlər əmələ gəlir.



Fırça 

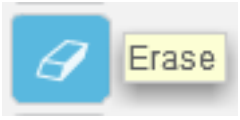
Xətt 

Düzbucaqlı 

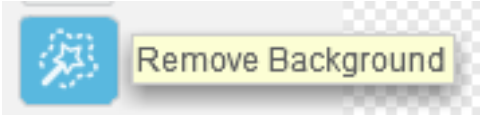
Ellips 

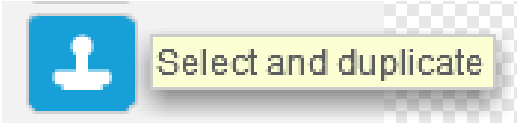
Mətn 

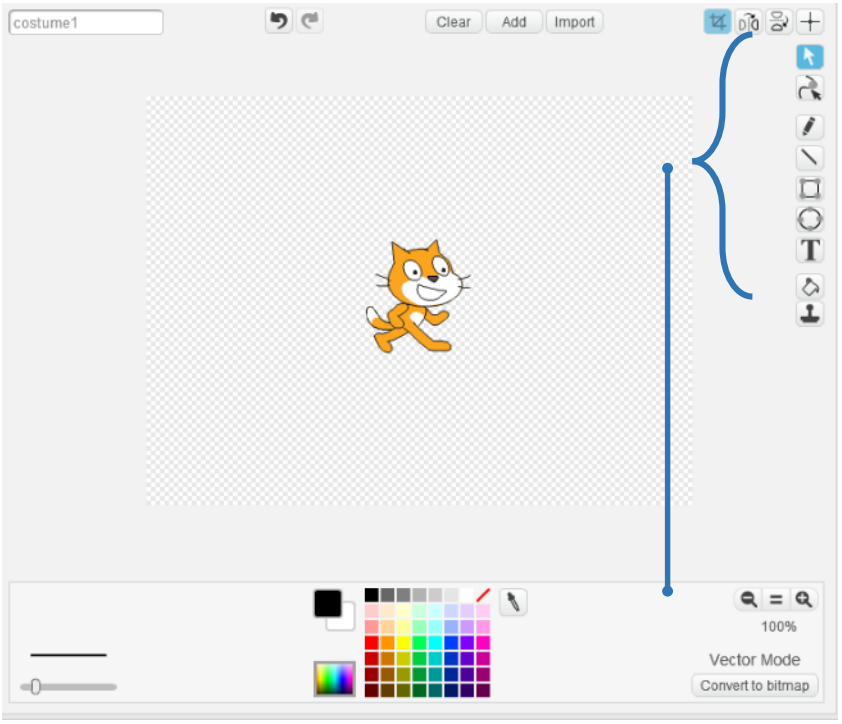
Rəngləmə 

Silmək 

Seçmək 

Fonu silmək 

Seç və surət çıxar 



Seçmə

Formanın dəyişdirilməsi

Qələm

Xətt

Düzbucaqlı

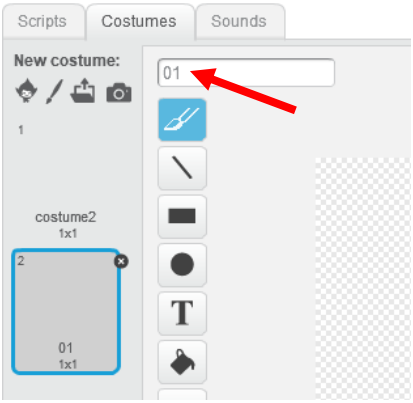
Ellips

Mətn 

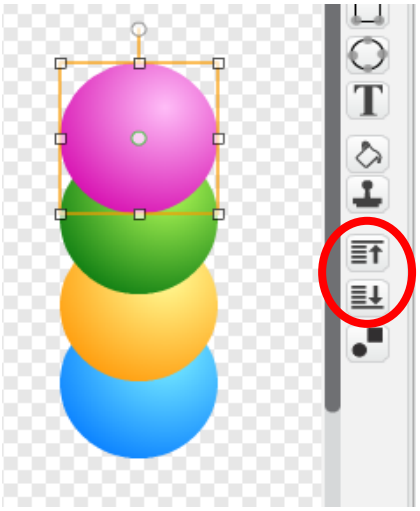
Rəngləmə 

Surətini çıxarma 

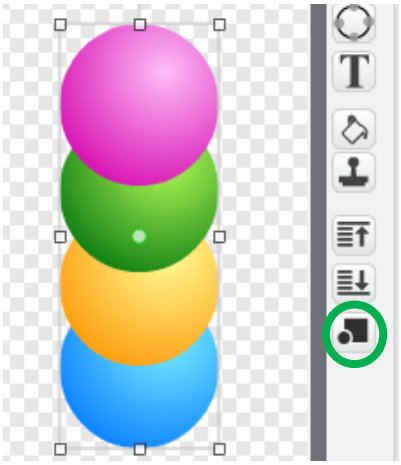
Daxili Paint proqramının **New costumes** bölməsində fırça işarəsi olan düyməni sıxdıqda biz yeni boş kostyum alacağıq. Qırmızı oxla göstərilən hissədən kostyumun adını dəyişib istənilən ad qoya bilərik (məsələn, **01**).



Vektor rejimdə işçi sahədəki fiqurlardan birini seçdikdə sağdakı alətlər içində qatları dəyişmək alətləri (qatı irəli, qatı arxaya) əmələ gəlir. Kürəcikləri laylara görə düzmək üçün fiquru seçib sağ tərəfdə alınan 2 düymədən (həmin düymələr qırmızı halqaya alınıb) birini seçməklə fiqurun yerini üst və ya alt qata dəyişmək olar.



Bir neçə fiquru qrulaşdırmaq istəyiriksə, onları birgə seçirik. Növbəti addımda dörd fiquru birgə seçirik və sağda yaşıl halqaya alınmış düyməni sıxmaqla onları qruplaşdırırıq.



İstinadlar

1. Qəhrəmanov A.B., Sadıqova S.M., Hacıyeva D.D., Rəcəbova K.S. və Cəfərova İ.A. Scratch 2.0 proqramlaşdırma dili. Müəllim nəşriyyatı. Bakı, 2016.
2. Golikov Denis & Golikov Artem. Scratch 2.0 Programming. Making games and cartoons. Detailed step by step guide for children. Copyright Golikov Denis & Golikov Artem, 2015
3. Голиков Денис и Голиков Артём. Программирование на Scratch 2. Часть 1.
4. Голиков Денис и Голиков Артём. Программирование на Scratch 2. Часть 2.
5. Голиков Денис и Голиков Артём. Книга юных программистов на Scratch 1.4. Вторая редакция. Copyright Голиков Денис и Голиков Артём, 2014.
6. Еремин Е.А. Газета «Информатика». Среда Scratch – первое знакомство. – М.: Первое сентября, 2008 – №20 (573) – С. 16–28.
7. https://scratch.mit.edu/ Scratch proqramının rəsmi saytı
8. http://letopisi.ru/index.php/Скретч - Скретч в Летописи.ру
9. http://megogo.net/ru/news/view/1562261-kogda-mozhno-nachinat-obucheniyu-programmirovaniyu.html
10. http://muz-teoretik.ru/pauzy-v-muzyke/
11. http://muz-teoretik.ru/raspolozhenie-not-na-klaviature-pianino/
12. http://nbspace.ru/math/
13. http://scratch.ucoz.net
14. http://scratch.uvk6.info/ Scratch proqramı üzrə sayt
15. http://scratch4russia.com/store/
16. http://setilab.ru/scratch/category/commun - Scratch öyrənin.
17. http://www.7not.ru/theory/01.phtml
18. http://www.iteach.ru/exp/articles.php?mpt_id_text=115
19. http://www.webplanet.ru/review/entertainment/2008/01/22/scratch21.html
20. http://zarapina.blogspot.com/p/scratch_2116.html
21. https://wiki.scratch.mit.edu/wiki/Scratch_2.0

Abdulla Qəhrəmanov

Abdulla Qəhrəmanov 1978-ci ildə BDU-nun Tətbiqi riyaziyyat fakültəsini bitirmişdir. Dövlət Statistika Komitəsində, Maliyyə Nazirliyi, Az. DETK Bakı ETM-də proqramlaşdırma və tədris şöbələrinə rəhbərlik etmişdir. Hal-hazırda informatika fənnindən dərs deyib. Kurikulum üzrə təlimçidir. 10 vəsaitin 200-dən artıq elektron resursun, 3 səmərələşdirici təklifin müəllifidir.

Sevda Sadıqova

Sevda Sadıqova 1994-cü ildə BDU-nun Tətbiqi riyaziyyat fakültəsini bitirmişdir. Bakı ş. 167 №-li tam orta məktəbdə informatika fənnini tədris edir. Son illər İT sahəsində araşdırmalar aparır. Avstriyada, Türkiyədə keçirilən beynəlxalq konfranslarda öz məqalələri ilə iştirak etmişdir.

İlahə Cəfərova

İlahə Cəfərova 2003-cü ildə ADNA-nın Cihazqayırma fakültəsini İnformasiya-ölçmə texnikası və texnologiyası ixtisası üzrə bitirmişdir. 2008-ci ildən Bakı ş. 23 №-li məktəbdə İnformatika fənnini tədris edir. "Python proqramlaşdırma dili" kitabının (2015-ci il) həmmüəllifi, 5 vəsaitin, 30-dan çox elektron resursun müəllifidir. Bir çox təhsil sərgilərinin iştirakçısıdır.