

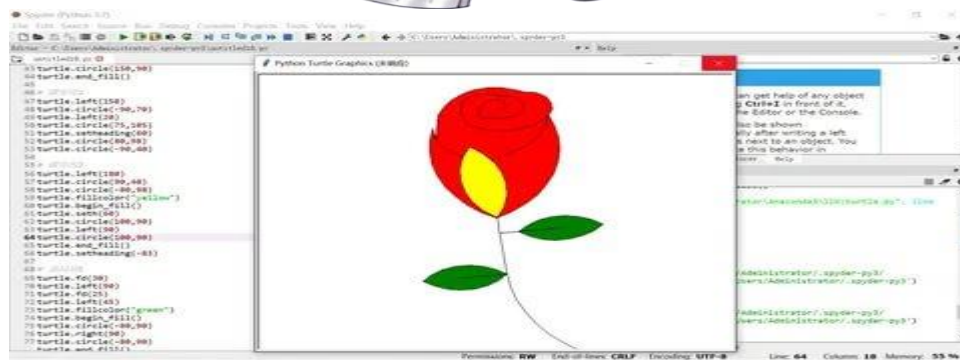
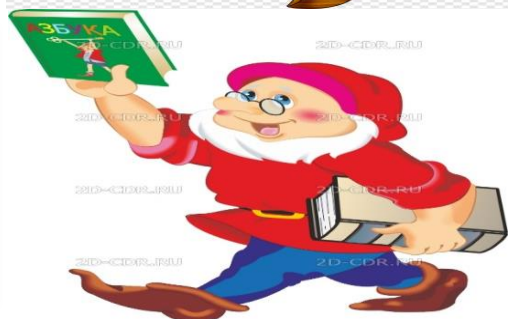
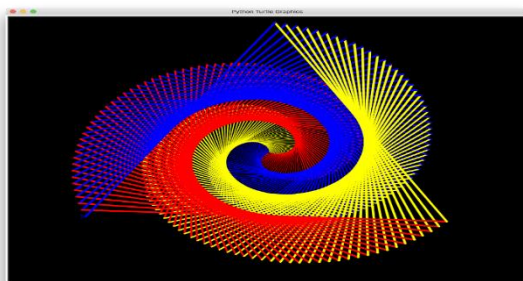
O'ZBEKISTON RESPUBLIKASI

OLIV VA O'RTA MAHSUS TA'LIM VAZIRLIGI

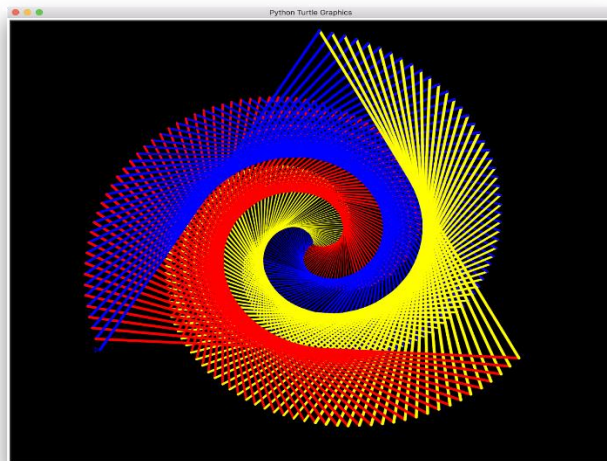
NAMANGAN DAVLAT UNIVERSITETI

GULSHOD YUNUSOVA

**OTA ONALAR, BOLALARGA PYTHON DASTURLASHTIRISHDAN
MURABBIY BO'LING!!!**



NAMANGAN 2021



Ushbu kitob bolalarni Python dasturlashtirishni o'ynab o'rganishga mo'ljallangan. O'rta va onalari bilan bolalarni o'rganishi tavsiya etiladi. 5-8 yoshli iqtidorli bolalar uchun hamda 9 yoshdan kattta bolalar uchun mo'ljallangan.



Dasturlashtirish inson miyasi faoliyatining eng yuqori mushohadalash va tafakkur oshirish istiqboli bo'lgani uchun ham biz bolalarni yoshligidan dasturlashtirish uchun o'yin shaklida kiritishimiz mumkin. BU g'oyani kod yozishdan boshlab o'yin shaklida kiritish g'oyasini oldinga surmoqchimiz.

Ushbu kitob bolalar uchun juda yaxshi o'rtacha og'irlikdagi Python dasturlash uchun qo'llanma. O'quvchilar Python tili haqida asosiy bilimlarga ega bo'ladilar, bu mavzuga yo'naltirilgan dasturlash to'g'risida ma'lumot olishadi, funktsiyalar, sinflar va modullar bilan ishlashni o'rganadilar. Grafika bilan ishlashga, animatsiya yaratishga va kitobda bolalarning o'z o'yinlarini rivojlantirishga katta e'tibor beriladi. Nashr dasturlashni yaxshi ko'radigan o'rta va yuqori sinf o'quvchilari uchun foydali bo'ladi, shuningdek, bolalar uchun qo'shimcha ta'lim kurslarida o'qituvchi sifatida ishlatilishi mumkin, dastlabki kodlarni terib, grafik namoyishlarni ko'rsatib yuqori guruh bog'cha bolalarini ham jalb qilish mumkin hamda bu

tarzdaagi informatikani rivojlantirib, uni o'qitilishini yanaada yoshartirish tendentsiyasini amalga oshirish g'oyasini talqin qilishni harakat qilindi.





MAZMUNI

IRISH

1. Dasturlashtirish haqida tushuncha.
2. Bolalarga dasturlashtirishni o'rgatishda nima maqsad
3. Qanday qilib bolalar dasturlashtirishga o'rganishlari mumkin.

I BOB. PYTHON ASOSLARI. MUHIT BILAN TANISHUV.

1. Pythonda dasturlash muhiti bilan tanishish.
2. Pythonda dasturlarni yozish.
3. Pythonda yozilgan dasturlarni ishga tushirish.
4. Dasturlashga doir masalalar.

II BOB. TOSHBAQA PERSONAJLI PYTHONDA RASM CHIZISH.

1. Birinchi turtle dasturi
2. Toshbaqaga doir dasturlar
3. Dasturlarga ranglar qo'shish.
4. Dasturlashga doir misollar.

III BOB. PYTHONDA MATEMATIKA VA SANASH.

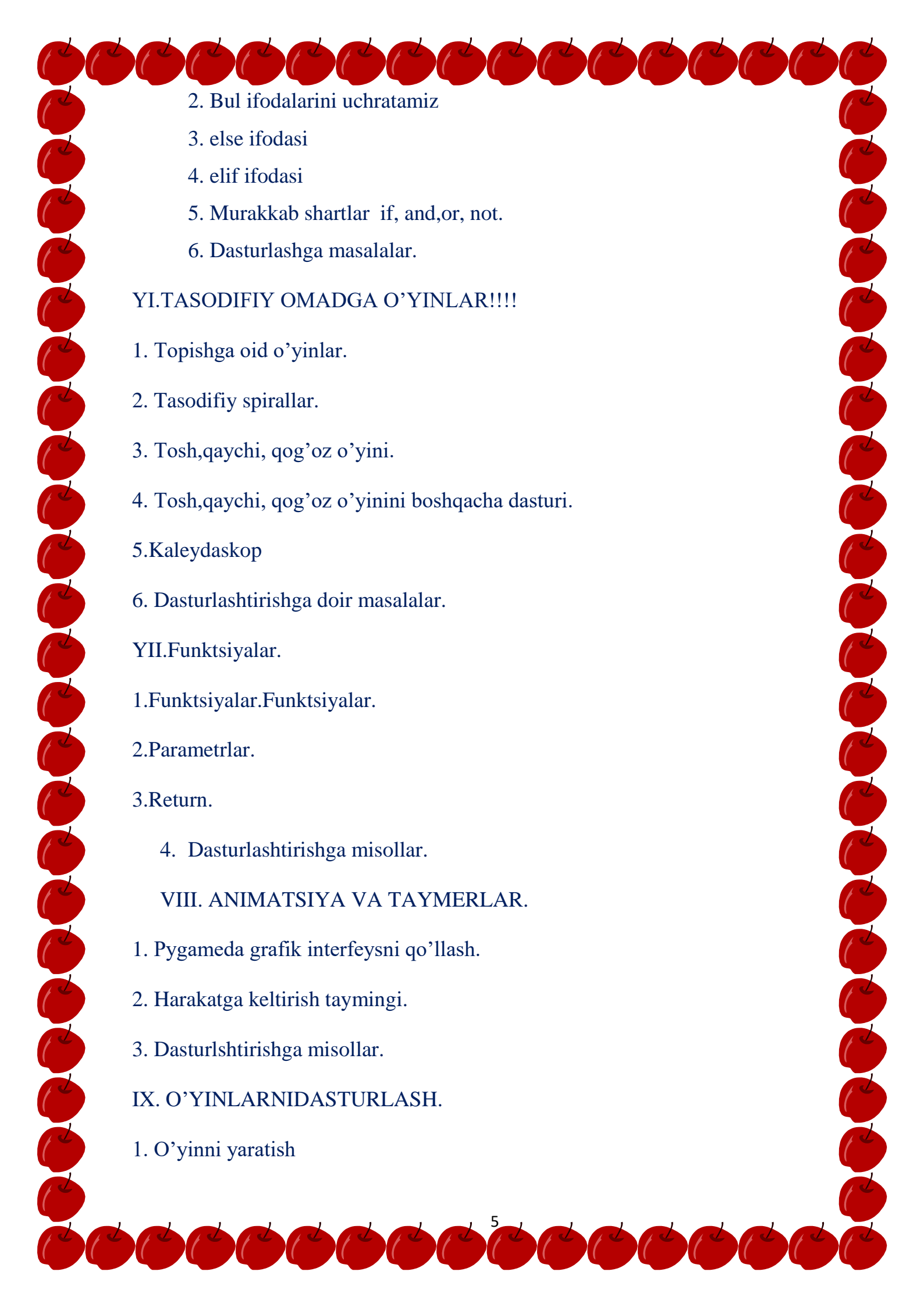
1. Pythonda berilganlar
2. Pythonda sonlar va matematika.
3. Pythonda satrlar va ro'yhatlar.
4. Dasturlashtirishga doir misollar.
5. Pythonda ro'yhatlar.
6. Dasturlashga misol va masalalar.

IY.PYTHONDA TAKRORLANISH.

1. Dasturingizni takrorlanuvchi operator for yordamida tuzing.
2. O'yin shaklida while bilan ishlash.
3. Spiral.
4. Dasturlashga misol va masalalar.

Y.SHARTLAR

1. if sharti.

- 
2. Bul ifodalarini uchratamiz
 3. else ifodasi
 4. elif ifodasi
 5. Murakkab shartlar if, and,or, not.
 6. Dasturlashga masalalar.

VI.TASODIFIY OMADGA O'YINLAR!!!!

1. Topishga oid o'yinlar.
2. Tasodifiy spirallar.
3. Tosh,qaychi, qog'oz o'yini.
4. Tosh,qaychi, qog'oz o'yinini boshqacha dasturi.
- 5.Kaleydaskop
6. Dasturlashtirishga doir masalalar.

VII.Funktsiyalar.

- 1.Funktsiyalar.Funktsiyalar.
- 2.Parametrlar.
- 3.Return.
4. Dasturlashtirishga misollar.

VIII. ANIMATSIYA VA TAYMERLAR.

1. Pygamedada grafik interfeysni qo'llash.
2. Harakatga keltirish taymingi.
3. Dasturlashtirishga misollar.

IX. O'YINLARNIDASTURLASH.

1. O'yinni yaratish

Pythonda dastur tuzamiz.

Algoritm

Sharik

Dasturlashtirishga masalalar.



KIRISH

ASTURLASHTIRISHHOZIRGIKUNDAENGZAMONAVIYSOHALARDANB
OLIB, UNGAHAMMAQIZIQADI.
BOLAGAYOSHLIGIDANDASTURLASHTIRISHBO'YICHA ,
KODTUZISHGADOIRKO'NIKMALArNIBERIBBORSAK,
LARNIKELAJAKDAINSONHAYOTINIUNUMLIVAQULAYDARAJADAKE
HISHIGATA'SIRQO'SHISHLARIGAERISHGANBO'LARDIKVAHAYOTIM
JZGAELEKTROMOBILLARVAENERGO-
ROBOTLARNIKIRIBKELISHINITEZLASHTIRAREDIK.
AMMANARSANIQIZIQARLIROQ,
AYTARILUVCHIJARAYONLARNIUNUMLIVAYAHSHIAMALGAOSHIRI
SHIMIZGAYORDAMBERADI,

BOLANIDO'STLARIVATEVARAKOLAMBILANMULOQOTINITA 'MINLARI
DI,
IJODQILISHVAYARATUVCHANLIKQOBILIYATINIAMALGAOSHIRADI.
DASTURLASHTIRISHHAQIDABILIMLARGAEGABOLALARJAMIYATNI,
FANVATEXNIKGAYANGILIKKIRITIBO'ZGARTIRISHGAHISSAQO'SHISH
ADI,
SMARTTEKNOLOGIYALARINIRIVOJLANISHIGAO'ZHISSASALALRINIQ
O'SHISHADI.



1. DASTURLASHTIRISHHAQIDATUSHUNCA.

Dasturlashtirish – eng zamonaviy kasblardan boʻlib uning fan va texnika hamda ishlab chiqarish bilan bogʻliqligi katta. Bolani kod yozishga oʻrgatish eng kerakli koʻnikmani bolada shakllantirishni talab etadi, bu uy vazifalarni qiziqarliroq bajarilishini, imkoni boricha qaytarilishini, doʻstlar va butun dunyo hamjamiyati bilan birgalikda dasturlashtirishni oʻrganishda faol ishtirok etishni talab etadi.

Bolalar oʻzgarayotgan dunyoga oʻz hissalarini qoʻshmoqchi boʻlsalar ular hoziroq dasturlashtirishni oʻrganishlari zarur.

2. Bolalarga dasturlashtirishni oʻrgatishdan nima maqsad

Bolalar bizning kelajagimiz va ularning bilimlari qanday boʻlishiga fan va texnikaning, dasturlashtirish qobiliyatlariga Smart olamining, dasturiy texnikaviy yozmalarning, inson hayotining shinam va qulay, icham va kompakt shakllanishi rivojlanishi toʻgʻri proporsionaldir. Dasturlashtirish bolni oʻz ustida ishlashga hamda ohirda natija olishga oʻrgatadi. Bilimlarinin oshirishga oʻz hissasini qoʻshadi. Eng oliy maqsadlardan biri, bolani bugungi kundgi oʻz doirasidagi dasturlashtirishga hiss qoʻshishi asta-asta oʻz ustida ishlab, shu yoʻnalishning yirik mutahassisi boʻlishiga oʻz hissasini qoʻshadi, u kelajakda oʻzgarayotgan olamga zarur oʻzgartirishlar kiritishga erishadi, yani oʻzgarish boʻlishi lozim sohalarda kashfiyotlar, yirik narsalar oʻylab chiqilishiga oʻz hissasini qoʻshadi.

2. Qanday qilib bolalar dasturlashtirishga oʻrganishlari mumkin

Bolalarga dasturlashtirish boʻyicha asosiy bilimlar berib, ularni oʻzgaruvchilar, dasturlash muhiti, satrlar, roʻyhatlar, funktsiyalar va grafika bilan tanishtirish, bu sohada koʻnikma va malakalarini toʻldirib boorish kerak boʻladi. Bollar Python dasturlashtirish tili oʻzgaaruvchilari va ularning turlari, dasturlashtirish strukturasi, oddiy strukturalardan boshlab, murakkabroq strukturalargacha asta-asta bilim olishlari kerak boʻladi.

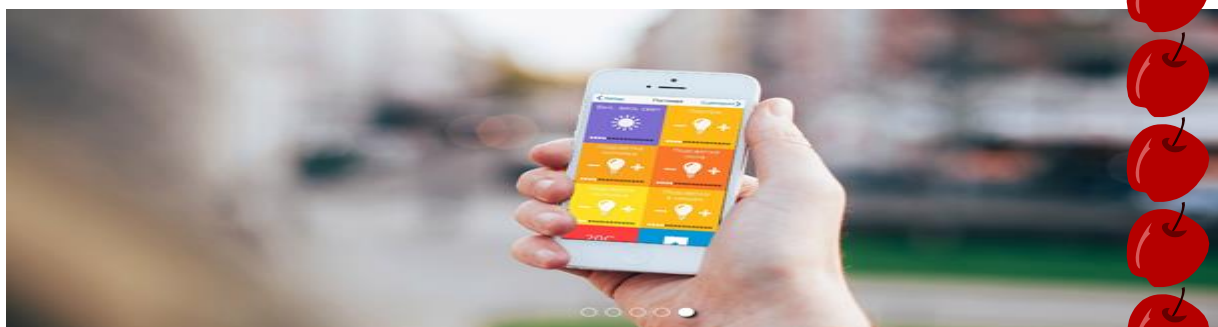
Bolalarni albatta dasturlashtirishga oʻrgatish oson emas deb oʻylarsiz, chunki bu sohada ham jiddiy va oson jarayonlardan emas albatta, lekin bu jarayonga erishish uchun bolalarga dasturlashtirishni oʻyin shaklda oʻrgatish mumkin va bunda bola

o'yin o'ynamay, balki avval o'yinni o'zi oylab topib, uni ko'z oldiga keltirib dastur kodini yozishi kerak bo'lad, buning uchun esa u o'zgaruvchilar, nima dasturda berilgan, nimani toppish kerakligini anglab, dasturlash tilining operatorlarini nafaqat yozishna va ularni qanday qo'llashni bilishlari kerak desak mubolag bo'lmaydi.

I BOB. PYTHON ASOSLARI. MUHIT BILAN TANISHUV. Python dasturlash muhiti bilan tanishish.

Hamma narsaning ichida dastur bo'lishi mumkin, bu telefon, kompyuter, sovitgich bo'lsin, masalan, robot, tabrik uchun otkritka, chang yutgizh, sanoat qurilmasi, yoki tibbiyot uskunasi, ta'lim jarayonida qo'llaniladigan teleeshituvchi yoki kuzatuvchi robot va boshqalar bo'lishi mumkin. Qurilmalarning ichida dasturlar bo'lib, ular qurilma vazifasini yanada qulay va ishlashini odam uchun yanada samarali amalga oshirishga yo'naltiriladi. Bunday dasturlarni ilovalar deyiladi, ular biror realistik joyda o'yin ko'rinishidagi jangni namoyish qilishi zarur bo'ladi yoki ular biror manzarani yoki grafik tasvirni chizishga mo'ljallanishi mumkin yoki onlayn rejimda biror narsani boshqarishni o'z ichiga olishi mumkin, masalan pult orqali boshqariladigan Smart uyni yoki honadonni.

Rasm 1. SMART TEXNOLOGIYALAR





asmdan ko'rinib turibdiki, ehm dasturi tufayli uy yoki honadonni signalizatsiya, o'chirish tizimini, hafsizlik tiziminin, gaz honadagi uskunalar yonish va o'chishini, suv, gaz yonish va o'chishini, jalyuzilar ochilish va yopilishini boshqarish mumkin. Dastur tufayli uy o'z egasinin masofada turib boshqariladi, mashinada kelishigagaraj eshiklari ochiladi.

SMART / SMARTER — bu belgilash uchun nom bo'lib, maqsadlarni aniqlash va boshqarish uchun nom bo'lib, menejment va proekt boshqaruvida qo'llaniladigan mnemonic abreviaturadir. 6y Birinchi bu termini Pop Meyra (Paul J Meyer) ishida 1965 yilda qo'llaniladigan va keyinchalik noyabrda 1981 yilda Management Review by George T. Doran, y'ani Georgiy Doranning maqolasida qo'llanilgan.^[1]

SMART aqilli maqsad degan ma'noga ega bo'lib, ingliz harflarining bosh harflarining bosh harflarini birlashtirishdan iborat, haqiqiy maqsad qanday bo'lishi kerakligini aniqlaydi: *Specific (aniqlik) — Measurable (o'lchanadigan) — Attainable (erishuvchanlik) — Relevant (Relevantlik) — Time-bounded (Vaqtda cheklilik)*



Rasm 2. Smart so'zining abreviaturaning ma'nosi, so'zlardan iboratligi.

Shunday ekan biz dasturlashtirishning biror muhiti bilan tanishib, unda ilk bo'lib birinchi dasturlarni terish haqida o'ylashimiz zarur.



Python – bu dasturlashtirish tili bo'lib, u 1960 yillarda paydo bo'lgan va uning nomlanishi o'sha vaqtlarda juda mashxur bo'lgan “Monti Payton” komediyasi guruhini nomi bilan bog'liq, u o'z chiqishlarida Brayin ismli kahramon haqida gapirgan bo'lgan, Pythonda algoritmi shakllangan insonga dasturlashtirishni o'rganish oson bo'lgan, Python juda mashxur dasturlardan biri bo'lib qoldi. Dasturlashtirish-bu kompyuterga tanish bo'lgan tilda kompyuter bajarishi kerak bo'lgan buyruqlar ketma-ketligi. Dasturlashtirish tilida siz yozgan dastur ba'zan hato ko'rsatishi mumkin. Bunda ushkunlikga tushmay uning yozilishiga e'tibor bering, syntaxis hatolarni to'g'irlang, Pthon ham sintaksis va orfografik hatolarni yoqtirmaydi. Ma'lumki, maktab to'liq dasturlashtirishni o'rgata olmaydi, bu soha bo'yicha boshlang'ich tushunchalar bera oladi va bu jarayonni ham hamma maktablarda uddasidan chiqa olmaydi. Dasturlashtirishdan malakaga ega bo'lish-bu kelajak

o'nikmasi, bugun qanchalik yozish va o'qish hamma uchun mumkin bo'lsa, shunchalik dastrashtirish ham kelajakda yozish va o'qish bilan bir qatorda bo'ladi. Bugungi kunda kimdir dasturlashtirishga qushimcha ta'lim sifatida kurslarda yoki maktablarda o'qiyapti, kimnidir dasturchi ota-oansi va qarindoshlari bo'lsa, lekin hohish va istak bo'lsa bu jarayonni mustaqil uddalash mumkin. Mahsus kitoblar orqali va onlayn kurslar yordamida ham dasturlashtirishni o'rganish mumkin.

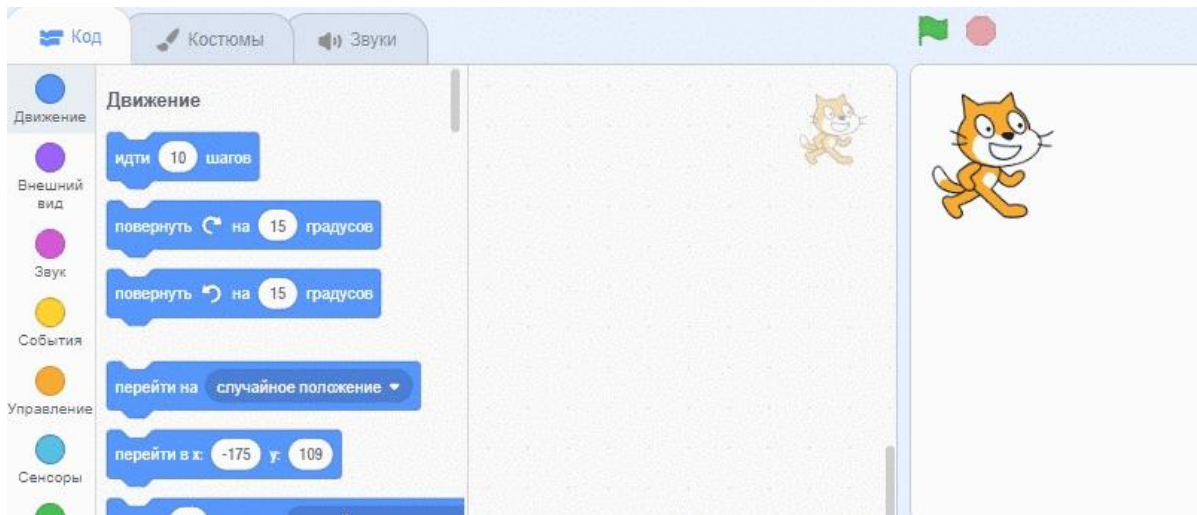


Python (payton yoki piton ПИТОН) – murakkab bo'lmagan til. Undan ham osonroq Scratch mavjud, u maktabgacha va boshlang'ichdagi bolalar uchun tavsiya etiladi. Scratch bo'yicha ham bolalarga kerakli adabiyotlarni taqdim etish kerak. Hozirgi kunda bolalaringizni olimpiada va boshqa konkurslarda qatnashishini hohlasangiz, ularni Python dasturlashtirishni bilishlarini amalga oshirishingiz kerak.

Python bilan tanishtirishning o'ziga hos uslubini taklif qilaman:

Hozirgi kunda ko'p bolalar daturlashtirish olami bilan scratch.mit.edu platformada proektlarni yaratish orqali tanishishadi. Proektlarni yaratish rangli bloklarni topib quyish orqali amalga oshiriladi, bolalar proektni klaviraturadan hech narsa kiritmay yaratishadi. Blalar katta bo'ladi, ulg'ayadi va ulaga Scratch olami toliq qiladi va bunda ularni biz Pythonga o'tkazishimiz kerak bo'ladi. Scratchda ishlab bolalarning bir qismi rassomchilkg, bir qismi esa professional

o'yinlarni Constructda amalga oshirishadi, dasturlashtirishni yahshi ko'rganlarga esa Pythonni tavsiya qilish mumkin. Boshlang'ich sinfo'quvchilari umuman klaviatura bilan tanish bo'lishmasa nima qilish kerak! Mana Scratchdagi proyektida bola faqat ikkita sonni klaviaturadan kiritish bilan cheklanadi.



“Yosho kttaroq” dasturlashtirish tillarida ko'proq narsani klaviaturadan kiritishni talab etiladi, klaviaturadan ko'proq matnni kiritish zarur bo'ladi, bolalarni Scratchdan Pythonga o'tish yo'lini ham soddalashtirish mumkin, bunda ham yo'riq bor. Во «взрослых» языках программирования приходится вводить гораздо больше текста с клавиатуры, но есть способ упростить детям жизнь при переходе от Scratch к Python.



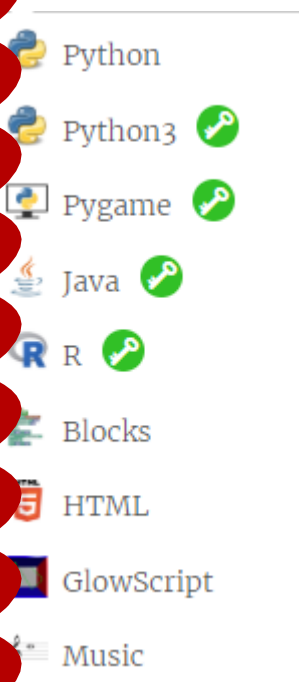
NIMA UCHUN AYNAN Python?

Kod uzun bo'lsa yuqori sinf o'quvchilari tez kiritishadi, lekin quyi sinf

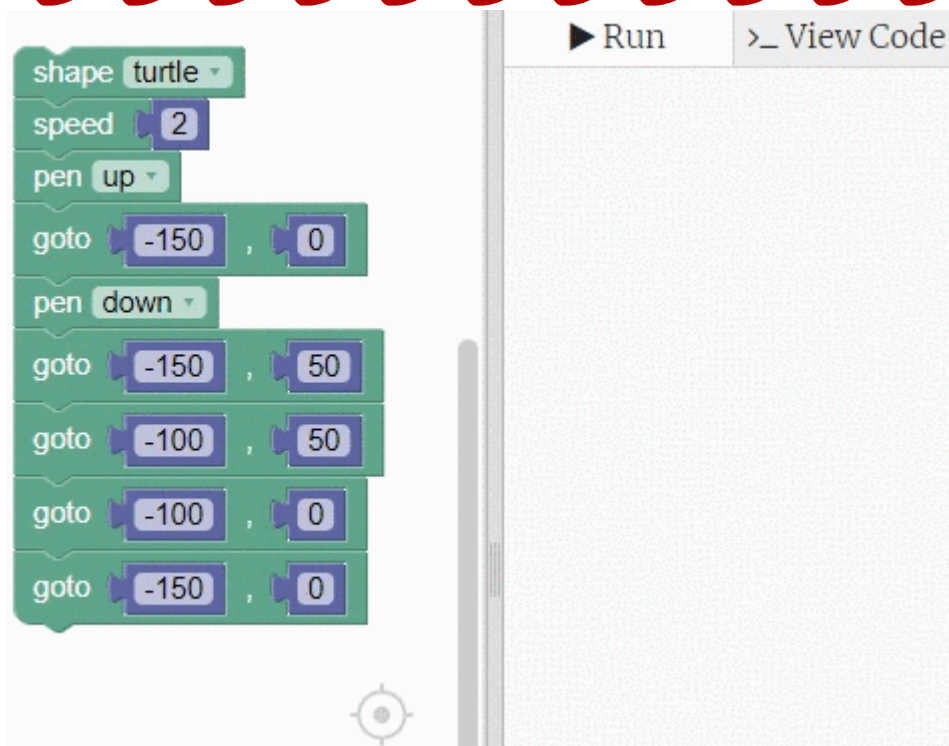
qovchilari uchun bu muammo bo'lishi mumkin. Trinket.io platformasi orqali Python bilan bolalarni tanishtirish mumkin. Bu platformada online rejimida quyidagi asusturlashtirish tillari bilan ishlash mumkin:

- Blockly Blokli til
- HTML
- GlowScript
- R
- Java
- Va albatta Python

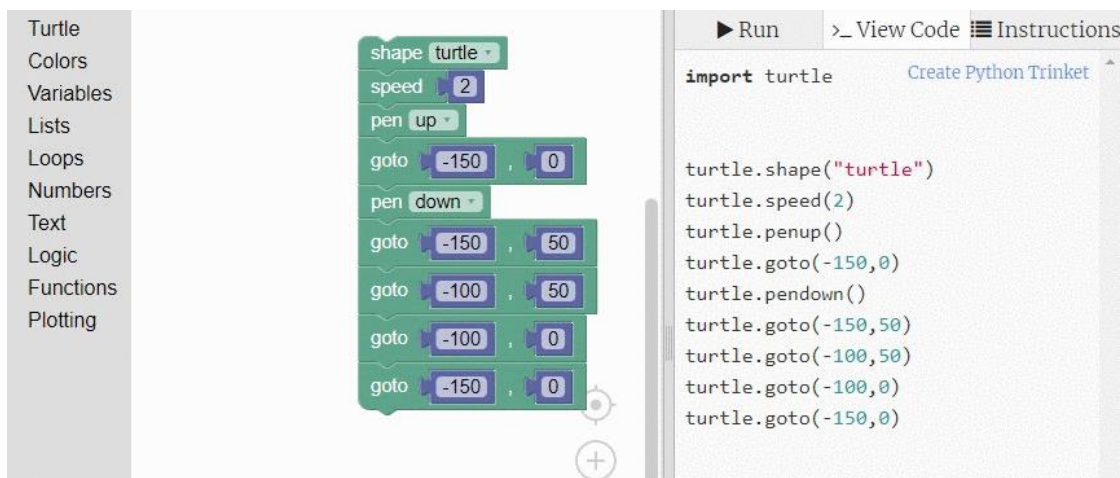
ullik versiyada (\$3 oyiga) Pygamedan o'yinlarni yaratish uchun foydalanish mumkin!



Blokli trinket uchta klikda Pythondagi kodga aylanishi eng buyuk yangilik bo'lib, birinchi klikda-kodni ko'rish.



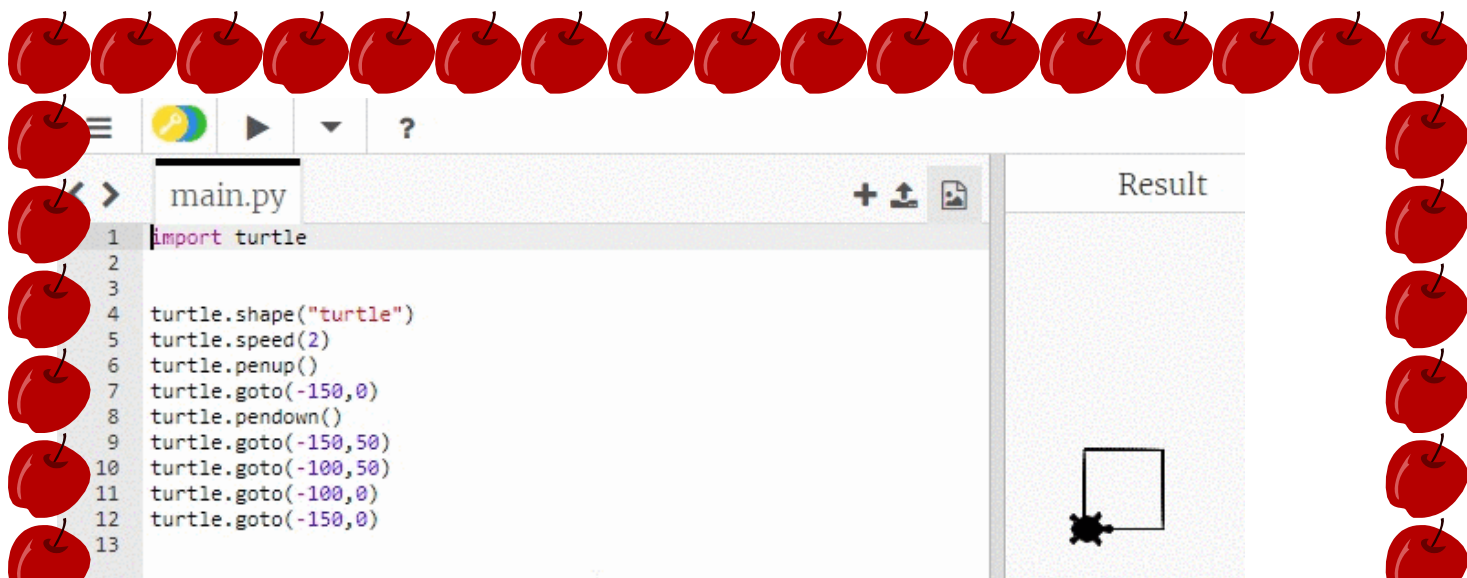
Yana kliklarning jufti —blokli trinketni Pythondagi trinketga almashinuvini



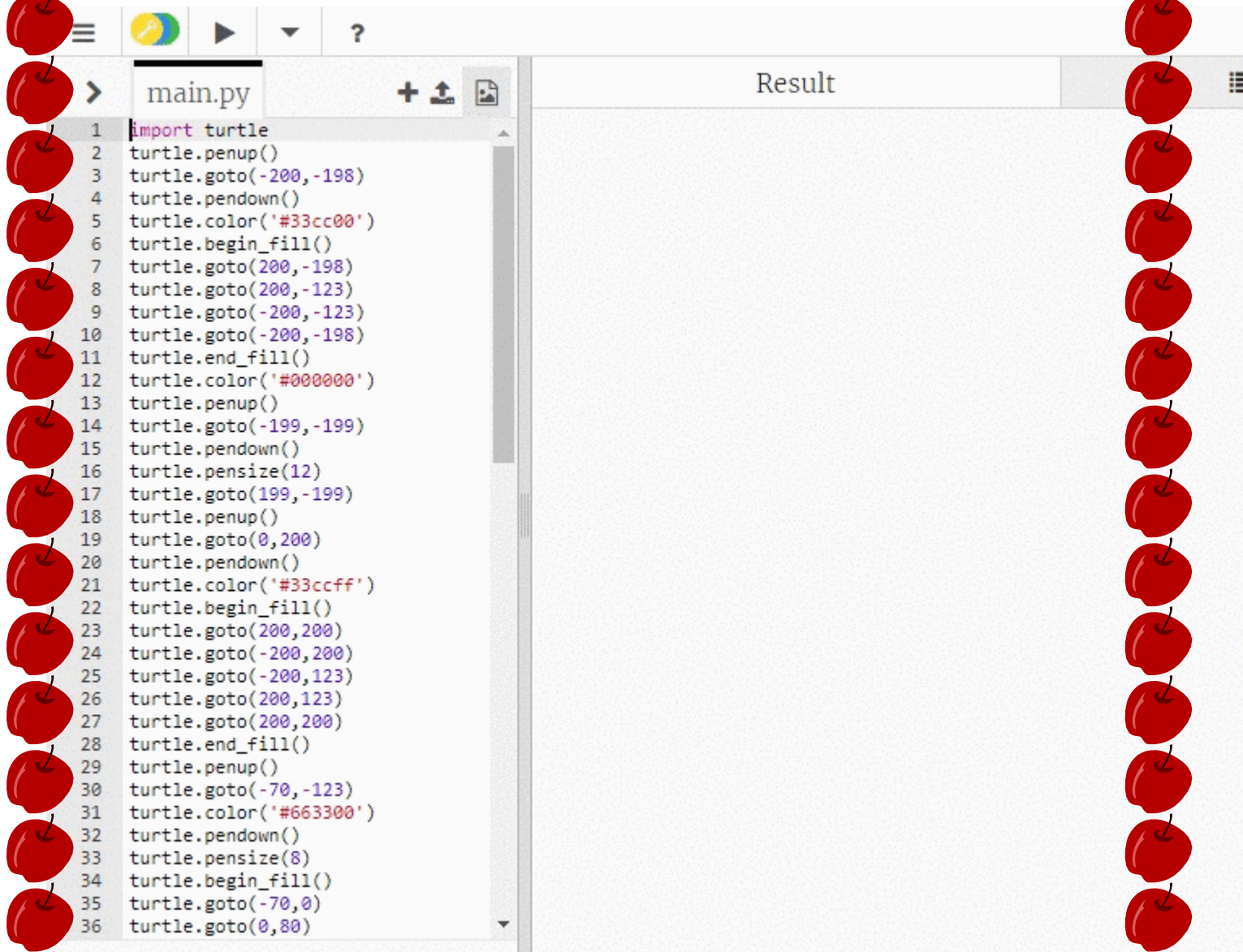
Tayyor

bo'ldi

Bola dasturni Pythonda hosil qiladi, u unga klaviaturada qiynalib klavishalarni izlab, topib kiritishdan halos bo'ldi, endi o'rgatuvchi blok til bilan analogiyalar amalga oshirishi, qolaversa, sintaksis haqida so'zlab berishi va koordinatalar o'zgartirish bo'yicha vazifalar berishi mumkin, koordinatalarni o'zgartirish vazifa berishi va kvadratni to'rtburchakga aylantirishni aytishi mumkin.



Bola keyinchlik murakkabroq proektni amalga oshirishi mumkin bo'ladi. Masalan
bu quyidagi rasm chizishning proektni.



Shunday qilib boshlang'ich sinf bolalariga biz Python bo'yicha saboqlarni berishga o'tishni tavsiya etamiz. Ular bina 45 minut ishlanadi va murakkab proektlan tuzilmaydi. Scratchda o'rganilgan shart va tkrorlanishlarga yangicha yondashish yordam beradi, oldin o'ylanmagan sonli va matnli o'zgaruvchilar to'g'risida o'ylanamiz. Murakkab berilganlar tiplari bilan tanishamiz, kesib olish va boshqa uslublardan foydalaanamiz. Bolalarni qiziqtirish uchun toshbaqa bilan chizish vazzifalaridan tashqari ASCII kodidan foydalanib chizishni amalga oshirmoqchimiz. Kaomodzi smaylilarinin chizamiz va yunikod simvollaridan rasmlarni hosil qilamiz.

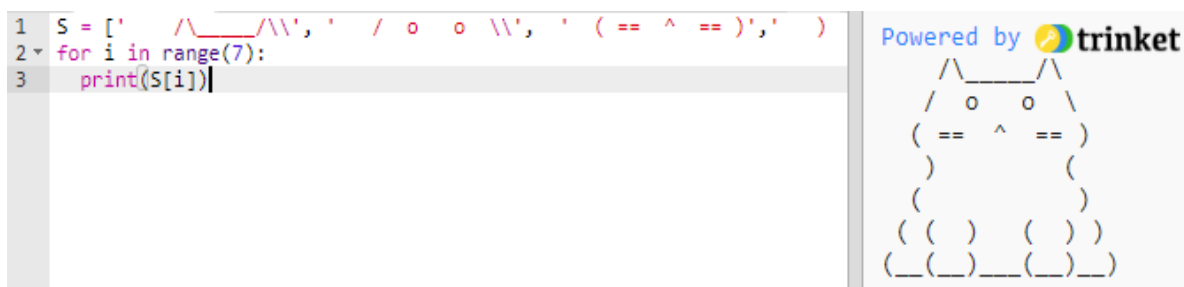
Ro'yhatli


mushikcha

```

1 s = [' / \_ / \', ' / o o \', ' ( = ^ = )', ' )
2 for i in range(7):
3     print(s[i])

```



Powered by  trinket

```

 / \_ / \
 / o o \
 ( = ^ = )
 )
 ( ( ) ( ) )
 ( ( ) _ ( ) )

```

Ikkita

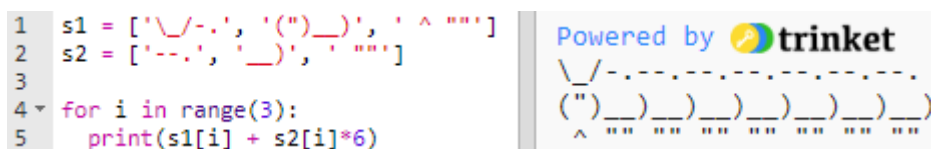
ro'yhatli


qu

```

1 s1 = [' \ / - .', '( " ) _', ' ^ ""']
2 s2 = [' - .', ' _', ' ""']
3
4 for i in range(3):
5     print(s1[i] + s2[i]*6)

```



Powered by  trinket

```

 \ / - .
 ( " ) _
 ^ ""

```

Yunikod

lugg'ati

belgalaridan

ras

```

1 d = {'человек':unicr(50743), 'Солнце':unicr(9788),
2     'домик':unicr(49826),'мяч':unicr(9917),'цветок':unicr(9883)}
3
4 s1=[unicr(9556), unichr(9552)*23, unichr(9559)]
5 s2=[unicr(9553), ' '*4, ' ', d['Солнце'], '\t', '\t', unichr(9553)]
6 s3=[unicr(9553), unichr(32)*23, unichr(9553)]
7 s4=[unicr(9553), d['человек'],d['мяч'], '\t', '\t', '\t', unichr(9553)]
8 s5=[unicr(9553), d['цветок']*13, d['домик'], '\t', unichr(9553)]
9 s6=[unicr(9553), d['цветок']*14, '\t', unichr(9553)]
10 s7=[unicr(9553), d['цветок']*14, '\t', unichr(9553)]
11 s8=[unicr(9562), unichr(9552)*23, unichr(9565)]

```



Ob'ektlar

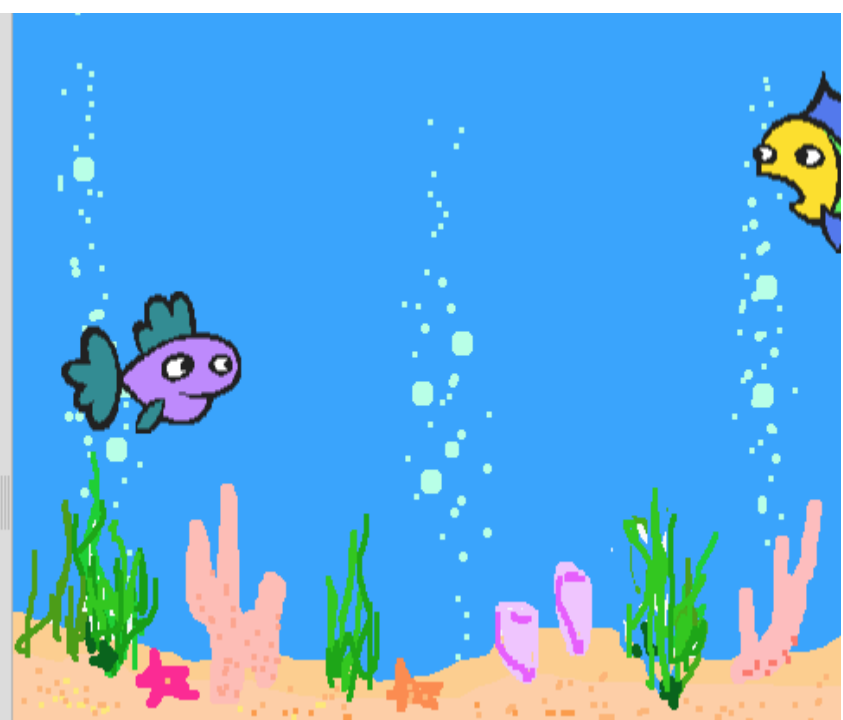
bian

ishlash.

```

1 import turtle
2
3
4
5 w=turtle.Screen()
6 w.setup(480,360)
7 w.bgpic('underwater.gif')
8
9
10 fish1=turtle.Turtle()
11 fish1.penup()
12 image0='fish5.gif'
13 w.addshape(image0)
14 fish1.shape(image0)
15 fish1.speed(0)
16 fish1.left(90)
17 fish1.goto(-240,0)
18
19
20 fish2=turtle.Turtle()
21 fish2.penup()
22 image1='fish4.gif'
23 w.addshape(image1)
24 fish2.shape(image1)
25 fish2.speed(0)
26 fish2.left(90)
27 fish2.goto(240,100)

```



Diziqarli

o'yin

«adashtirma»

—

oddiy,

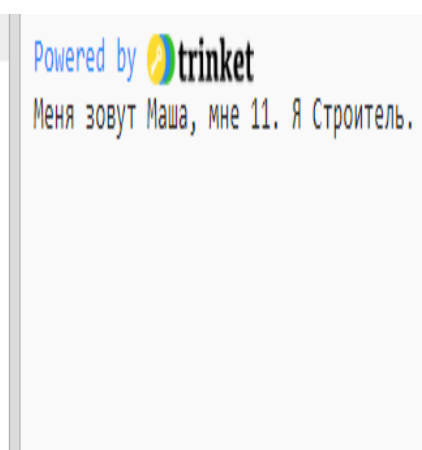
juda

quvnoq!

```

1 import random
2
3 Name = ['Маша', 'Миша', 'Ашот', 'Вахтанг', 'Элла']
4 Job = ['Строитель', 'Школьник', 'Грузчик', 'Двоечник', 'Балерина', 'Таксист']
5 Age = str(random.randint(5, 28))
6 Text = ['Меня зовут ', ' ', 'мне ', ' ', ' ', 'Я ', ' ']
7 Rnd_name = Name[random.randint(0, 4)]
8 Rnd_job = Job[random.randint(0, 5)]
9 print(Text[0] + Rnd_name + Text[1] + Age + Text[2] + Text[3] + Rnd_job+ Text[4])

```



trinket.io platforma —Python dasturlashtirish tili bilan bollarni o'rgatish uchun eng qulay yordamchi. Lekin undan ham zo'r platforma bo'lib, u o'z navbatida Python dasturlashtirishni o'rgatishga mo'ljallangan va Minecraft olami dasturlashtirishni bizga taklif etadi. Bu platformning nomi progkids.com bo'lib yaqinda uning mualliflari Facebookda promokod FREESELF COURSE ni o'rnatishdi. Bu platforma uchun juda ko'p darslar mana bu yerga ishlatilgan qo'yilgan: codim.online/pythonvm, ular ichma ich joylashgan tsikllar, ko'p o'lchovli massivlar haqida tushuncha bera oladi.

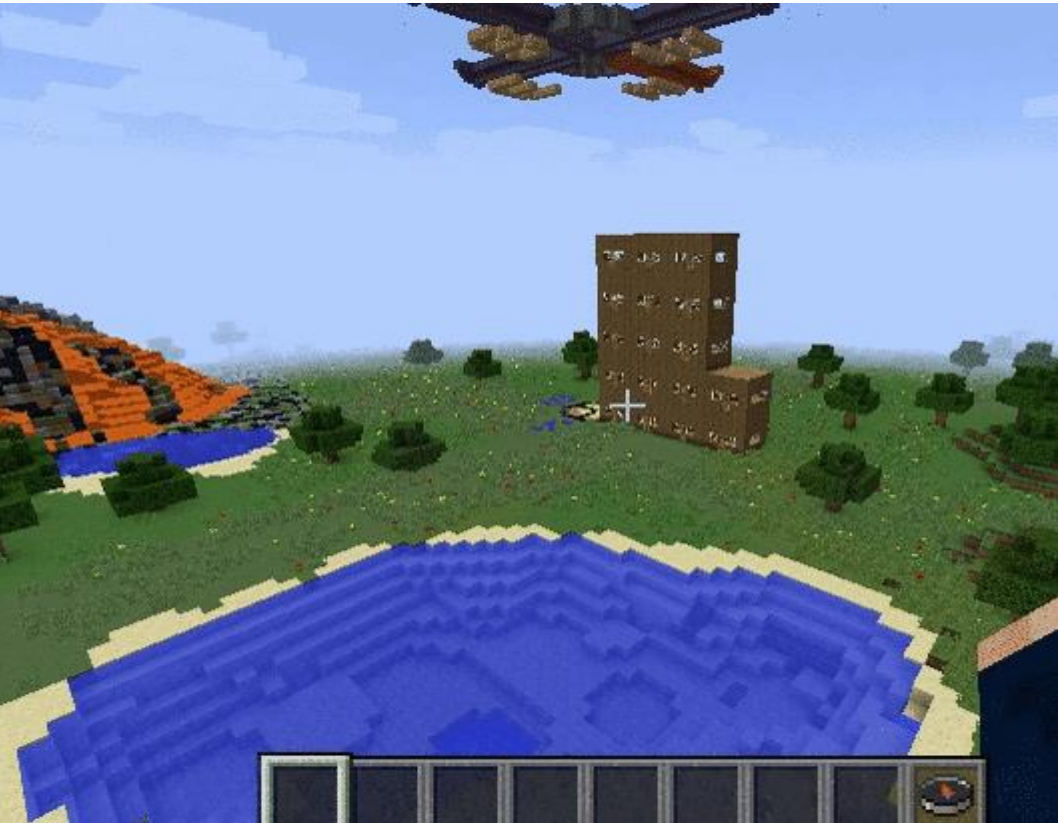
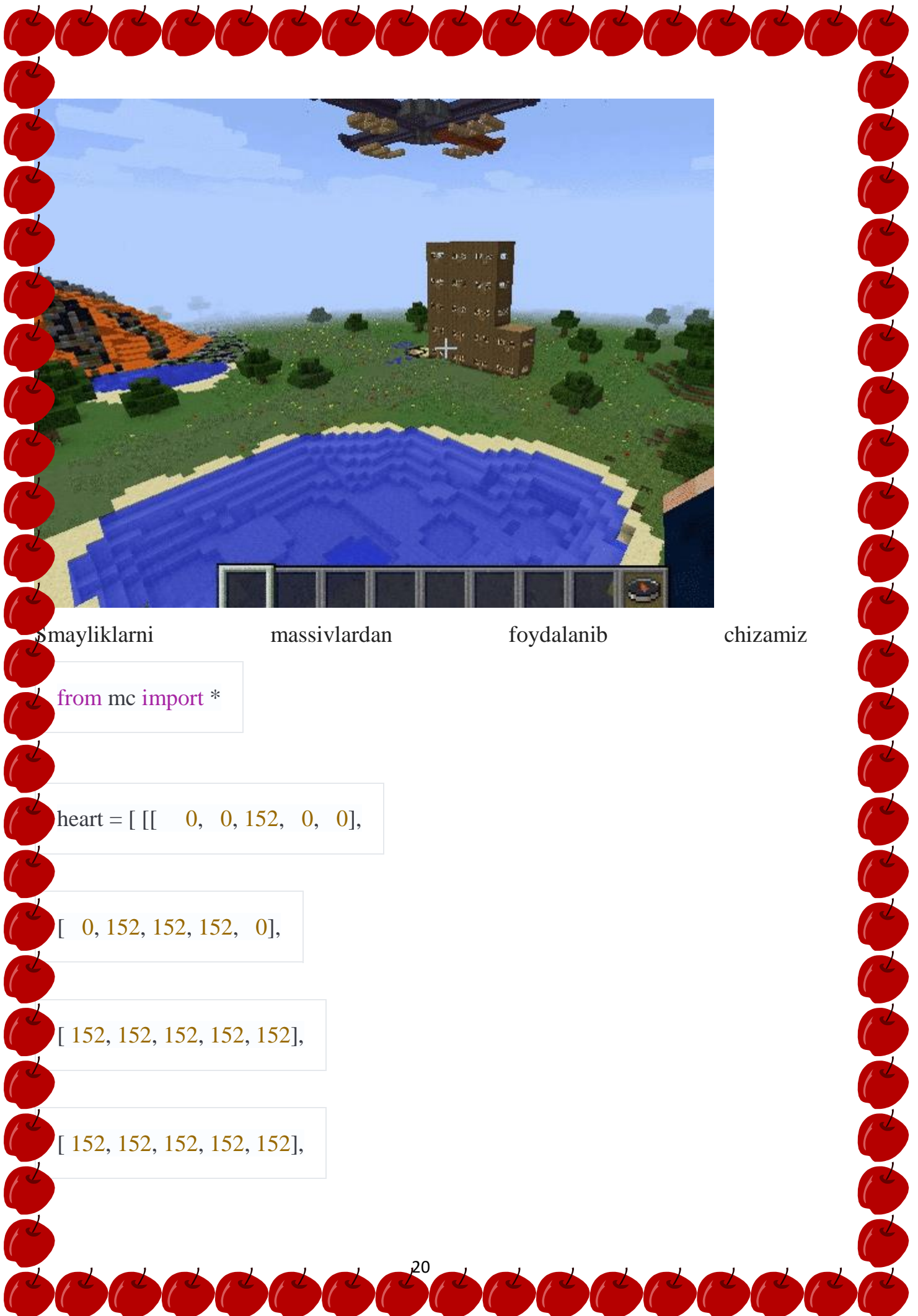
```
from mc import world, blocks, player
```

```
for i in range(5):
```

```
for j in range(5):
```

```
for k in range(5):
```

```
world.buildHome(5 + (j * 5), 0 + (k * 5), 0 + (i * 10), 5, 5, 5, blocks.WOOD)
```



smayliklarni

massivlardan

foydalanib

chizamiz

```
from mc import *
```

```
heart = [ [ [ 0, 0, 152, 0, 0],
```

```
[ 0, 152, 152, 152, 0],
```

```
[ 152, 152, 152, 152, 152],
```

```
[ 152, 152, 152, 152, 152],
```

```
[ 0, 152, 0, 152, 0] ]
```

```
scull = [[ [ 0, 173, 173, 173, 0],
```

```
[ 0, 173, 173, 173, 0],
```

```
[ 173, 173, 173, 173, 173],
```

```
[ 173, 0, 173, 0, 173],
```

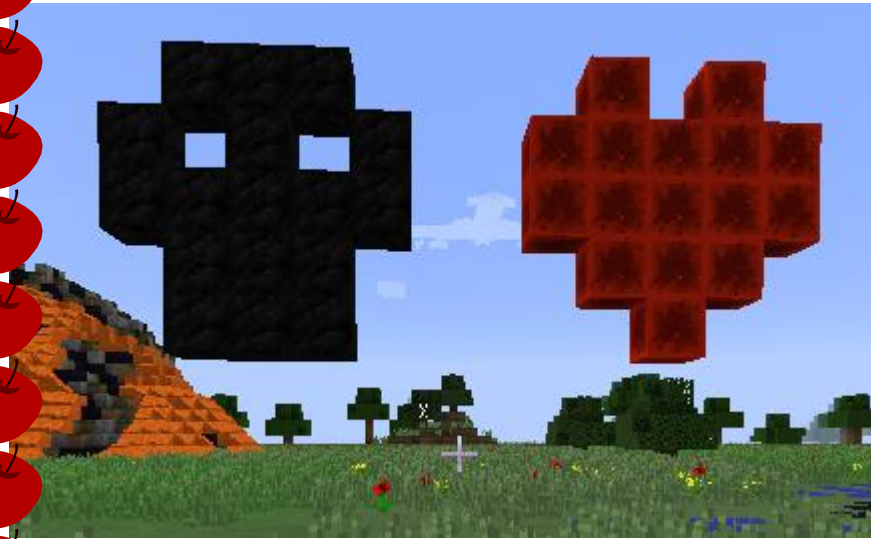
```
[ 0, 173, 173, 173, 0] ]
```

```
world.setBlocks(0, 3, 0, sad)
```

```
world.setBlocks(0, 3, 7, heart)
```

```
world.setBlocks(0, 3, 14, scull)
```

```
world.setBlocks(0, 3, -7, butterfly)
```



platformajudako'pnarsalagaboy,

arilmalarniyartilishiniosonlashtiruvchiustqurmaldaniborat,

arob'ektlarnisolishtirshni, dronlarniboshqarishni, shaxarlarnibarpoetishni ,

metroiqazishni,

onnellarniqurishnivayanajudahamzo'rnarsalarniqurishniamalqashiradi!!!!

<p><code>setCuboid(x0, y0, z0, x, y, z, type)</code> Заполняет пространство от x0, y0, z0 до x, y, z блоком type. Возвращает True, если успешно</p> <p><code>setBlocks(x, y, z, list)</code> Строит куб из блоков в трёхмерном массиве list начиная с x, y, z. Возвращает True, если успешно</p> <p><code>buildArc(x, y, z, w, h, type)</code> Строит арку из блока с ID type, высотой h, шириной w в позиции x, y, z. Возвращает True, если успешно</p> <p><code>buildColumn(x, y, z, h, type)</code> Строит колонну из блока с ID type, высотой h в позиции x, y, z. Возвращает True, если успешно</p> <p><code>buildSphere(x, y, z, r, type)</code> localization error at gui.toolbox.world.buildSphere</p> <p><code>buildHome(x, y, z, w, l, h, type)</code> Строит дом из блока с ID type, высотой h, длиной l, шириной w в позиции x, y, z. Возвращает True, если успешно</p>	<p><code>getDirection(entity)</code> Возвращает единичный вектор, указывающий в направлении взгляда существа</p> <p><code>getPitch(entity)</code> Возвращает угол направления взгляда создания относительно параллельному земле Число с плавающей точкой от -90 до 90.</p> <p><code>getPos(entity)</code> Получает текущую позицию создания Возвращает массив из трех элементов x, y, z.</p> <p><code>getRotation(entity)</code> Возвращает угол поворота создания относительно юга(?) Число с плавающей точкой от 0 до 360</p> <p><code>setPos(entity, x, y, z)</code> Телепортировать создания в указанные координаты Возвращает True, если успешно</p> <p><code>setVelocity(entity, x, y, z)</code> Задаёт вектор ускорения сущности</p>	<p><code>moveForward(distance)</code> Вызывается от объекта дрона. Двигает дрона по направлению взгляда на distance блоков. Возвращает true или false в зависимости от результата.</p> <p><code>move(direction, distance)</code> Вызывается от объекта дрона. Двигает дрона в направлении direction на distance блоков. Возвращает true или false в зависимости от результата.</p> <p><code>lookTo(direction)</code> Вызывается от объекта дрона. Поворачивает взгляд дрона в направлении direction. Возвращает true или false в зависимости от результата.</p> <p><code>turn(pitch, yaw)</code> Вызывается от объекта дрона. Поворачивает взгляд дрона на pitch градусов по горизонтали и yaw градусов по вертикали. Возвращает true или false в зависимости от результата.</p>
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

animatsiyalanganqurilishlarnihamayaratishmumkin.



Ushbu platformalarni bolalarga namoyish eting, ular o'qituvchilar va ota-onalarni xursand qilish uchun ko'plab qiziqarli loyihalarni bajara olishlariga aminman.

Shuningdek, men barcha bolalarni haftalik Python marafonlariga taklif qilaman, shunda ular o'qituvchi nazorati ostida ushbu dasturlash tili bilan tanishishlari va Google Drive-da ko'nikmalarga ega bo'lishlari uchun bonus olishlari mumkin. codim.online/marafon



Bu platformalarni bolalarga ko'rsating, ular juda qiziq loyihlarni o'qituvchi va ota-

bolalarga bajarishlari mumkin. Bir haftalik marafonlarda o'qituvchi nazoratida ishlashlari mumkin va google Diskda ishlashga kunikmalar olishlari mumkin: codim.online/marafon. Minecraftni o'zida ham bolalar ayniqsa maktabgacha yoshdagi bolalar va maktab yoshidagi bolalar ham o'yin o'ynashlari mumkin. Minecraftda nafqat o'ynash mumkin, balki unda datur ham tuzish mumkinligini bolalarga o'rgatish kerak.

2. Pythonda dasturlarni yozish.

Python - bu oson o'rganiladigan dasturlash tili, ayniqsa yangi boshlanuvchilar uchun juda yaxshi. Asosiysi, unga ko'p vaqt sarf qilmasdan sodda va samarali dasturlarni yozishingiz mumkin.

U ko'pgina boshqa tillardan farqli o'laroq, Python kodini o'qish oson va interaktiv qobiq dasturlarni kiritishga va darhol natijalarga erishishga imkon beradi. Oddiy til tuzilishi va interaktiv qobiqdan tashqari Pythonda o'rganishni sezilarli darajada soddalashtiradigan va video o'yinlaringiz uchun oddiy animatsiyalar yaratishga imkon beradigan vositalar mavjud.

Python dasturlash tilida masala va yechimlar.

Python dasturlash tili obektga yo'naltirilgan til bo'lib, 1991 yil 20-fevralda Python Software Foundation nomli notijorat tashkilot va Gvido van Rossum tomonidan ishlab chiqilgan.

Python dasturlash tili dastur yaratish samaradorligi va kodlarning o'qilishi qulayligi yuqori darajada bo'lgan dasturlash tili hisoblanadi. Bundan tashqari standart kutubxonada foydali funktsiyalarning katta to'plami mavjud. Python nomi sudralib yuruvchilar oilasining nomidan kelib chiqmagan. Muallif ushbu til nomini 1970 yillarda Britaniyada mashhur bo'lgan "[Летающий цирк Монти Пайтона](#)" nomli tele-shoudan ilhomlanib qo'llagan.

Python dasturlash tilida yozilgan dasturni 2 usulda ishga tushirish mumkin: interaktiv interpretator yordamida va dastur matni yozilgan fayldan foydalangan holda.

1-usul. Linux distributivi terminaliga **python3** buyrug'ini yozib, ishga tushiramiz. Buyruq ishga tushgandan so'ng siz satr boshida `>>>` belgisi paydo bo'ladi. Bu yerda biror operatori yozish mumkin va bu **python interpretatorining buyruqlar qatori** deb ataladi.


2-usul. Biz biror dasturni ishga tushirmoqchi bo'lsak interpretatorga dasturni har safar qayta yozishimizga to'g'ri keladi. Shuning uchun yozilgan dastur kodini biror faylga yozib saqlab qo'yish dasturni istalgan vaqtda ishga tushirish imkonini beradi. Bu usulda bizga biror redaktor kerak bo'ladi va faylning kengaytmasi **.py** bo'lishiga ahamiyat berish kerak.

Yuza va perimetrni hisoblash.

(To'g'ri burchakli uchburchak misolida)

To'g'ri burchakli uchburchakning yuzini va perimetrini hisoblash uchun katetlar uzunligi berilgan. Pifagor teoremasi ($c^2 = a^2 + b^2$) orqali gipatenuza uzunligi aniqlaymiz.

Python dasturlash tilida sonni ildizdan chiqarish uchun **math** modulining **sqrt()** funksiyasidan foydalanamiz.



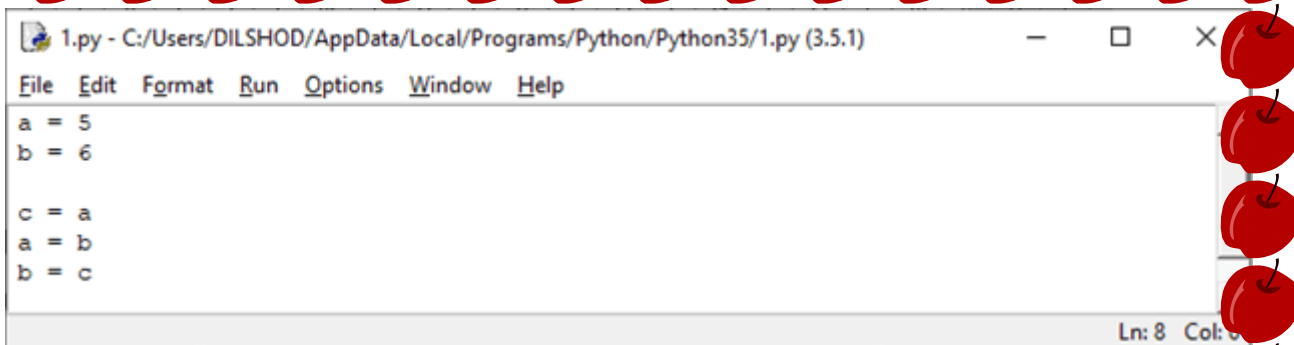
```
*Yuza.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/Yuza.py (3.5.1)*
File Edit Format Run Options Window Help
import math
AB = input("Birinchi katet uzunligini kiriting: ")
C = input("Ikkinchi katet uzunligini kiriting: ")
AB = float(AB)
C = float(AC)
BC = math.sqrt(AB**2 + AC**2)
S = (AB * AC) / 2
P = AB + AC + BC
print("Uchburchak yizi: %.2f" % S)
print("Uchburchak perimetri: %.2f" % P)
Ln: 16 Col: 0
```

Natija:

```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
Python 3.5.1 (v3.5.1:37a07cee5969, Dec 6 2015, 01:54:25) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/Yuza.py ==
Birinchi katet uzunligini kiriting: 5
Ikkinchi katet uzunligini kiriting: 4
Uchburchak yizi: 10.00
Uchburchak perimetri: 15.40
> |
Ln: 9 Col: 4
```

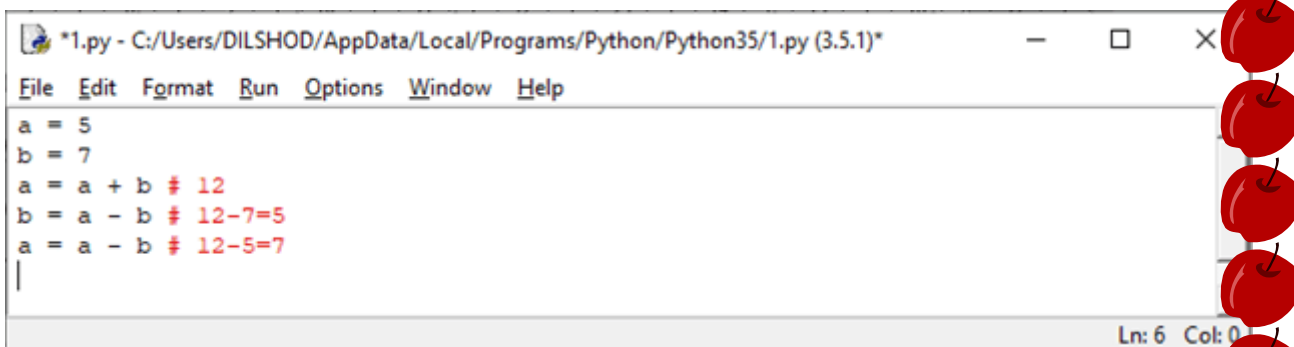
O'zgaruvchi qiymatlari o'rnini almashtirish.

Ikki o'zgaruvchiga qiymat kiritib ($a=5$ va $b=6$), to'g'ridan-to'g'ri bir o'zgaruvchi qiymatini ikkinchi o'zgaruvchiga beradigan bo'lsa, birinchisining qiymati yo'qotiladi. ($a=5$, $b=6$ bo'lsa, $a=b$ bo'lgan taqdirda $a=6$, $b=6$ hosil bo'ladi. $a=5$ qiymat yo'qotiladi.) Shuning uchun ikki o'zgaruvchi qiymatlarini o'rnini almashtirishda uchunchi o'zgaruvchi kiritiladi. Birinchi o'zgaruvchining qiymatini uchunchi o'zgaruvchiga, ikkinchi o'zgaruvchining qiymatini birinchi o'zgaruvchiga undan so'ng uchunchi o'zgaruvchi qiymatini ikkinchi o'zgaruvchiga beriladi.



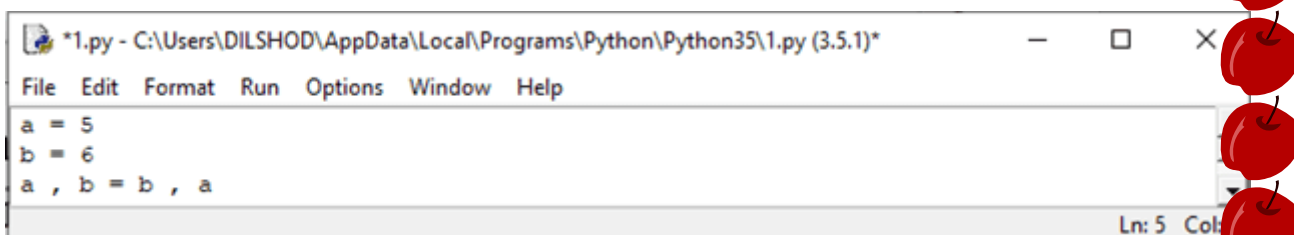
```
1.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/1.py (3.5.1)
File Edit Format Run Options Window Help
a = 5
b = 6
c = a
a = b
b = c
Ln: 8 Col: 0
```

Bundan tashqari o'zgaruvchi qiymatini o'rnini almashtirishning quyidagicha usuli ham mavjud.



```
*1.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/1.py (3.5.1)*
File Edit Format Run Options Window Help
a = 5
b = 7
a = a + b # 12
b = a - b # 12-7=5
a = a - b # 12-5=7
Ln: 6 Col: 0
```

Python dasturlash tilida bunday holatlarni yengillashtirilgan ko'rinishda amalga oshirish mumkin. Python dasturlash tilida bir tipdagi o'zgaruvchilar qiymatini almashtirishdan tashqari, turli tipdagi o'zgaruvchilar qiymatini ham shunday tarz to'g'ridan-to'g'ri almashtirish mumkin.



```
*1.py - C:\Users\DILSHOD\AppData\Local\Programs\Python\Python35\1.py (3.5.1)*
File Edit Format Run Options Window Help
a = 5
b = 6
a , b = b , a
Ln: 5 Col: 0
```

Uch xonali sonning raqamlar yig'indisini topish.

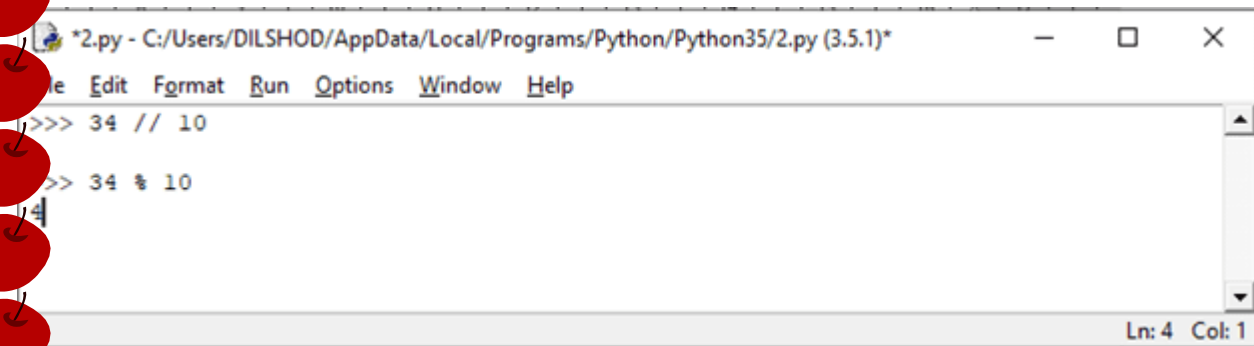
Foydalanuvchi uch xonali son kiritadi. Dastur uch xonali sonning raqamlar yig'indisini topishi lozim. Misol uchun: 349 kiritilgan bo'lsa, dastur (3 + 4 + 9 = 16) 16 sonini chiqarishi lozim.

Bunday masalalarni hal qilish uchun dasturlash tillarida 2 ta jarayon mavjud:

1) Ikkita sonni bo'lganda bo'linmaning butun qismini olish.

2) Ikkita sonni bo'lganda bo'linmaning qoldiq qismini olish.

Python dasturlash tilida butun qismini olish uchun //, kasr qismini olish uchun % belgilari mavjud.



```
*2.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/2.py (3.5.1)*
File Edit Format Run Options Window Help
>>> 34 // 10
10
>>> 34 % 10
4
```

Uch xonali sonni abc deb olaylik va quyidagicha algoritmni bajaramiz:

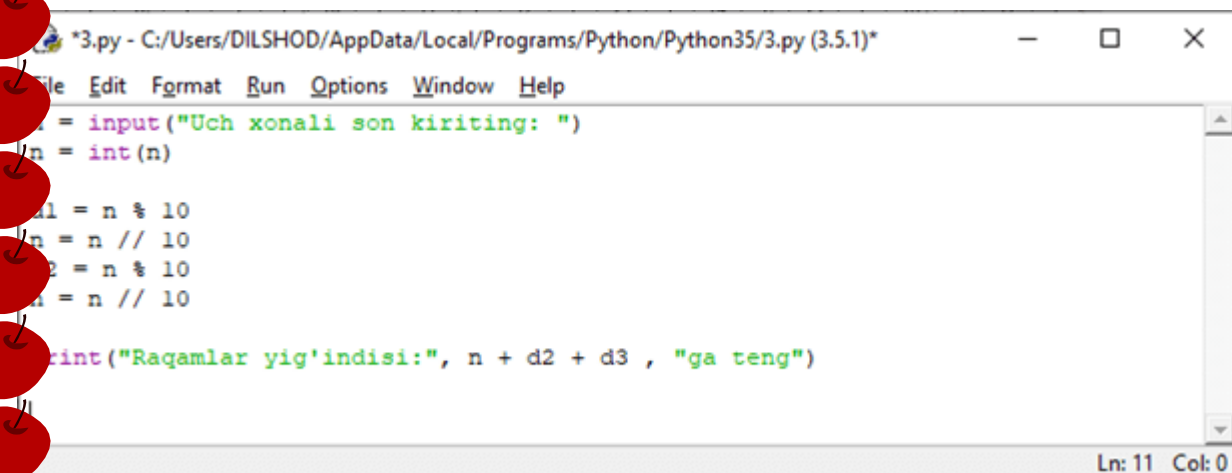
1. abc ni 10 bo'lgandagi qoldiqni topib, d1 o'zgaruvchiga taminlaymiz. Bu esa c soni bo'ladi.

2. abc ni 10 bo'lib, c sonidan ajratamiz.

3. ab ni 10 bo'lgandagi qoldiqni topib, d2 o'zgaruvchiga taminlaymiz. Bu esa b soni bo'ladi.

4. ab ni 10 bo'lib, b sonidan ajratamiz.

5. Qolgan a ga d1 va d2 ni qo'shib chiqamiz.



```
*3.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/3.py (3.5.1)*
File Edit Format Run Options Window Help
n = input("Uch xonali son kiriting: ")
n = int(n)
d1 = n % 10
n = n // 10
d2 = n % 10
n = n // 10
print("Raqamlar yig'indisi:", n + d2 + d3 , "ga teng")
```

Natija:

```
Python 3.5.1 Shell - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/3.py (3.5.1)
File Edit Shell Debug Options Window Help
>>>
=== RESTART: C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/3.py ===
Uch xonali son kiriting: 759
Raqamlar yig'indisi: 21 ga teng
>>>
...
Ln: 32 Col: 0
```

Kvadrat tenglamaning ildizlarini topish.

$ax^2 + bx + c = 0$ kvadrat tenglama berilgan. Kvadrat tenglamaning ildizlarini aniqlashdan oldin uning diskriminantini $D = b^2 - 4ac$ aniqlab olamiz. Agar $D > 0$ bo'lsa, kvadrat tenglama ikkita ildizga, agar $D = 0$ bo'lsa, kvadrat tenglama bitta ildizga, agar $D < 0$ bo'lsa, kvadrat tenglama ildizga ega bo'lmaydi.

```
*4.py - C:\Users\DILSHOD\AppData\Local\Programs\Python\Python35\4.py (3.5.1)*
File Edit Format Run Options Window Help
import math

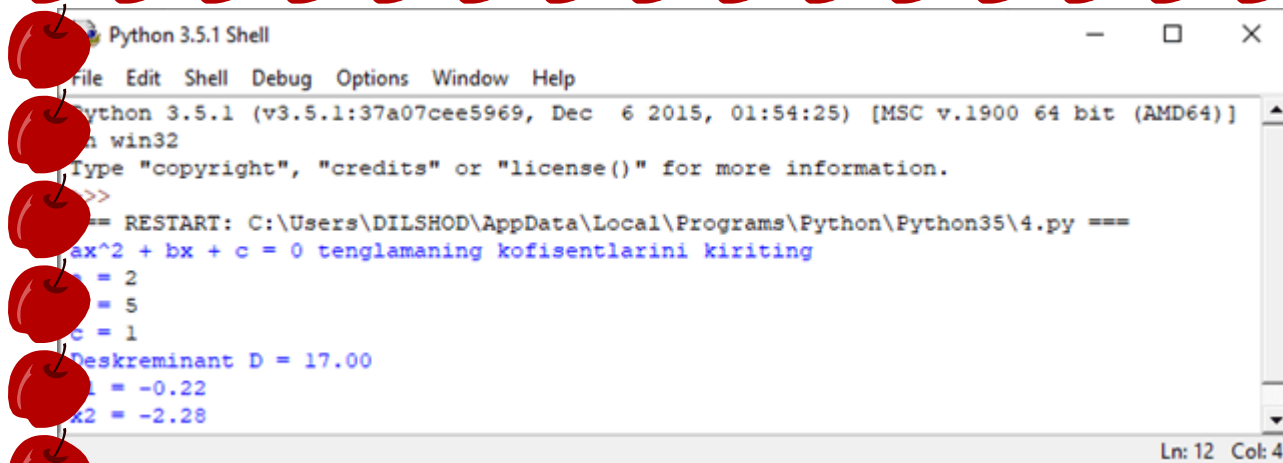
print("ax^2 + bx + c = 0 tenglamaning kofisientlarini kiriting ")
a = float(input("a = "))
b = float(input("b = "))
c = float(input("c = "))

discr = b ** 2 - 4 * a * c
print("Diskreminant D = %.2f" % discr)

if discr > 0:
    x1 = (-b + math.sqrt(discr)) / (2 * a)
    x2 = (-b - math.sqrt(discr)) / (2 * a)
    print("x1 = %.2f \nx2 = %.2f" % (x1, x2))
elif discr == 0:
    x = -b / (2 * a)
    print("x = %.2f" % x)
else:
    print("Kvadrat tenglama ildizga ega emas")

Ln: 21 Col: 0
```

Natija:



```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
Python 3.5.1 (v3.5.1:37a07cee5969, Dec 6 2015, 01:54:25) [MSC v.1900 64 bit (AMD64)]
on win32
Type "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:\Users\DILSHOD\AppData\Local\Programs\Python\Python35\4.py ===
ax^2 + bx + c = 0 tenglamaning kofisientlarini kiriting
a = 2
b = 5
c = 1
Diskreminant D = 17.00
x1 = -0.22
x2 = -2.28
Ln: 12 Col: 4
```

To'rt burchak, uchburchak va doiraning yuzlaridan birini hisoblash.

Foydalanuvchi tomonidan berilgan shakllarni (To'rt burchak, uchburchak va doira) tanlash orqali shaklning yuzini hisoblash dasturini tuzish lozim. Buning uchun shakllarni yuzini hisoblovchi formulalarni aniqlab olish lozim.

1. To'rt burchak yuzini hisoblash uchun uning tomonlarini (a, b) kiritish va $S=a*b$ formula orqali hisoblaymiz.

2. Uchburchakning yuzini hisoblash uchun uning tomonlarini (a, b, c) kiritish va Heron formulasidan foydalanib hisoblaymiz.

$$p = \frac{a + b + c}{2}, \quad S = \sqrt{p(p - a)(p - b)(p - c)}$$

3. Doiraning yuzini hisoblash uchun uning radiusini (r) kiritish va $S=\pi r^2$ formula orqali hisoblaymiz.

```
5.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/5.py (3.5.1)
File Edit Format Run Options Window Help
print("1-To'rtburchak, 2-Uchburchak, 3-Doira")
figure = input("Shakni tanlang: ")

if figure == '1':
    print("To'rtburchak tomonlarini kiriting:")
    a = float(input("a = "))
    b = float(input("b = "))
    print("Yuzi: %.2f" % (a * b))
elif figure == '2':
    print("Uchburchak tomonlarini kiriting:")
    a = float(input("a = "))
    b = float(input("b = "))
    c = float(input("c = "))
    p = (a + b + c) / 2
    from math import sqrt
    s = sqrt(p * (p - a) * (p - b) * (p - c))
    print("Yuza: %.2f" % s)
elif figure == '3':
    r = float(input("Doira radiusini kiriting:"))
    from math import pi
    print("Yuza: %.2f" % (pi * r ** 2))
else:
    print("Qiyamat kiritishda hatolik")
```

Ln: 23 Col: 3

Natija:

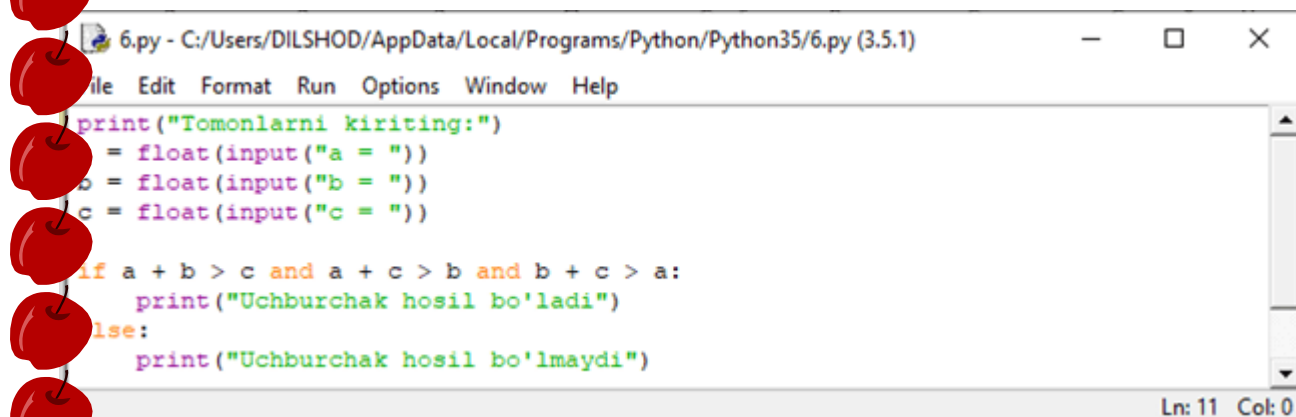
```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
Python 3.5.1 (v3.5.1:37a07cee5969, Dec 6 2015, 01:54:25) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/5.py ====
1-To'rtburchak, 2-Uchburchak, 3-Doira
Shakni tanlang: 2
Uchburchak tomonlarini kiriting:
a = 4
b = 5
c = 2
Yuza: 3.80
```

Ln: 12 Col: 4

Uchta kesma orqali uchburchak hosil qilish.

Uchburchak hosil bo'lishi uchun uchburchakni hosil qiladigan 2 ta tomon uzunligi 3 tomondan katta bo'lishi lozim. Buning uchun 3 ko'rinishda $a+b$, $b+c$, $a+c$

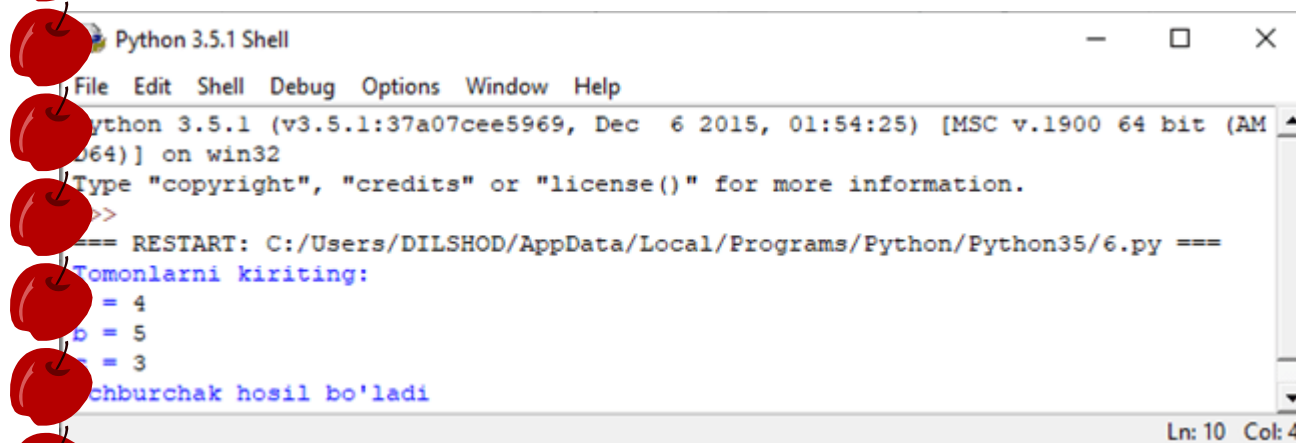
tomonlarni qo'shamiz va har bir ko'rinishdagi qiymatni qolgan tomon bilan taqqaslaymiz. Agar barcha tengsizlik qanoatlantirilsa uchburchak hosil bo'ladi.



```
6.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/6.py (3.5.1)
File Edit Format Run Options Window Help
print("Tomonlarni kiriting:")
a = float(input("a = "))
b = float(input("b = "))
c = float(input("c = "))

if a + b > c and a + c > b and b + c > a:
    print("Uchburchak hosil bo'ladi")
else:
    print("Uchburchak hosil bo'lmaydi")
Ln: 11 Col: 0
```

natija:



```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
Python 3.5.1 (v3.5.1:37a07cee5969, Dec 6 2015, 01:54:25) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>
==== RESTART: C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/6.py ====
Tomonlarni kiriting:
a = 4
b = 5
c = 3
Uchburchak hosil bo'ladi
Ln: 10 Col: 4
```

Dasturni murakkabroq ko'rinishga, yani qaysi tomonning uzunligi qolgan tomonlar uzunlik yig'indisidan katta ekanligini aniqlash lozim bo'lsa dastur kodini quyidagicha yozamiz.


```
6.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/6.py (3.5.1)
File Edit Format Run Options Window Help
print("Tomonlarni kiriting:")
a = float(input("a = "))
b = float(input("b = "))
c = float(input("c = "))

flag = ''
if a + b > c:
    if a + c > b:
        if b + c > a:
            print("Uchburchak hosil bo'ladi")
        else:
            flag = 'a'
    else:
        flag = 'b'
else:
    flag = 'c'

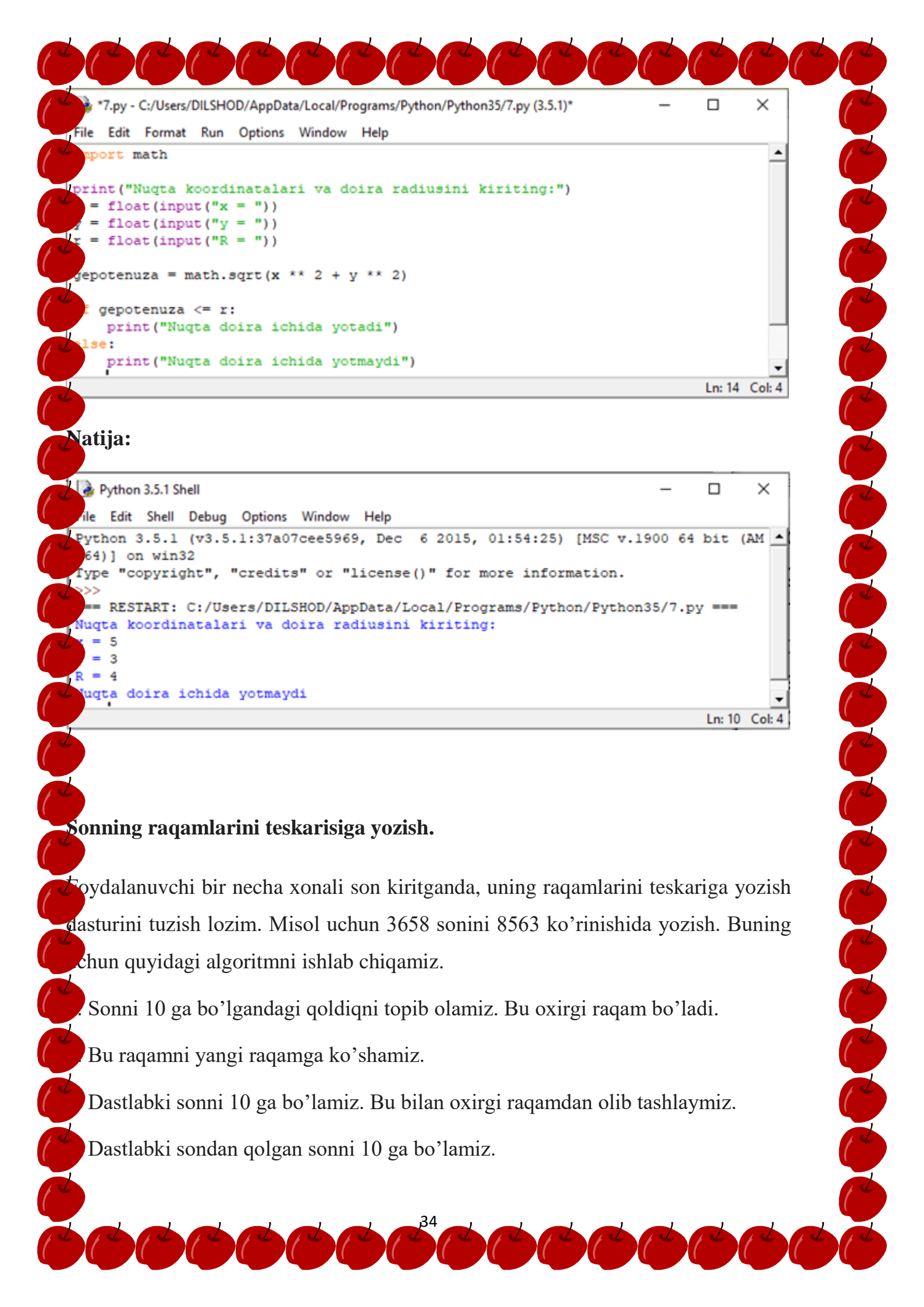
if flag != '':
    print("Uchburchak hosil bo'lmaydi")
    print("%s tomon uzunligi katta" % flag)
```

Natija:

```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
=== RESTART: C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/6.py ===
Tomonlarni kiriting:
a = 6
b = 2
c = 3
Uchburchak hosil bo'lmaydi
'a' tomon uzunligi katta
>>>
```

Nuqtani doira ichida yotish va totmasligini aniqlash.

Foydalanuvchi tomonidan dasturga nuqtaning koordinatalarini va uchburchakning radiusini kiritadi. Bunda doiraning markazi koordinata o'qining boshida yotadi. Nuqta joylashgan koordinataning x va y o'qiga proeksiyasi to'g'ri burchakli uchburchakning katetlari deb oladigan bo'lsak, uning gipotenuzasini topamiz. Topilgan gipotenuzani aylananing radiusi bilan taqqoslab, agar gipotenuza uzunligi aylana radiusidan kichik bo'lganda nuqta doira ichida yotadi. Aks holda nuqta doira ichida yotmaydi.



```
*7.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/7.py (3.5.1)*
File Edit Format Run Options Window Help
import math

print("Nuqta koordinatalari va doira radiusini kiriting:")
x = float(input("x = "))
y = float(input("y = "))
r = float(input("R = "))

gepotenuza = math.sqrt(x ** 2 + y ** 2)

if gepotenuza <= r:
    print("Nuqta doira ichida yotadi")
else:
    print("Nuqta doira ichida yotmaydi")

Ln: 14 Col: 4
```

Natija:

```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
Python 3.5.1 (v3.5.1:37a07cee5969, Dec 6 2015, 01:54:25) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/7.py ====
Nuqta koordinatalari va doira radiusini kiriting:
x = 5
y = 3
R = 4
Nuqta doira ichida yotmaydi

Ln: 10 Col: 4
```

Sonning raqamlarini teskarisiga yozish.

Foydalanuvchi bir necha xonali son kiritganda, uning raqamlarini teskariga yozish dasturini tuzish lozim. Misol uchun 3658 sonini 8563 ko'rinishida yozish. Buning uchun quyidagi algoritmni ishlab chiqamiz.

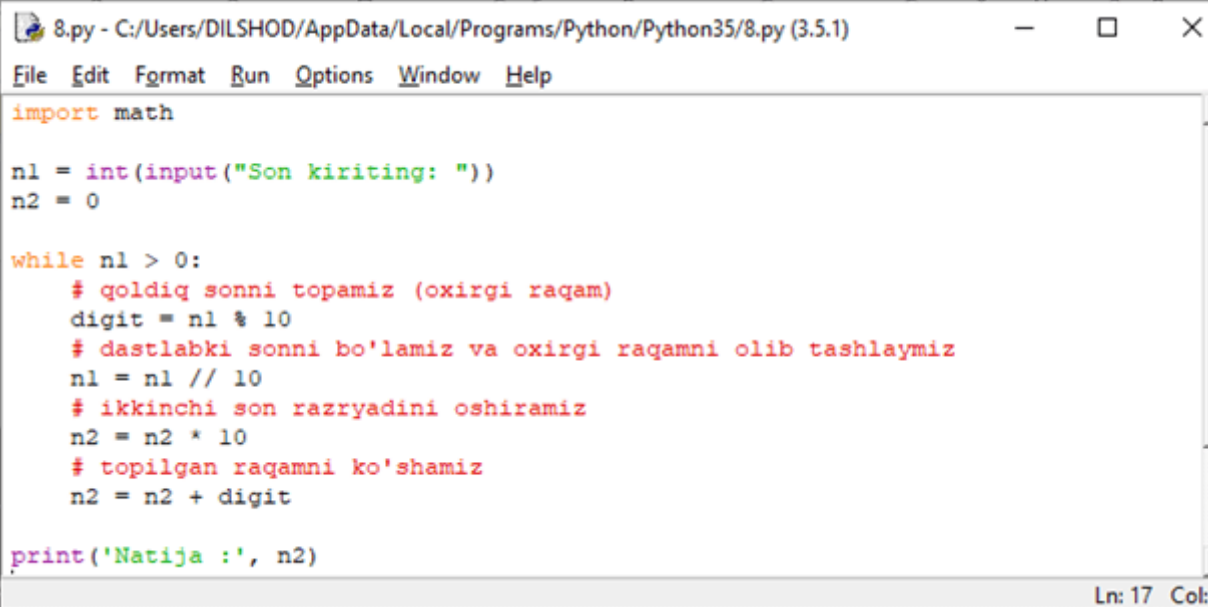
1. Sonni 10 ga bo'lgandagi qoldiqni topib olamiz. Bu oxirgi raqam bo'ladi.

2. Bu raqamni yangi raqamga ko'shamiz.

3. Dastlabki sonni 10 ga bo'lamiz. Bu bilan oxirgi raqamdan olib tashlaymiz.

4. Dastlabki son dan qolgan sonni 10 ga bo'lamiz.

5. Keyingi sonni 10 ga ko'paytiramiz. Shunday qilib, biz uning razriyadini ikkitaga oshiramiz va birinchi raqam razriyadini 10 xonalikka suramiz.
6. Oldingi yodda qolgan sonni birinchi sondan olingan ikkinchi songa qo'shamiz.
7. Bu xolatni dastlabki son 0 dan kichik bo'lguniga qadar davom ettiramiz.



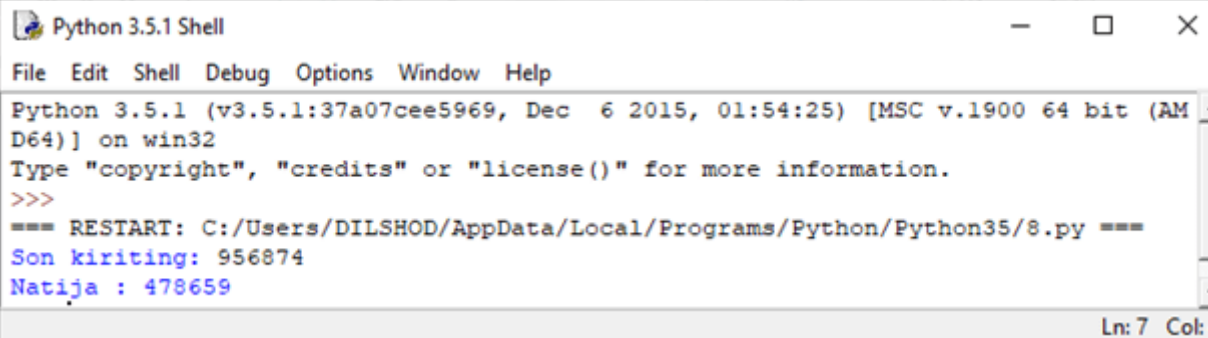
```
8.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/8.py (3.5.1)
File Edit Format Run Options Window Help
import math

n1 = int(input("Son kiriting: "))
n2 = 0

while n1 > 0:
    # qoldiq sonni topamiz (oxirgi raqam)
    digit = n1 % 10
    # dastlabki sonni bo'lamiz va oxirgi raqamni olib tashlaymiz
    n1 = n1 // 10
    # ikkinchi son razriyadini oshiramiz
    n2 = n2 * 10
    # topilgan raqamni ko'shamiz
    n2 = n2 + digit

print('Natija :', n2)
```

Natija:



```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
Python 3.5.1 (v3.5.1:37a07cee5969, Dec 6 2015, 01:54:25) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
=== RESTART: C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/8.py ===
Son kiriting: 956874
Natija : 478659
```

Masalani bunday algoritm bilan hal qilish barcha dasturlash tillari uchun mo'ljallanish tushadi. Python esa bunday hollar uchun `reverse()` metodini ishlab chiqqan. Bu metod berilganlarni teskari tartibda yozish imkonini beradi. `join()` satr metodi bilan esa barchasi bir satrga birlashtiriladi.

```
*9.py - C:\Users\DILSHOD\AppData\Local\Programs\Python\Python35\9.py (3.5.1)*
File Edit Format Run Options Window Help
n1 = input("Son kiriting: ")
n_list = list(n1)
n_list.reverse()
n2 = "".join(n_list)
print('Natija:', n2)
Ln: 6 Col: 0
```

Natija:

```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
Python 3.5.1 (v3.5.1:37a07cee5969, Dec 6 2015, 01:54:25) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:\Users\DILSHOD\AppData\Local\Programs\Python\Python35\9.py ===
Son kiriting: 69854
Natija: 45896
Ln: 7 Col: 4
```

berilgan songacha sonlarning kvadratlarini topish.

dan foydalanuvchi tomonidan kiritgan songacha sonlarning darajaga ko'tarish uchun quyidagi algoritmni bajaramiz.

1. Ko'tarilishi lozim bo'lgan darajani kiritamiz. (p)

2. Chegara raqamini kiritamiz. (n)

3. Takrorlanuvchi i soni chegara raqamiga teng bo'lgunicha p darajasini topamiz.

```
*10.py - C:\Users\DILSHOD\AppData\Local\Programs\Python\Python35\10.py (3.5.1)*
File Edit Format Run Options Window Help
p = int(input("Darajani kiriting: "))
n = int(input("Chegara sonini kiriting: "))
i = 1
while i ** p <= n:
    print(i ** p, end=' ')
    i += 1
print("\n Chegara songagacha ", i - 1, " son darajaga ko'rtarildi")
Ln: 10 Col: 0
```

Natija:

```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
Python 3.5.1 (v3.5.1:37a07cee5969, Dec 6 2015, 01:54:25) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
=== RESTART: C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/10.py ===
Darajani kiriting: 3
Chegara sonini kiriting: 300
1 8 27 64 125 216
Chegara songagacha 6 son darajaga ko'rtarildi
Ln: 9 Col:
```

Sonning raqamlarini toq yoki juftligini topish.

Foydalanuvchi tomonidan kiritilgan sonning raqamlari juft yoki toqligini aniqlash lozim bo'lsin. Buning uchun har bir raqamini 2 ga bo'lganda qoldiq 0 bo'lsa juft, dan farqli bo'lganda esa toq ekanligini aniqlaymiz. Python dasturlash tili qoldiqni aniqlash uchun % belgisidan foydalanamiz. Sonni raqamlarga ajratish uchun sonni // orqali 10 ga bo'lamiz.

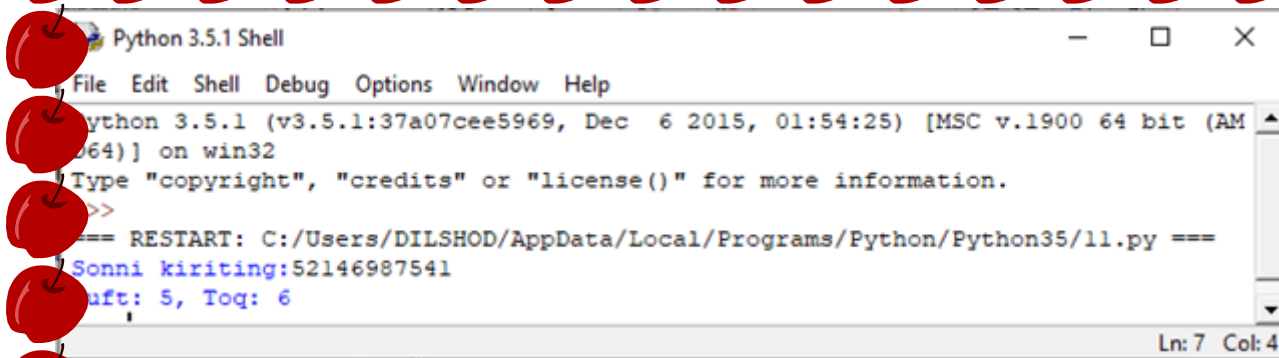
```
*11.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/11.py (3.5.1)*
File Edit Format Run Options Window Help
a = input("Sonni kiriting:")
a = int(a)

juft = 0
toq = 0

while a > 0:
    if a % 2 == 0:
        juft += 1
    else:
        toq += 1
    a = a // 10

print("Juft: %d, Toq: %d" % (juft, toq))
Ln: 15 Col: 0
```

Natija:



```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
Python 3.5.1 (v3.5.1:37a07cee5969, Dec 6 2015, 01:54:25) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/11.py ====
Sonni kiriting:52146987541
Suf: 5, Toq: 6
Ln: 7 Col: 4
```

Faktorialni hisoblash.

Faktorial 1 dan boshlab, berilgan songa qadar sonlarning ko'patmasini aniqlaydi.

Misol uchun 5 faktorial $1 * 2 * 3 * 4 * 5 = 120$ ga teng. Faktorial formulasini $n! = 1$

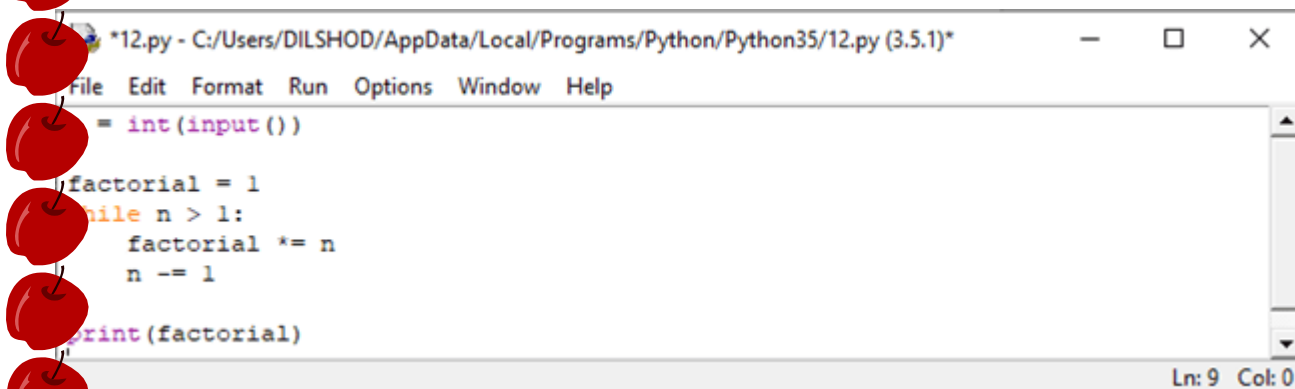
$2 * \dots * n$ yoki $n! = 1 * \dots * (n-2) * (n-1) * n$ ko'rinishdagi formulalar orqali

aniqlash mumkin.

Faktorialni aniqlash dasturini tuzish uchun takrorlash operatorlaridan

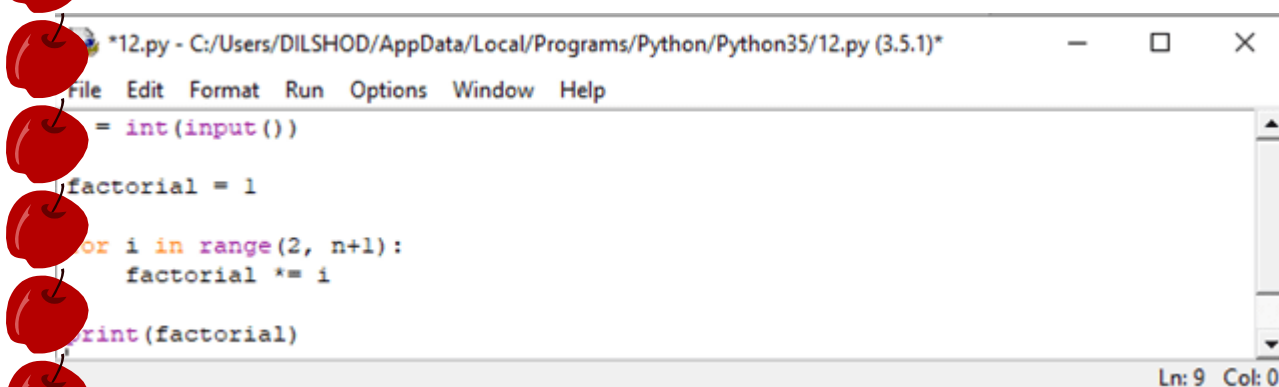
faydalanish mumkin.

While operatori orqali:



```
*12.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/12.py (3.5.1)*
File Edit Format Run Options Window Help
n = int(input())
factorial = 1
while n > 1:
    factorial *= n
    n -= 1
print(factorial)
Ln: 9 Col: 0
```

For operatori orqali:



```
*12.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/12.py (3.5.1)*
File Edit Format Run Options Window Help
n = int(input())
factorial = 1
for i in range(2, n+1):
    factorial *= i
print(factorial)
Ln: 9 Col: 0
```

Qaytish operatori orqali:

```
*12.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/12.py (3.5.1)*
File Edit Format Run Options Window Help
def fac(n):
    if n == 0:
        return 1
    return fac(n-1) * n

print(fac(5))
Ln: 8 Col: 0
```

Python dasturlash tili faktorialni hisoblash uchun yuqorida ko'rsatilgan dastur kodlarini tuzishni qisqartirish uchun **math** modulidagi **factorial()** funksiyasi ishlab chiqqan.

```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
Python 3.5.1 (v3.5.1:37a07cee5969, Dec 6 2015, 01:54:25) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> import math
>>> math.factorial(8)
40320
Ln: 6 Col: 0
```

Fibonachi sonlarini hisoblash.

Fibonachi sonlari shunday sonlar qatoriki o'zidan oldingi kelgan ikki sonning yig'indisidan tashkil topadi. Misol uchun 0,1,1,2,3,5,8,13,21,...

Fibonachi sonlarini topish uchun quyidagi formuladan foydalanamiz:

$$F_1 = 1$$

$$F_2 = 1$$

$$F_n = F_{n-1} + F_{n-2}$$

Fibonachi sonlarining yig'indisini topish dasturi quyidagi ko'rinishda tuzilishi mumkin:

```
*13.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/13.py (3.5.1)*
File Edit Format Run Options Window Help
fib1 = 1
fib2 = 1

n = input("Fibonachi raqamlar chegarasini kiriting: ")
n = int(n)

i = 0
while i < n - 2:
    fib_sum = fib1 + fib2
    fib1 = fib2
    fib2 = fib_sum
    i = i + 1

print(fib2)
Ln: 15 Col: 0
```

natija:

```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
Python 3.5.1 (v3.5.1:37a07cee5969, Dec 6 2015, 01:54:25) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/13.py ====
Fibonachi raqamlar chegarasini kiriting: 9
4
Ln: 7 Col: 4
```

Euklid algoritmi. Ikkita butun sonlarning eng katta umumiy bo'luvchisini topish.

Ikkita butun sonlarning eng katta umumiy bo'luvchisini topish algoritmlari:

1-algoritm.

1. Berilgan sonlarning kattasini kichigiga bo'lamiz.
2. Agar bo'lganda qoldiq 0 bo'linsa, u holda kichik son EKUB hisoblanadi.
3. Agar qoldiq chiqsa, unda katta sonni kichik son bilan, kichik sonni qoldiq bilan almashtiramiz.
4. 1 punktga qaytamiz.

Misol

uchur

30 va	18	sonlarning EKUB	ni	hisoblasak
30	/	18	=	1 (qoldiq 12)
18	/	12	=	1 (qoldiq 6)
12	/	6	=	2 (qoldiq 0)

Yakunlandi: EKUB (30, 18) = 6 ga teng.

2-algoritm.

1. Berilgan sonlarning kattasidan kichigini ayiramiz.
2. Agar ayirganda qoldiq 0 bo'linsa, u holda berilgan sonlar bir-biriga teng va shu sonlarning o'zi EKUB hisoblanadi.
3. Agar qoldiq chiqsa, unda katta sonni kichik son bilan, kichik sonni qoldiq bilan almashtiramiz.
4. 1 punktga qaytamiz.

Misol

uchur

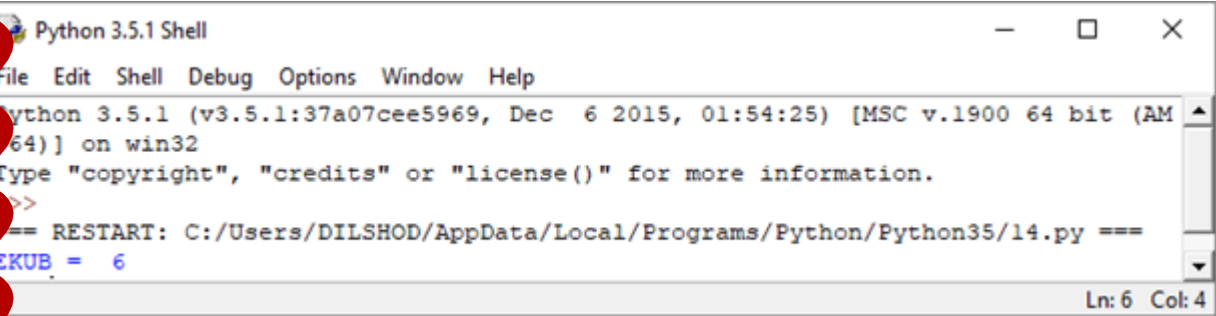
30 va	18	sonlarning EKUB	ni	hisoblasak
30	-	18	=	12
18	-	12	=	6
12 - 6 = 6				
6	-	6	=	0

Yakunlandi: EKUB (30, 18) = 6 ga teng.

```
14.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/14.py (3.5.1)
File Edit Format Run Options Window Help
a=30
b=18
while a!=b :
    if a>b :
        a = a - b
    else :
        b = b - a
print ('EKUB = ', a)
```

Ln: 9 Col: 0

Natija:



```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
Python 3.5.1 (v3.5.1:37a07cee5969, Dec 6 2015, 01:54:25) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/14.py ===
EKUB = 6
Ln: 6 Col: 4
```

Python turli platformalarda ishlaydi. (**Windows, Linux, Mac** va h.k)

Python ingliz tiliga o'xshash oddiy sintaksisga ega.

Python dasturlash tili boshqa dasturlash tillariga qaraganda dasturchiga kamroq kod yozishga imkonini beradi.

Python da **WEB, Desktop** va **Mobile** dasturlar yaratish imkoniyati mavjud.

Python kutubxonalar anchagina ko'p! deyarli barcha ishingizni kutubxonalar orqali bajarasiz!

Python xotiradan kam joy egallab ishlash tezligini ancha yuqori! [pypi](#) - pythonkutubxonalarsayti!

Python dasturlash tilining eng katta imkoniyati uning ochiq kodligida va kutubxonalarni ko'pligida va shu bilan birga undagi kodni ixchamligida qayta ishlash imkonlanadi. hozir sizga 3 ta dasturlash tilida C, Java va Python dasturlash tilida

"Hello,

World"

so'zini

yo'zib

ko'rami



"Hello, World"

- **C**

```
#include <stdio.h>

int main(int argc, char ** argv)
{
    printf("Hello, World!\n");
}
```
- **Java**

```
public class Hello
{
    public static void main(String argv[])
    {
        System.out.println("Hello, World!");
    }
}
```
- **now in Python**

```
print "Hello, World!"
```

2

3. Pythonda yozilgan dasturlarni ishga tushirish.

Pythonda yozilgan dasturlarni ishga tushirish.

Dasturni o'rnatish

Python-ni Windows 7, Mac OS X va Ubuntu tizimlariga qanday o'rnatishni endi bosqichma-bosqich amalga oshiramiz. Shuningdek, biz ish stolida Python ishlab chiqish muhiti bo'lgan IDLE yorlig'ini yaratamiz.

Python-ni Windows 7-ga yoki o'rnatish

Python-ni Microsoft Windows 7-ga o'rnatish uchun veb-brauzerni oching, <http://www.python.org/> kiriting va Windows uchun eng yangi Python 3 o'rnatuvchisini yuklab oling (Yuklashlar menyusiga o'ting va Windows-ni tanlang).

Pythonda qanday qilib 4 yoshlik bolani o'rgatish mumkin?

Men bolalarimga kod yozishni o'rgatmoqchi edim, chunki bu juda katta mahorat va kuchli fikrlash uslubi. Farzandlarim iPhone-da Angry Birds o'ynab ko'p vaqt o'tkazishdi va men ularga parda ortiga qarash va dasturlash qanday ishlashini o'rganish vaqti keldi deb qaror qildim. Shunday qilib, mening maslahatlarim. Ular butunlay shaxsiy tajribadan olingan va kimligingizdan qat'iy nazar siz uchun foydali bo'ladi - hardcore dasturchi yoki ilgari hech qachon kod bilan shug'ullanmagan kishi.

1-sonli maslahat.

Boshlang'ich sinfdan boshlang

O'g'lim Iftihor to'rt yoshda, biz iPad-da logotip yaratish uchun dasturlardan foydalanishni boshladik (Mad Logo) va bolalar uchun Scratch dasturlash tilidan animatsiyalar yaratdik. Ammo men o'g'limni ichkariga qarashini xohlardim va Pythonda dasturlash bilan uni jalb qilmoqdaman bir fokuslarni o'ylab topdim. Bo'lim tili bolani tushunishi uchun yetarlicha sodda va shu bilan birga mening xayolimga kelgan barcha g'oyalarni amalga oshirish uchun yetarlicha kuchli.

2-sonli maslahat. Python-ga o'ting.

Python allaqachon MacBook-da o'rnatilganligini bilasizmi? Siz faqat terminal oynasini ochishingiz va uni ishga tushirish uchun python-ni kiritishingiz kerak (chiqish uchun ctrl + d tugmalarini bosishingiz kerak). Python bilan ishlash uchun kerak bo'lgan hamma narsa ushbu qo'llanmada.

3-sonli maslahat. Kompyuteringiz bilan gaplashing.

Bolaga e'tibor berish uchun menda maxsus layf-xaker bor. Kompyuteringiz bilan suhbatlashing! Agar sizda MacBook bo'lsa, u holda terminal oynasini oching va "xush kelibsiz" buyrug'ini kiriting. Kompyuter "salom" deb javob beradi. Bu sehr emasmi?

4-sonli maslahat. Python raqamini taxmin qilish o'yinini o'ynang.

Ushbu dasturni ishga tushirish uchun:

Manba kodini yuklab oling

Terminal oynasini oching

CD yuklashlarini kiriting

python mathquiz.py-ni kiriting (va ovoz yoqilganligini tekshiring)

Bu haqiqiy taxminiy o'yin. Avvaliga o'g'lim Sai shunchaki o'ynashni yoqtirardi ammo keyin biz uning qanday ishlashini tushunish uchun kodning har bir satrini o'rganib chiqdik.

Ushbu faylni matn muharriri bilan oching va max_number raqamini 100 o'zgartiring.

5-sonli maslahat. Qolganlarning hammasini Internetda osongina topishingiz mumkin.

Farzandingizga qiyin narsani o'rgatmoqchi ekaningizni tasavvur qiling. Aytaylik, rekursiya. Bu nima ekanligini eslaysizmi? Men esladim, lekin la'nat, men juda qiziqar edim - bolaga haqiqatan ham qiyin bo'lgan narsani (masalan, rekursiya) qanday tushuntira olaman? Men Xan akademiyasining rekursiv faktorial dasturi haqida so'zlab beradigan juda zo'ridan foydalanardim.

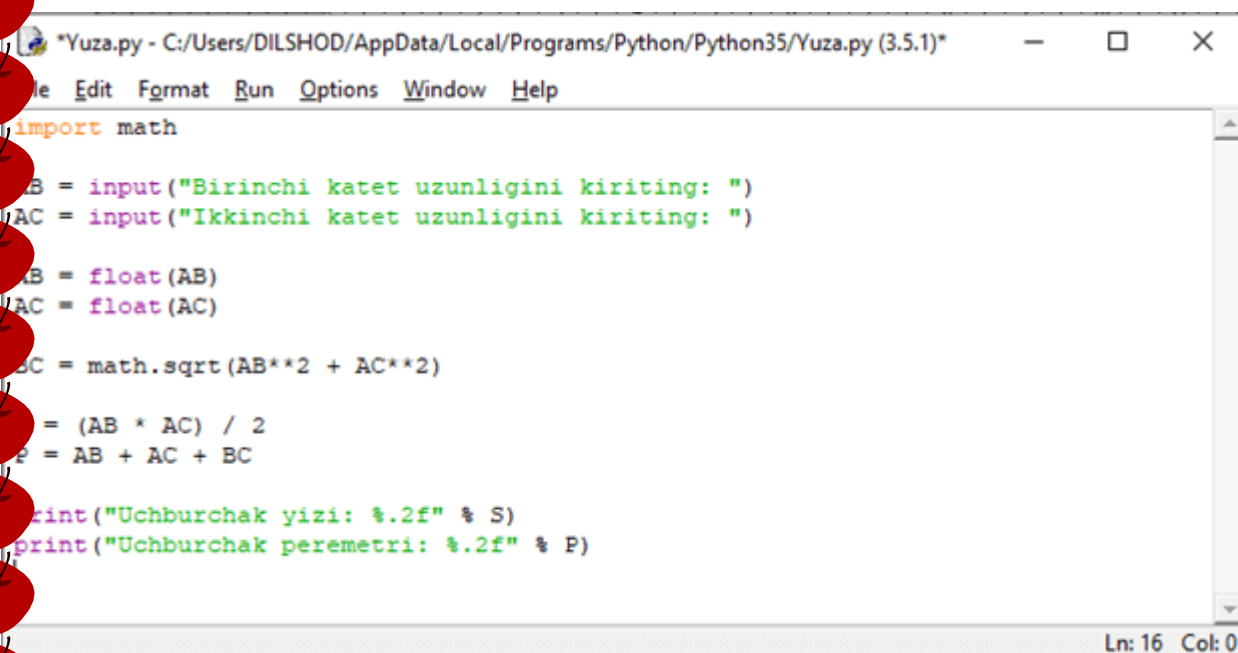
Mana o'g'lim bilan birga ishlatgan kod. U o'shanda 6 yoshda edi va menimcha, u rekursiyani tushunishga juda yosh edi, lekin u haqiqatan ham katta faktoriallarni kompyuterda tomosha qilishni juda yaxshi ko'rardi

Yuza va perimetrni hisoblash.

(To'g'ri burchakli uchburchak misolida)

To'g'ri burchakli uchburchakning yuzini va perimetrini hisoblash uchun katetlar uzunligi berilgan. Pifagor teoremasi ($c^2 = a^2 + b^2$) orqali gipatenuza uzunligini aniqlaymiz.

Python dasturlash tilida sonni ildizdan chiqarish uchun **math** modulining **sqrt()** funksiyasidan foydalanamiz.



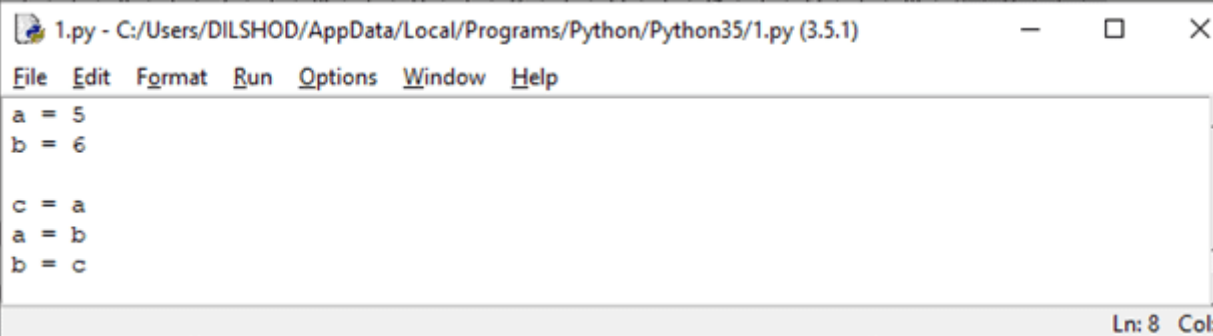
```
*Yuza.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/Yuza.py (3.5.1)*
File Edit Format Run Options Window Help
import math
AB = input("Birinchi katet uzunligini kiriting: ")
AC = input("Ikkinchi katet uzunligini kiriting: ")
AB = float(AB)
AC = float(AC)
BC = math.sqrt(AB**2 + AC**2)
S = (AB * AC) / 2
P = AB + AC + BC
print("Uchburchak yizi: %.2f" % S)
print("Uchburchak perimetri: %.2f" % P)
Ln: 16 Col: 0
```

Natija:

O'zgaruvchi qiymatlari o'rnini almashtirish.

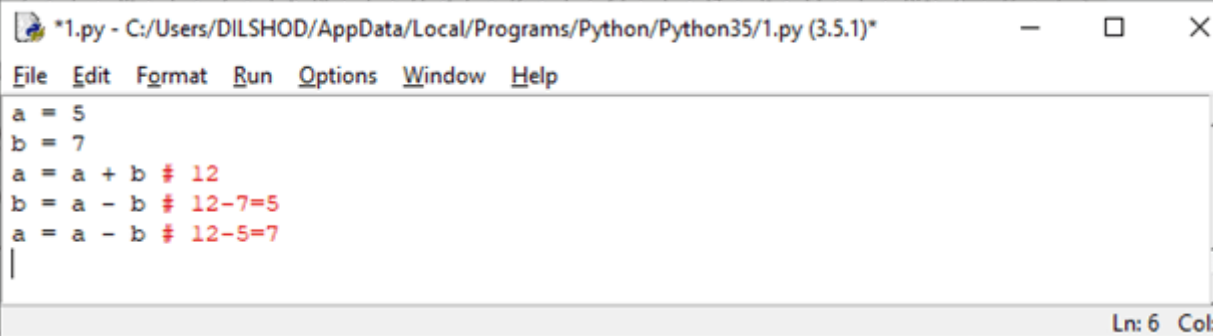
Ikki o'zgaruvchiga kiymat kiritib ($a=5$ va $b=6$), to'g'ridan-to'g'ri bir o'zgaruvchi qiymatini ikkinchi o'zgaruvchiga beradigan bo'lsa, birinchisining qiymati yo'qotiladi. ($a=5$, $b=6$ bo'lsa, $a=b$ bo'lgan taqdirda $a=6$, $b=6$ hosil bo'ladi. $a=5$ qiymat yo'qotiladi.) Shuning uchun ikki o'zgaruvchi qiymatlarini o'rnini almashtirishda uchunchi o'zgaruvchi kiritiladi. Birinchi o'zgaruvchining qiymatini

uchunchi o'zgaruvchiga, ikkinchi o'zgaruvchining qiymatini birinchi o'zgaruvchiga undan so'ng uchunchi o'zgaruvchi qiymatini ikkinchi o'zgaruvchiga beriladi.



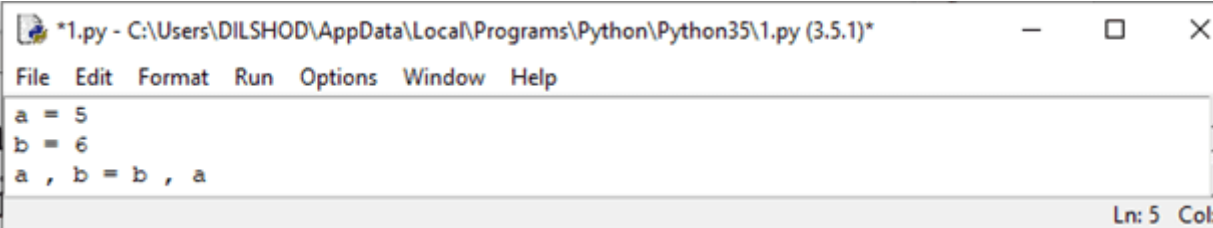
```
1.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/1.py (3.5.1)
File Edit Format Run Options Window Help
a = 5
b = 6
c = a
a = b
b = c
Ln: 8 Col:
```

Bundan tashqari o'zgaruvchi qiymatini o'rnini almashtirishning quyidagicha usuli ham mavjud.



```
*1.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/1.py (3.5.1)*
File Edit Format Run Options Window Help
a = 5
b = 7
a = a + b # 12
b = a - b # 12-7=5
a = a - b # 12-5=7
Ln: 6 Col:
```

Python dasturlash tilida bunday holatlarni yengillashtirilgan ko'rinishda amalga oshirish mumkin. Python dasturlash tilida bir tipdagi o'zgaruvchilar qiymatini almashtirishdan tashqari, turli tipdagi o'zgaruvchilar qiymatini ham shunday tarz to'g'ridan-to'g'ri almashtirish mumkin.

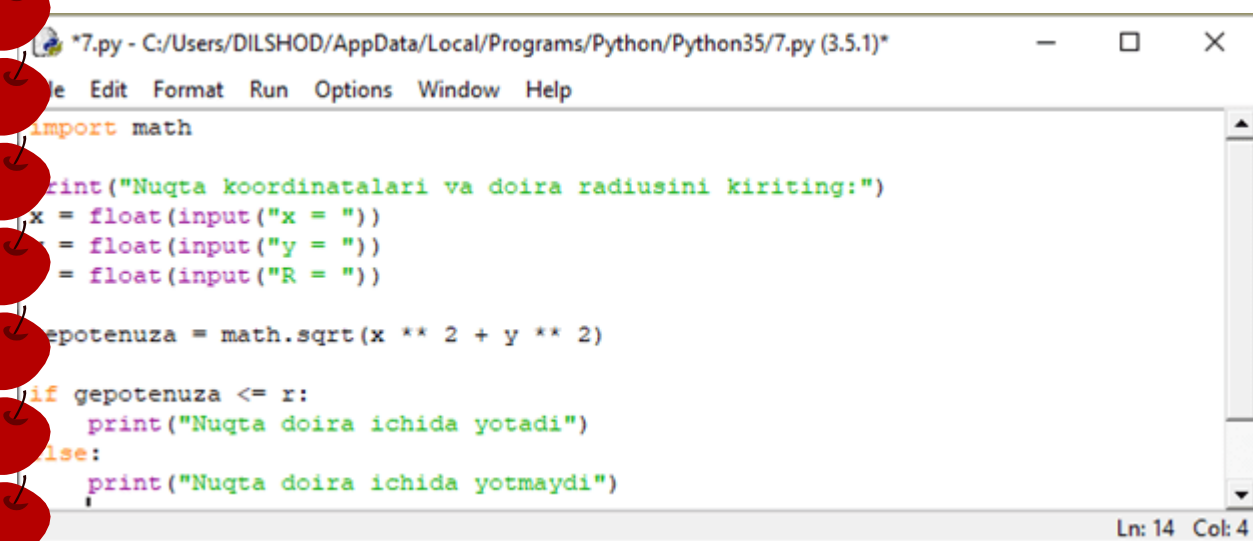


```
*1.py - C:\Users\DILSHOD\AppData\Local\Programs\Python\Python35\1.py (3.5.1)*
File Edit Format Run Options Window Help
a = 5
b = 6
a , b = b , a
Ln: 5 Col:
```

4.Dasturlashga doir masalalar.

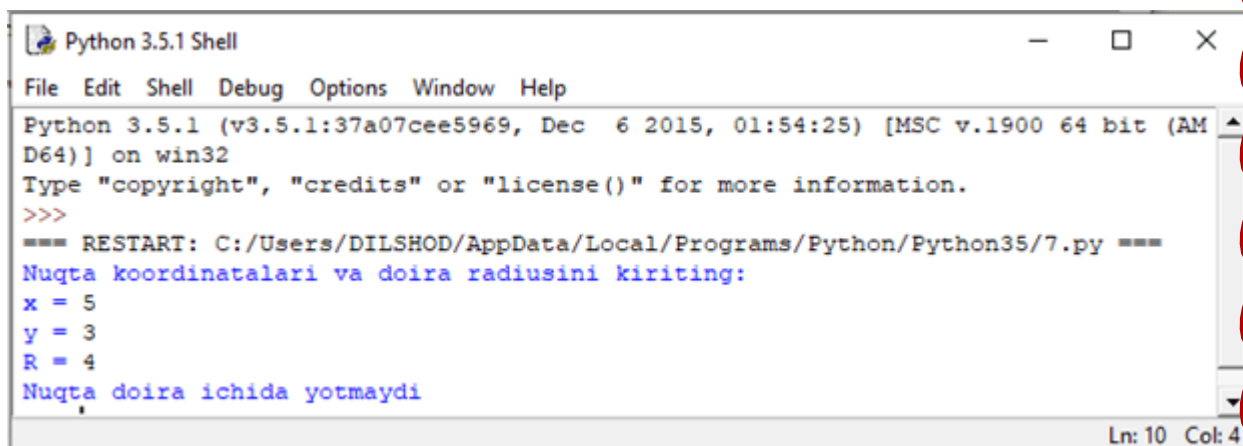
Nuqtani doira ichida yotish va totmasligini aniqlash.

Foydalanuvchi tomonidan dasturga nuqtaning koordinatalarini va uchburchak radiusini kiritadi. Bunda doiraning markazi koordinata o'qining boshida yotadi. Nuqta joylashgan koordinataning x va y o'qiga proeksiyasi to'g'ri burchakli uchburchakning katetlari deb oladigan bo'lsak, uning gipotenuzasini topamiz. Topilgan gipotenuzani aylananing radiusi bilan taqqoslab, agar gipotenuza uzunligi aylana radiusidan kichik bo'lganda nuqta doira ichida yotadi. Aks holda nuqta doira ichida yotmaydi.



```
*7.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/7.py (3.5.1)*
File Edit Format Run Options Window Help
import math
print("Nuqta koordinatalari va doira radiusini kiriting:")
x = float(input("x = "))
y = float(input("y = "))
R = float(input("R = "))
gipotenuza = math.sqrt(x ** 2 + y ** 2)
if gipotenuza <= r:
    print("Nuqta doira ichida yotadi")
else:
    print("Nuqta doira ichida yotmaydi")
Ln: 14 Col: 4
```

Natija:

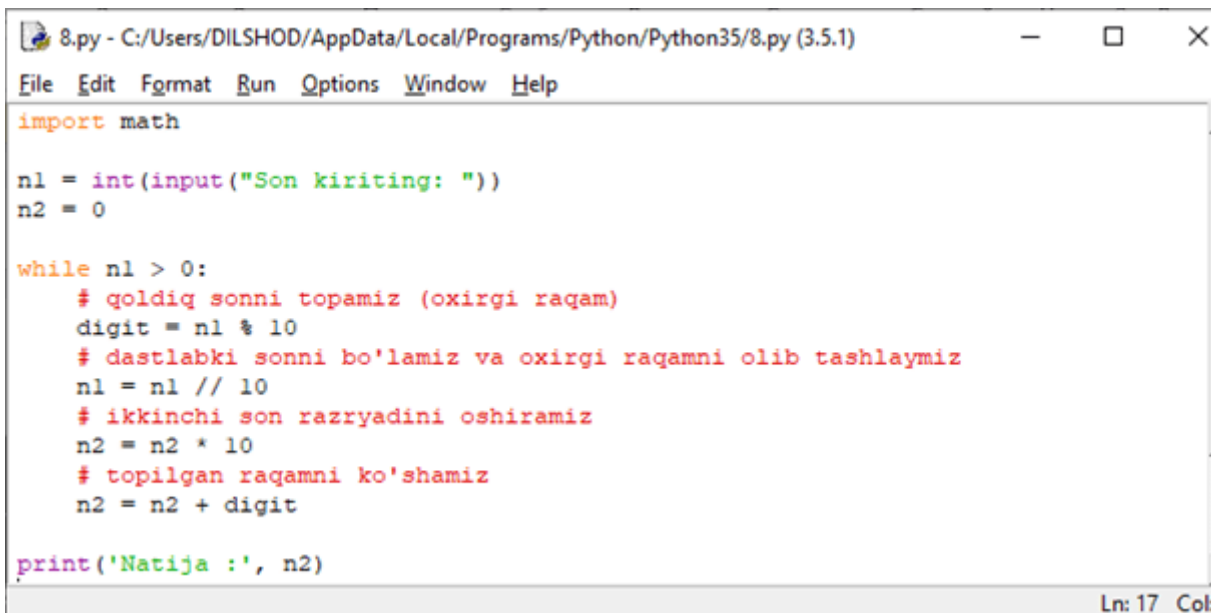


```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
Python 3.5.1 (v3.5.1:37a07cee5969, Dec 6 2015, 01:54:25) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/7.py ====
Nuqta koordinatalari va doira radiusini kiriting:
x = 5
y = 3
R = 4
Nuqta doira ichida yotmaydi
Ln: 10 Col: 4
```

Sonning raqamlarini teskarisiga yozish.

Foydalanuvchi bir necha xonali son kiritganda, uning raqamlarini teskariga yozish dasturini tuzish lozim. Misol uchun 3658 sonini 8563 ko'rinishida yozish. Buning uchun quyidagi algoritmni ishlab chiqamiz.

1. Sonni 10 ga bo'lgandagi qoldiqni topib olamiz. Bu oxirgi raqam bo'ladi.
2. Bu raqamni yangi raqamga ko'shamiz.
3. Dastlabki sonni 10 ga bo'lamiz. Bu bilan oxirgi raqamdan olib tashlaymiz.
4. Dastlabki sondan qolgan sonni 10 ga bo'lamiz.
5. Keyingi sonni 10 ga ko'paytiramiz. Shunday qilib, biz uning razriyadini ikkitaga oshiramiz va birinchi raqam razriyadini 10 xonalikka suramiz.
6. Oldingi yodda qolgan sonni birinchi sondan olingan ikkinchi songa qo'shamiz.
7. Bu xolatni dastlabki son 0 dan kichik bo'lguniga qadar davom ettiramiz.



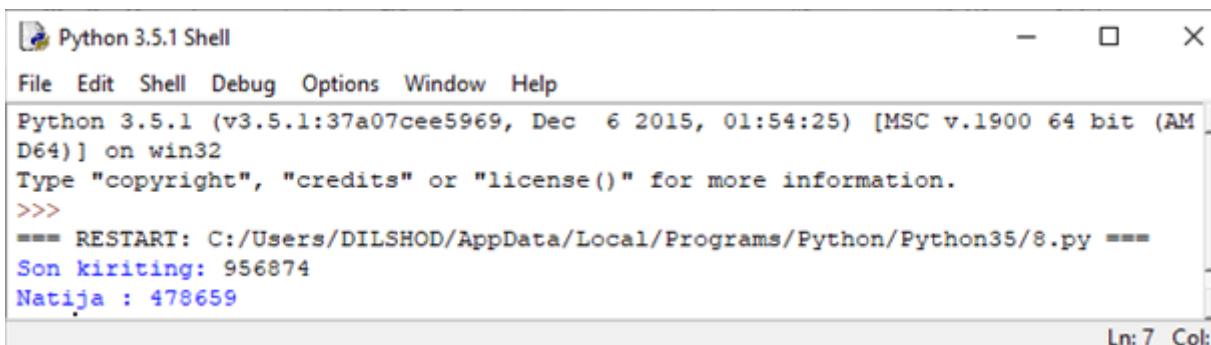
```
8.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/8.py (3.5.1)
File Edit Format Run Options Window Help
import math

n1 = int(input("Son kiriting: "))
n2 = 0

while n1 > 0:
    # qoldiq sonni topamiz (oxirgi raqam)
    digit = n1 % 10
    # dastlabki sonni bo'lamiz va oxirgi raqamni olib tashlaymiz
    n1 = n1 // 10
    # ikkinchi son razriyadini oshiramiz
    n2 = n2 * 10
    # topilgan raqamni ko'shamiz
    n2 = n2 + digit

print('Natija :', n2)
```

Natija:



```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
Python 3.5.1 (v3.5.1:37a07cee5969, Dec 6 2015, 01:54:25) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
=== RESTART: C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/8.py ===
Son kiriting: 956874
Natija : 478659
```

Masalani bunday algoritm bilan hal qilish barcha dasturlash tillari uchun mo'ljallanish tushadi. Python esa bunday hollar uchun `reverse()` metodini ishlab chiqqan. B

Metod berilganlarni teskari tartibda yozish imkonini beradi. **join()** satr metodi bilan esa barchasi bir satrga birlashtiriladi.

```
9.py - C:\Users\DILSHOD\AppData\Local\Programs\Python\Python35\9.py (3.5.1)*
File Edit Format Run Options Window Help
n1 = input("Son kiriting: ")
n_list = list(n1)
n_list.reverse()
n2 = "".join(n_list)
print('Natija:', n2)
Ln: 6 Col: 0
```

Natija:

```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
Python 3.5.1 (v3.5.1:37a07cee5969, Dec 6 2015, 01:54:25) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:\Users\DILSHOD\AppData\Local\Programs\Python\Python35\9.py ====
Son kiriting: 69854
Natija: 45896
Ln: 7 Col: 4
```

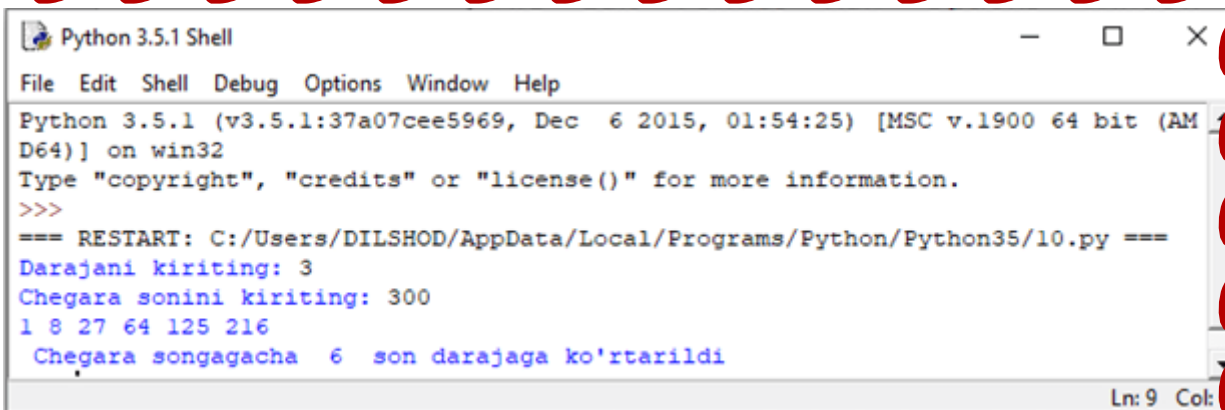
berilgan songacha sonlarning kadratlarini topish.

dan foydalanuvchi tomonidan kiritgan songacha sonlarning darajaga ko'tarish uchun quyidagi algoritmni bajaramiz.

- 1. Ko'tarilishi lozim bo'lgan darajani kiritamiz. (p)
- 2. Chegara raqamini kiritamiz. (n)
- 3. Takrorlanuvchi i soni chegara raqamiga teng bo'lgunicha p darajasini topamiz.

```
10.py - C:\Users\DILSHOD\AppData\Local\Programs\Python\Python35\10.py (3.5.1)*
File Edit Format Run Options Window Help
p = int(input("Darajani kiriting: "))
n = int(input("Chegara sonini kiriting: "))
i = 1
while i ** p <= n:
    print(i ** p, end=' ')
    i += 1
print("\n Chegara songagacha ", i - 1, " son darajaga ko'rtarildi")
Ln: 10 Col: 0
```

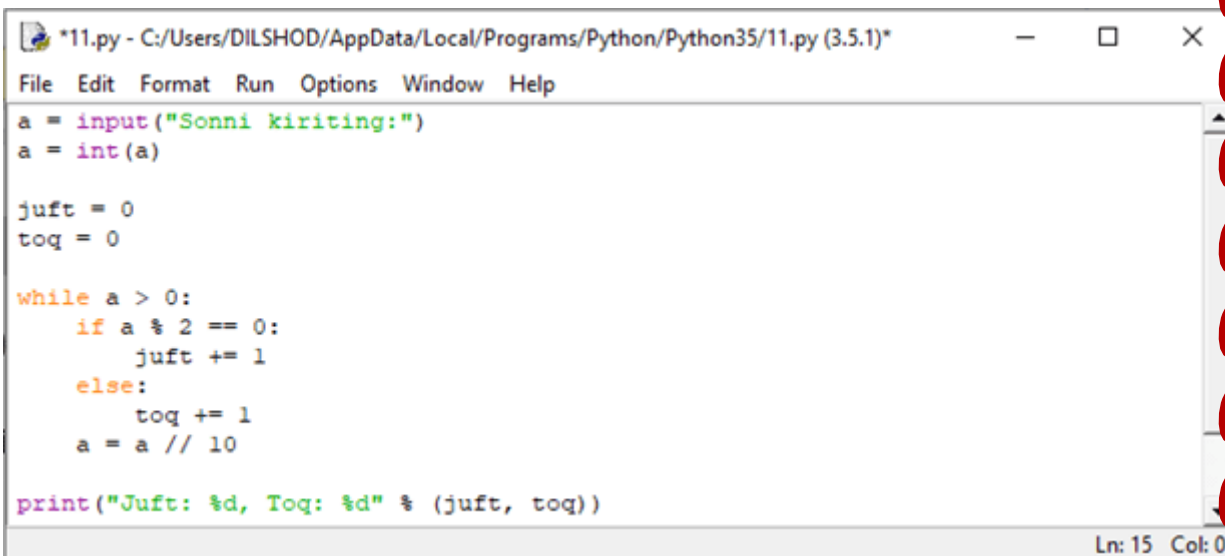
Natija:



```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
Python 3.5.1 (v3.5.1:37a07cee5969, Dec 6 2015, 01:54:25) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
=== RESTART: C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/10.py ===
Darajani kiriting: 3
Chegara sonini kiriting: 300
1 8 27 64 125 216
Chegara songagacha 6 son darajaga ko'rtarildi
Ln: 9 Col:
```

Sonning raqamlarini toq yoki juftligini topish.

Foydalanuvchi tomonidan kiritilgan sonning raqamlari juft yoki toqligini aniqlash lozim bo'lsin. Buning uchun har bir raqamini 2 ga bo'lganda qoldiq 0 bo'lsa juft, dan farqli bo'lganda esa toq ekanligini aniqlaymiz. Python dasturlash tili qoldiqni aniqlash uchun % belgisidan foydalanamiz. Sonni raqamlarga ajratish uchun sonni // orqali 10 ga bo'lamiz.



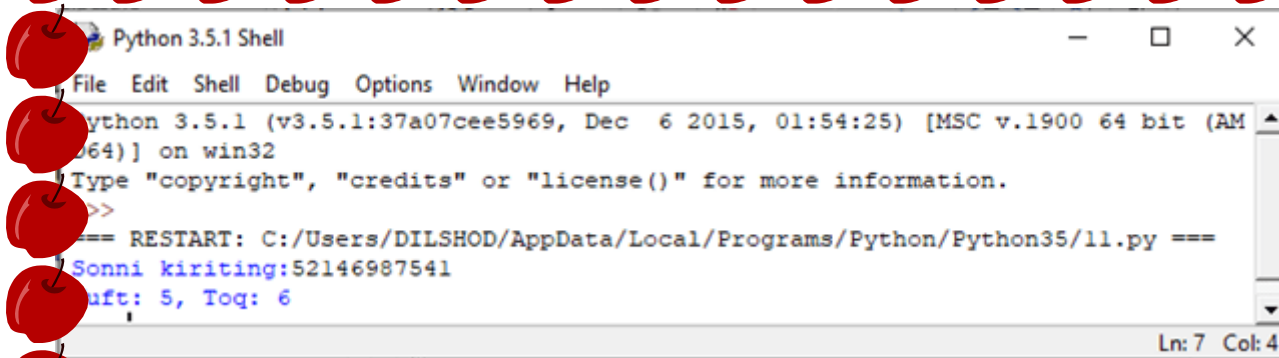
```
*11.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/11.py (3.5.1)*
File Edit Format Run Options Window Help
a = input("Sonni kiriting:")
a = int(a)

juft = 0
toq = 0

while a > 0:
    if a % 2 == 0:
        juft += 1
    else:
        toq += 1
    a = a // 10

print("Juft: %d, Toq: %d" % (juft, toq))
Ln: 15 Col: 0
```

Natija:



```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
Python 3.5.1 (v3.5.1:37a07cee5969, Dec 6 2015, 01:54:25) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/11.py ====
Sonni kiriting:52146987541
Suf: 5, Toq: 6
Ln: 7 Col: 4
```

Faktorialni hisoblash.

Faktorial 1 dan boshlab, berilgan songa qadar sonlarning ko'patmasini aniqlaydi.

Misol uchun 5 faktorial $1 * 2 * 3 * 4 * 5 = 120$ ga teng. Faktorial formulasini $n! = 1$

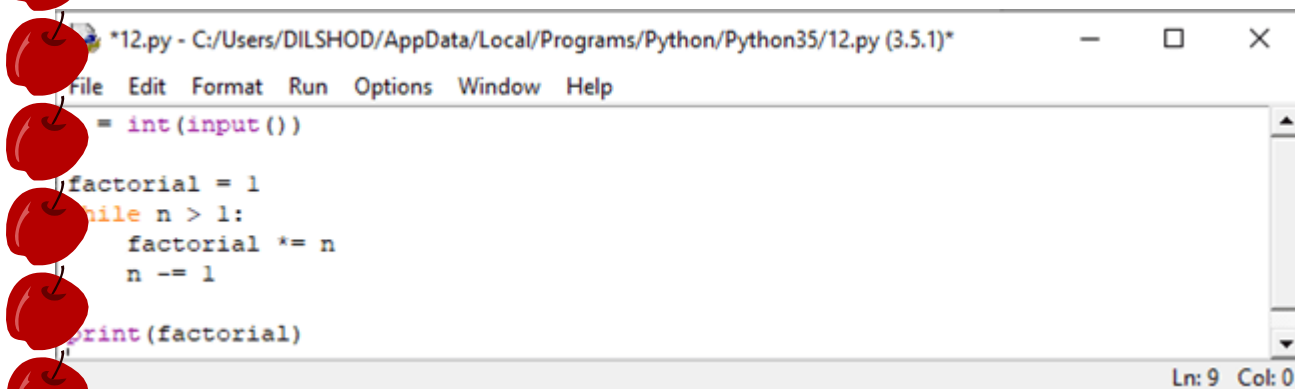
$2 * \dots * n$ yoki $n! = 1 * \dots * (n-2) * (n-1) * n$ ko'rinishdagi formulalar orqali

aniqlash mumkin.

Faktorialni aniqlash dasturini tuzish uchun takrorlash operatorlaridan

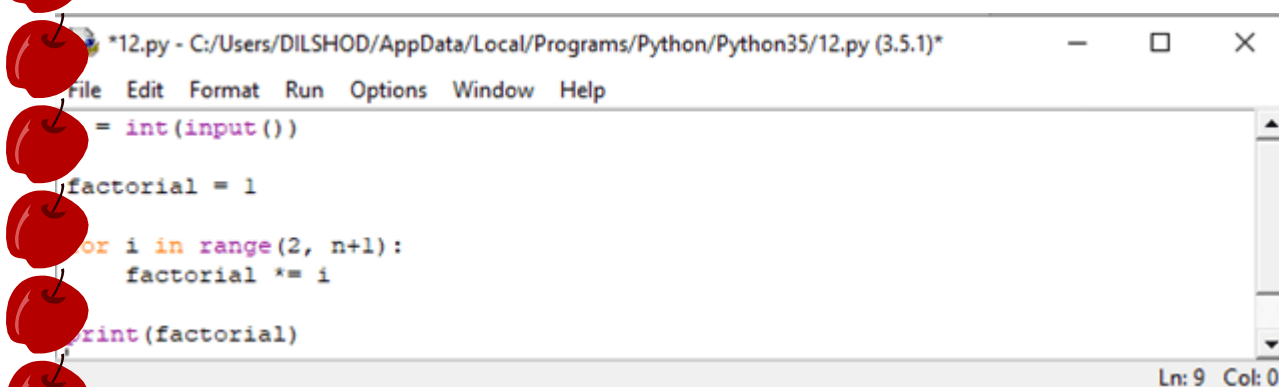
foydalanish mumkin.

While operatori orqali:



```
*12.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/12.py (3.5.1)*
File Edit Format Run Options Window Help
n = int(input())
factorial = 1
while n > 1:
    factorial *= n
    n -= 1
print(factorial)
Ln: 9 Col: 0
```

For operatori orqali:



```
*12.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/12.py (3.5.1)*
File Edit Format Run Options Window Help
n = int(input())
factorial = 1
for i in range(2, n+1):
    factorial *= i
print(factorial)
Ln: 9 Col: 0
```

Qaytish operatori orqali:

```
*12.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/12.py (3.5.1)*
File Edit Format Run Options Window Help
def fac(n):
    if n == 0:
        return 1
    return fac(n-1) * n

print(fac(5))
Ln: 8 Col: 0
```

Python dasturlash tili faktorialni hisoblash uchun yuqorida ko'rsatilgan dastur kodlarini tuzishni qisqartirish uchun **math** modulidagi **factorial()** funksiyasi ishlab chiqqan.

```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
Python 3.5.1 (v3.5.1:37a07cee5969, Dec 6 2015, 01:54:25) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> import math
>>> math.factorial(8)
40320
Ln: 6 Col: 0
```

Fibonachi sonlarini hisoblash.

Fibonachi sonlari shunday sonlar qatoriki o'zidan oldingi kelgan ikki sonning yig'indisidan tashkil topadi. Misol uchun 0,1,1,2,3,5,8,13,21,...

Fibonachi sonlarini topish uchun quyidagi formuladan foydalanamiz:

$$F_1 = 1$$

$$F_2 = 1$$

$$F_n = F_{n-1} + F_{n-2}$$

Fibonachi sonlarining yig'indisini topish dasturi quyidagi ko'rinishda tuzish mumkin:

```
*13.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/13.py (3.5.1)*
File Edit Format Run Options Window Help
fib1 = 1
fib2 = 1

n = input("Fibonachi raqamlar chegarasini kiriting: ")
n = int(n)

i = 0
while i < n - 2:
    fib_sum = fib1 + fib2
    fib1 = fib2
    fib2 = fib_sum
    i = i + 1

print(fib2)
Ln: 15 Col: 0
```

natija:

```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
Python 3.5.1 (v3.5.1:37a07cee5969, Dec 6 2015, 01:54:25) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/13.py ====
Fibonachi raqamlar chegarasini kiriting: 9
9
Ln: 7 Col: 4
```

Euklid algoritmi. Ikkita butun sonlarning eng katta umumiy bo'luvchisini topish.

Ikkita butun sonlarning eng katta umumiy bo'luvchisini topish algoritmlari:

1-algoritm.

1. Berilgan sonlarning kattasini kichigiga bo'lamiz.
2. Agar bo'lganda qoldiq 0 bo'linsa, u holda kichik son EKUB hisoblanadi.
3. Agar qoldiq chiqsa, unda katta sonni kichik son bilan, kichik sonni qoldiq bilan almashtiramiz.
4. 1 punktga qaytamiz.

Misol uchun:

30 va 18 sonlarning EKUB ni hisoblasak.

$$30 / 18 = 1 \text{ (qoldiq 12)}$$

$$18 / 12 = 1 \text{ (qoldiq 6)}$$

$$12 / 6 = 2 \text{ (qoldiq 0)}$$

Yakunlandi: EKUB (30, 18) = 6 ga teng.

2-algoritm.

1. Berilgan sonlarning kattasidan kichigini ayiramiz.
2. Agar ayirganda qoldiq 0 bo'linsa, u holda berilgan sonlar bir-biriga teng va shu sonlarning o'zi EKUB hisoblanadi.
3. Agar qoldiq chiqsa, unda katta sonni kichik son bilan, kichik sonni qoldiq bilan almashtiramiz.
4. 1 punktga qaytamiz.

Misol

uchun

30 va	18	sonlarning EKUB	ni	hisoblasak
30	-	18	=	12
18	-	12	=	6
12 - 6 = 6				
6	-	6	=	6

Yakunlandi: EKUB (30, 18) = 6 ga teng.

```
14.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/14.py (3.5.1)
File Edit Format Run Options Window Help
a=30
b=18
while a!=b :
    if a>b :
        a = a - b
    else :
        b = b - a
print ('EKUB = ', a)
```

Ln: 9 Col: 0

Natija:

```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
Python 3.5.1 (v3.5.1:37a07cee5969, Dec 6 2015, 01:54:25) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/14.py ===
>>> EKUB = 6
Ln: 6 Col: 4
```

Dasturlashga doir masalalar.

1. Berilgan raqamlar ro'yxatidagi o'rtacha qiymatni hisoblash dasturi

1.1. Yazifaning tavsifi

Dastur birma-bir kiritilgan raqamlarni qabul qiladi, ular ro'yxatda saqlanadi. Keyin ushbu ro'yxatdagi barcha elementlarning o'rtacha qiymatini nashr etadi.

1.2. Masalaning yechimi

1. Kiritish sifatida ro'yxatda saqlanadigan narsalar sonini saqlash uchun o'zgaruvchini kiriting.

2. Ro'yxat elementlarini kiritish uchun for loopdan foydalanamiz.

3. Biz ro'yxatning barcha elementlari yig'indisini hisoblaymiz.

4. Ro'yxatdagi barcha elementlarning yig'indisini ushbu elementlarning soniga bo'ling.

5. Oxiri.

1.3. Dasturning manba kodi

Quyida ro'yxatdagi o'rtacha qiymatni hisoblash uchun dasturning manba kodi keltirilgan. Ushbu dasturning bajarilish natijasi quyida keltirilgan:

```
n = int(input("Ro'yhatning elementlar sonini kiriting: "))
```

```
l = []
```

```
for i in range(0, n):
```

```
    elem = int(input("Ro'yhatning elementini kiriting: "))
```




```
a.append(elem)
```

```
avg = sum(a) / n
```

```
print("Ro'yhat elementlarining o'rtacha qiymati "),round(avg, 2))
```

Dasturning qanday ishlashini tushuntirish

1. Foydalanuvchi ro'yxatga n (int turi) o'zgaruvchida saqlanadigan elementlar sonini kiritishi kerak.
2. Bo'sh ro'yxatni yarating a .
3. for tsiklining i o'zgaruvchisining qiymati 0 dan avval kiritilgan o'zgaruvchisining qiymatiga (bu ro'yxatdagi elementlar sonini bildiradi) o'zgaradi va tsiklning har bir takrorlanishi bilan 1 ga ko'payadi.
4. Keyin foydalanuvchi kiritishi kerak bo'lgan qiymat $elem$ o'zgaruvchisiga yoziladi.
5. $a.append(elem)$ usuli $elem$ o'zgaruvchisining qiymatini a ro'yxatiga qo'shadi.

6. Endi i tsikli o'zgaruvchisining qiymati bittaga ko'paytiriladi va 1 ga teng bo'ladi.

7. Keyin, foydalanuvchi yangi qiymatni kiritadi, u yana o'zgaruvchiga yoziladi va keyin a ro'yxatiga qo'shiladi.

8. Shunday qilib, tsikl i o'zgaruvchining qiymati n o'zgaruvchiga teng bo'lguncha ishlaydi.

9. $sum(a)$ funktsiyasi ro'yxatdagi barcha elementlarning yig'indisini hisoblab chiqadi. Ushbu summani n ga bo'linib, a ro'yxatidagi elementlarning o'rtacha qiymatini olamiz.

10. $round(avg, 2)$ funktsiya o'rtacha qiymatni o'nli kasrgacha yaxlitlaydi.

11. Shundan so'ng, bosib chiqarish funktsiyasi yordamida ekranda o'rtacha qiymat ko'rsatiladi.

Natijalar:

Misol 1:



↳ o'yhat elementlarining sonini kiriting: 3

↳ o'yhat elementini kiriting: 23

↳ o'yhat elementini kiriting: 45

↳ o'yhat elementini kiriting: 56

↳ o'yhat elementlarining o'rtacha qiymati 41.33

↳ Misol 2:

↳ o'yhat elementlarining sonini kiriting: 5

↳ o'yhat elementinin kiriting: 12

↳ o'yhat elementinin kiriting: 24

↳ o'yhat elementinin kiriting: 33

↳ o'yhat elementinin kiriting: 25

↳ o'yhat elementinin kiriting: 18

↳ o'yhat elementlarining o'rtacha qiymati 22.4

↳ Ikki o'zgaruvchining qiymatlarini almashinuvi

↳ vazifaning tavsifi

↳ Ushbu dastur foydalanuvchidan ikkita qiymat oladi va ularning har birini alohida o'zgaruvchiga yozadi. Keyin vaqtinchalik o'zgaruvchini kiritmasdan ushbu o'zgaruvchilarning qiymatlarini almashtiradi.

↳ Masalning echimi

↳ Foydalanuvchi ikki xil o'zgaruvchiga yozilgan ikkita qiymatni kiritadi.

↳ 1. Ikki o'zgaruvchining qiymatlari qo'shiladi va qo'shilish natijasi birinchi o'zgaruvchiga yoziladi.

↳ Keyin ikkinchi o'zgaruvchi birinchi o'zgaruvchidan ayiriladi va natija ikkinchi o'zgaruvchiga yoziladi.



4. Keyin birinchi o'zgaruvchidan ikkinchisini chiqaramiz (allaqachon yangilangan qiymat bilan) va olingan natijani birinchi o'zgaruvchiga yozamiz.

5. Olingan qiymatlarni namoyish etamiz.

6. Ohiri.

Dasturning manba kodi

Quyida ikkita o'zgaruvchining qiymatlarini uchinchi o'zgaruvchini ishlatmasdan almashtirish dasturining manba kodi keltirilgan. Quyida ushbu kod natijasi ham ko'rsatiladi.

```
a = int(input("Birinchi o'zgaruvchining qiymatini kiriting: "))
```

```
b = int(input("Ikkinchi o'zgaruvchining qiymatini kiriting: "))
```

```
a = a + b
```

```
b = a - b
```

```
a = a - b
```

```
print("bu a:", a, " bu b:", b)
```

Dasturning ishini tushuntirish:

1. Foydalanuvchi ikkita o'zgaruvchining qiymatini kiritadi va ular a va b o'zgaruvchilarga yoziladi.

2. a va b yig'indisi a o'zgaruvchiga yoziladi.

3. b o'zgaruvchisi a o'zgaruvchisi (hozirda a va b boshlang'ich qiymatlar yig'indisini o'z ichiga oladi) va b o'zgaruvchisi o'rtasidagi farq yoziladi. Shunday qilib, dastlab a o'zgaruvchisi tomonidan kiritilgan qiymat b o'zgaruvchiga o'tkazildi.

4. Endi a o'zgaruvchisidan chiqaring (dastlab kiritilgan qiymatlarning yig'indisi hali ham mavjud) b o'zgaruvchining joriy qiymati (dastlab a o'zgaruvchisi yozilgan). Shunday qilib, dasturning boshida faqat b o'zgaruvchiga yozilgan qiymat a o'zgaruvchisida qoladi.

5. Va keyin biz olingan qiymatlarni ekranda namoyish etamiz.

Dastur natijalari



Case 1

1-holat

Birinchi o'zgaruvchining qiymatini kiriting: 3

Ikkinchi o'zgaruvchining qiymatini kiriting: 5

5 b: 3

Case 2

2-holat

Birinchi o'zgaruvchining qiymatini kiriting: 56

Ikkinchi o'zgaruvchining qiymatini kiriting: 25

25 b: 56

Tarjimonning eslatmasi

Yuqorida tavsiflangan hamma narsa (2, 3, 4-bandlarda) Python-da bitta satrda yozilishi mumkin. Va bu ushbu dasturlash tilining asosiy afzalliklaridan biridir.

```
print(input("Введите значение первой переменной: "))
```

```
print(input("Введите значение второй переменной: "))
```

```
b = b, a # собственно здесь и происходит обмен значениями
```

```
print("a это:",a," b это:",b)
```

Berilgan n sonida n+nn+nnn sonini hisoblash

vazifaning tavsifi

Dastur n sonini oladi va keyin n + nn + nnn kabi yig'indini qaytaradi. Bu erda nn va nnn n sonli kasrlar bilan yozilgan 2 yoki 3 marta takrorlangan n raqamini bildiradi. Masalan, n = 5 bo'lsa, nn va nnn mos ravishda 55 va 555 bo'ladi.

vazifaning yechimi

Biz sonni hisoblaymiz va uni n o'zgaruvchiga yozamiz.

2. Ip turidagi o'zgaruvchini yarataylik, unga n (oldin int turining butun sonini) qiymatini yozamiz.

3. Ushbu qatorni o'ziga qo'shib qo'ying (bu operatsiya ikkita satrni birlashtirish yoki birlashtirish deb ataladi) va natijani yangi o'zgaruvchiga yozing.

4. So'ngra yana oxirgi o'zgaruvchiga birinchi qatorni qo'shing va natijani yangi o'zgaruvchiga yozing.

5. Keyinchalik, int funksiyasi yordamida so'nggi ikkita o'zgaruvchimizni butun son turiga o'tkazamiz.

6. Nihoyat, natijani yangi o'zgaruvchiga yozish orqali uchta int o'zgaruvchimizni hammasini qo'shing.

7. Endi biz ushbu natijani bosib chiqarish funksiyasi yordamida chop etamiz.

Dasturning manba kodi

Quyida ushbu dasturning manba kodi va uning ish natijalari keltirilgan.

```
n = int(input(" n: sonin kiriting "))
```

```
temp = str(n)
```

```
t1 = temp + temp
```

```
t2 = temp + temp + temp
```

```
comp = n + int(t1) + int(t2)
```

```
print("natija teng:", comp)
```

Dasturning qanday ishlashini tushuntirish

1. Raqamni kiriting va uni n o'zgaruvchisiga saqlang.

2. Butun son turi mag'lubiyatga tashlanadi va yangi o'zgaruvchan temp saqlanadi.

3. Temp o'zgaruvchisidagi satr o'ziga qo'shiladi va natija t1 o'zgaruvchiga yoziladi.

4. Temp o'zgaruvchisidagi qatorga qo'shimcha ravishda, faqat oldingi xatboshidan farqli o'laroq, ikki marta qo'shiladi. Natijada t2 o'zgaruvchiga joylashtiriladi.

Keyin t1 va t2 o'zgaruvchilar butun songa o'tkaziladi va n o'zgaruvchiga qo'shiladi. Natijada o'zgaruvchiga yoziladi.

g. Ushbu natija ko'rsatiladi.

Ushbu dastur bajarilishining natijalari:

Ushbu misol 1:

Sonini kiriting: 5

Natija teng: 615

Ushbu misol 2:

Sonini kiriting: 20

Natija teng: 204060

Ushbu dastur berilgan sonni teskari tartibda yozish dasturi

Nazifaning tavsifi

Ushbu dastur raqamni kirish sifatida qabul qiladi va uni "teskari" shaklda aks ettiradi (ya'ni kiritilgan raqamni tashkil etadigan raqamlar, teskari tartibda yoziladi).

Muammoning echimi

1. Birinchidan, raqamni o'qing va o'zgaruvchiga yozing.

2. while tsikli yordamida biz sonning har bir raqamini alohida ajratamiz va natijada "teskari" raqamni yangi o'zgaruvchiga yozamiz.

3. Natijada paydo bo'lgan qiymatni ekranda aks ettiramiz.

4. Oxiri.

Ushbu dasturning asxodniy kod programmi

```
n = int(input("sonni kiriting: "))
```

```
while(n > 0):
```

```
    dig = n % 10
```

```
    rev = rev * 10 + dig
```

```
n = n // 10
```

```
print("raqamlari teskari tartibda bo'lgan son:", rev)
```

Dasturning qanday ishlashini tushuntirish

1. Birinchidan, raqamni kiriting va uni n o'zgaruvchisiga yozing.
2. Biz while tsiklidan foydalanamiz va modul operatoridan foydalanib (natijada butun son bo'linishining qolgan qismini beradi) biz sonimizning oxirgi raqami olamiz.
3. Ushbu raqam yangi o'zgaruvchiga yozilgan.
4. Keyin raqamning oxirgi raqami 10 ga bo'linish yo'li bilan o'chiriladi.
5. Shundan so'ng, protsedura barcha raqamlardan tugamaguncha takrorlanadi. N ga aylanganda tsikl ishlashni to'xtatadi.
6. Keyin ekranda "teskari" raqam ko'rsatiladi.

Dastur natijalari

Case 1:

Sonni kiriting: 124

Raqamlari teskari tartibda bo'lgan son: 421

Case 2:

Sonni kiriting: 4538

Raqamlari teskari tartibda bo'lgan son: 8354

Tarjimonning izohi

Python-dagi string xususiyatidan foydalangan holda, hozirgacha qilingan barcha ishlarni ancha qisqa muddatlarda amalga oshirish mumkin. Va bu yana bir b... tilning moslashuvchanligi va ixchamligi haqidi isbotidir.

```
n = input() #son darrov satr formatida eslab qolinadi
```



```
v = n[::-1] #satr invertirlanadi
```

```
print("Raqamlari teskari tartibdagi son:",rev)
```

Yoki umuman bitta satrda:

```
print("raqamlari teskari tartibda bo'lgan son:", input()[::-1])
```

Son ishorasini Tekshiruvchi dastur

Mazifaning tavsifi

Bu dastur raqamni kirish sifatida qabul qiladi va uning ijobiy yoki salbiy ekanligini tekshiradi.

Muammoning echimi

1. Raqamning qiymati o'zgaruvchiga o'qiladi va yoziladi.

2. Agar belgini aniqlash uchun ishlatilsa, shartli operator.

3. Oxiri.

Dasturning manba kodi

Dasturda esa budasturning manba kodi va uning ishining natijasi ko'rsatilgan.

```
n = int(input("Sonni kiriting: "))
```

```
if n > 0:
```

```
    print("Son musbat")
```

```
else:
```

```
    print("Son manfiy")
```

Dasturning qanday ishlashini tushuntirish

1. Foydalanuvchi raqam kiritadi va u n o'zgaruvchiga yoziladi.

2. Kiritilgan raqam 0 dan katta yoki kichikligini tekshirish uchun if operatoridan foydalaniladi.

3. Agar raqam 0 dan katta bo'lsa, unda "Raqam ijobiy" satri ko'rsatiladi.

4. Agar raqam 0 dan kam bo'lsa, unda "Raqam manfiy" qatori ko'rsatiladi.

5. Oxiri.

Dastur natijalari

Misol 1:

Sonni kiriting: 45

Musbat SON

Misol 2:

Sonni kiriting: -30

Manfiy son

Baholari bo'yicha talabanning o'zlashtirishini aniqlash dasturi

Vazifaning tavsifi

Dastur beshta fan bo'yicha baholarni kirish sifatida qabul qiladi va ma'lum mezonlarga muvofiq o'quv ko'rsatkichlarini belgilaydi.

Muammoning echimi

1. Biz beshta qiymatni kirish sifatida qabul qilamiz va ularni besh xil o'zgaruvchiga yozamiz.
2. Ushbu taxminlarning o'rtacha qiymatini toping.
3. If-elif-else shartli operatoridan foydalanib, biz bunday baholarga ega bo'lgan talabanning ishlashning besh darajasidan qaysi biriga ega ekanligiga qaror qilamiz.
4. Oxiri.

Manba kodi

Quyida talabalarning yutuqlarini uning baholari bo'yicha aniqlash dasturining manba kodi keltirilgan. Dastur natijalari ham quyida keltirilgan.

.

```
sub1 = int(input("Birinchi fan bo'yicha bahoni kiriting: "))
```

```
sub2 = int(input("Ikkinchi fan bo'yicha bahoni kiriting: "))
```

```
sub3 = int(input("Uchinchi fan bo'yicha bahoni kiriting: "))
```



```
sub4 = int(input("To'rtinchi fan bo'yicha bahoni kiriting: "))
```

```
sub5 = int(input("Введите оценку по пятому предмету: "))
```

```
avg = (sub1+sub2+sub3+sub4+sub4)/5
```

```
if avg >= 90):
```

```
print("A darajasi:")
```

```
if avg >= 80 & avg < 90):
```

```
print("B darajasi:")
```

```
if avg >= 70 & avg < 80):
```

```
print("C darajasi: C ")
```

```
if avg >= 60 & avg < 70):
```

```
print("D B darajasi: ")
```

```
else:
```

```
print(" F B darajasi:")
```

asturning qanday ishlashini tushuntirish

Foydalanuvchi beshta turli xil reytinglarni kiritadi, ular alohida o'zgaruvchilarda qayd etiladi.

3. Bundan tashqari, o'rtacha qiymatni aniqlash uchun ushbu taxminlar yig'ilib, taxminlar soniga, ya'ni 5-raqamga bo'linadi.

4. Agar o'rtacha 90 balldan yuqori bo'lsa, u holda ekran: "A daraja" ko'rsatiladi.

5. Agar o'rtacha 80 dan oshsa, lekin 90 dan kam bo'lsa, unda ekran ko'rinadi: "B darajasi".

6. Agar o'rtacha 70 dan oshsa, lekin 80 dan kam bo'lsa, unda ekran ko'rinadi: "S daraja".

7. Agar o'rtacha 60 dan oshsa, lekin 70 dan kam bo'lsa, unda ekran ko'rinadi: "D darajasi".

7. Agar o'rtacha 60 dan kam bo'lsa, unda ekranda "F darajasi" ko'rinadi.

Dastur natijalari

Misol 1:

Birinchi fan bo'yicha bahoni kiriting: 85

Ikkinchi fan bo'yicha bahoni kiriting: 95

Uchinchi fan bo'yicha bahoni kiriting: 99

To'rtinchi fan bo'yicha bahoni kiriting: 93

Beshinchi fan bo'yicha bahoni kiriting: 100

Darajasi: A

Misol 2:

Birinchi fan bo'yicha bahoni kiriting: 81

Ikkinchi fan bo'yicha bahoni kiriting: 72

Uchinchi fan bo'yicha bahoni kiriting: 94

To'rtinchi fan bo'yicha bahoni kiriting: 85

Beshinchi fan bo'yicha bahoni kiriting: 80

Darajasi: B

TALABALARNING YUTUQLARINI UNING BAHOLARI BO'YICHA ANIQLASH DASTURI

Vazifaning tavsifi

Dastur beshta fan bo'yicha baholarni kirish sifatida qabul qiladi va ma'lum mezonlarga muvofiq o'quv ko'rsatkichlarini belgilaydi.

Muammoning echimi

1. Biz beshta qiymatni kirish sifatida qabul qilamiz va ularni besh x o'zgaruvchiga yozamiz.

Ushbu taxminlarning o'rtacha qiymatini toping.

β. If-elif-else shartli operatoridan foydalanib, biz bunday ko'rsatkichlarga ega bo'lgan talabanning ishlashning besh darajasidan qaysi biriga ega ekanligiga qaror qilamiz.

Oxiri.

Manba kodi

Quyida talabalarning yutuqlarini uning baholari bo'yicha aniqlash dasturining manba kodi keltirilgan. Dastur natijalari ham quyida keltirilgan.

```
b1 = int(input("Birinchi fan bo'yich bahoni kiriting: "))
```

```
b2 = int(input("Ikkinchi fan bo'yich bahoni kiriting: "))
```

```
b3 = int(input("Uchinchi fan bo'yich bahoni kiritingВведите оценку по третьему предмету: "))
```

```
sub4 = int(input("to'rtinchi fan bo'yicha bahoingizni kiriting: "))
```

```
sub5 = int(input("beshinchi fan bo'yicha bahoni kiriting "))
```

```
avg = (sub1+sub2+sub3+sub4+sub4)/5
```

```
if(avg >= 90):
```

```
print("Daraasi: A")
```

```
elif(avg >= 80 & avg < 90):
```

```
print("Darajasi: B")
```

```
elif(avg >= 70 & avg < 80):
```

```
print("Darajasi: C")
```

```
elif(avg >= 60 & avg < 70):
```

```
print("Daraja: C")
```

```
else:
```



```
print("Daraja: F")
```

Dasturning qanday ishlashini tushuntirish

1. Foydalanuvchi beshta turli xil reytinglarni kiritadi, ular alohida o'zgaruvchilarda qayd etiladi.
2. Bundan tashqari, o'rtacha qiymatni aniqlash uchun ushbu taxminlar yig'ilib o'rtacha taxminlar soniga, ya'ni 5-raqamga bo'linadi.
3. Agar o'rtacha 90 balldan yuqori bo'lsa, u holda ekran: "A daraja" ko'rsatiladi.
4. Agar o'rtacha 80 dan oshsa, lekin 90 dan kam bo'lsa, unda ekran ko'rinadi: "B darajasi".
5. Agar o'rtacha 70 dan oshsa, lekin 80 dan kam bo'lsa, unda ekran ko'rinadi: "C daraja".
6. Agar o'rtacha 60 dan oshsa, lekin 70 dan kam bo'lsa, unda ekran ko'rinadi: "D darajasi".
7. Agar o'rtacha 60 dan kam bo'lsa, unda ekranda "F darajasi" ko'rinadi.

Dastur natijalar

Misol 1:

Birinchi fan bo'yicha bahoni kiriting: 85

Ikkinchi fan bo'yicha bahoni kiriting: 95

Uchinchi fan bo'yicha bahoni kiriting: 99

Введите оценку по четвертому предмету: 93

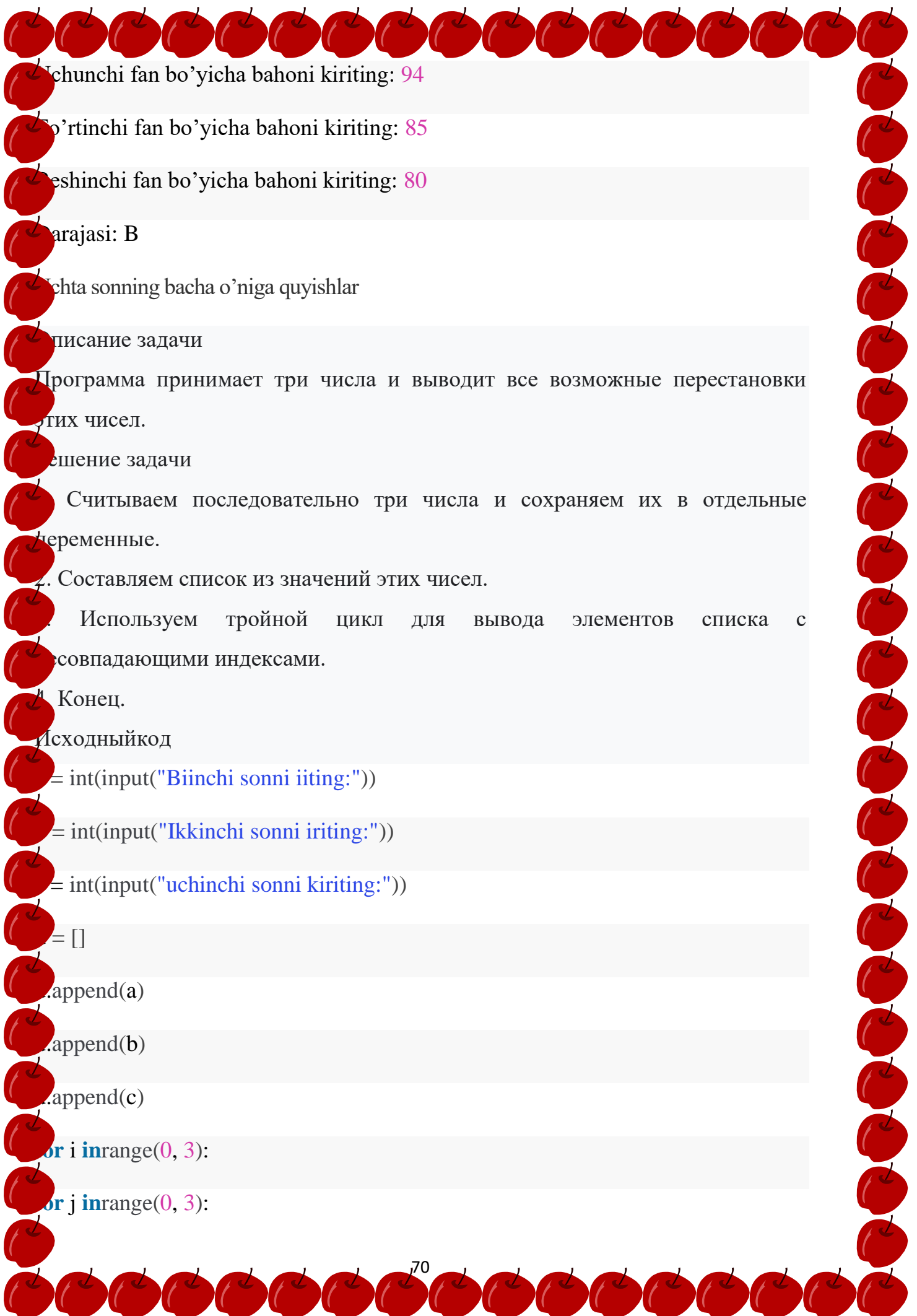
Введите оценку по пятому предмету: 100

Уровень: A

Misol 2:

Birinchi fan bo'yicha bahoni kiriting: 81

Ikkinchi fan bo'yicha bahoni kiriting: 72



1. Birinchi fan bo'yicha bahoni kiriting: 94

2. Ikkinchi fan bo'yicha bahoni kiriting: 85

3. Uchinchi fan bo'yicha bahoni kiriting: 80

4. Darajasi: B

5. Mavzu: Dastur sonning barcha o'ringa quyishlar

6. Topshiriq: Yozma

7. Masala: Dastur sonning barcha o'ringa quyishlar. Dastur sonning barcha o'ringa quyishlar. Dastur sonning barcha o'ringa quyishlar.

8. Masala: Dastur sonning barcha o'ringa quyishlar.

9. Masala: Dastur sonning barcha o'ringa quyishlar. Dastur sonning barcha o'ringa quyishlar. Dastur sonning barcha o'ringa quyishlar.

10. Masala: Dastur sonning barcha o'ringa quyishlar.

11. Masala: Dastur sonning barcha o'ringa quyishlar. Dastur sonning barcha o'ringa quyishlar. Dastur sonning barcha o'ringa quyishlar.

12. Masala: Dastur sonning barcha o'ringa quyishlar.

13. Masala: Dastur sonning barcha o'ringa quyishlar.

```
a = int(input("Birinchi sonni kiriting:"))
```

```
b = int(input("Ikkinchi sonni kiriting:"))
```

```
c = int(input("Uchinchi sonni kiriting:"))
```

```
l = []
```

```
l.append(a)
```

```
l.append(b)
```

```
l.append(c)
```

```
for i in range(0, 3):
```

```
    for j in range(0, 3):
```

```
for k in range(0, 3):
```

```
if(i != j & j != k & k != i):
```

```
print(d[i], d[j], d[k])
```

Dasturning qanday ishlashini tushuntirish

1. Foydalanuvchi ketma-ket uchta raqamni kiritadi.
2. Ushbu raqamlar append usuli yordamida ro'yxatga qo'shiladi.
3. Keyin, 0 dan 2 gacha bo'lgan ko'chadan uchun uchta foydalanib, biz uchta raqamning to'liq ro'yxatini ishga tushiramiz.
4. Ro'yxatlar indeksleri bir-biriga teng bo'lmagan holda, ushbu ko'rsatkichlarga mos keladigan qiymatlarni ekranda ko'rsating.

Dastur natijalari

Miso 1:

Birinchi sonni kiriting:1

Ikinchi sonni kiriting:2

Uchinchi sonni kiriting:3

123

132

213

231

312

321

Miso 2:

Birinchi sonni kiriting:5

Ikkinchi sonni kiriting:7

Uchinchi sonni kiriting:3

73

37

53

35

57

75

BERILGAN DIAPAZONDA BARCHA TOQ SONLARNI CHIQUARISH DASTURINI TUZING

Nazifaning tavsifi

Dastur kirish sifatida ikkita raqamni oladi, ular diapazon chegaralarini belgilaydi va shu diapazondagi barcha toq sonlarni chiqaradi.

Uammoning echimi

Biz kirish sifatida berilgan diapazonni aniqlaydigan ikkita sonni qabul qilamiz va ularni har xil o'zgaruvchilarga yozamiz.

Ushbu diapazondan o'tish uchun for loopdan foydalanamiz.

So'ngra biz par-paritni tekshirish uchun if iborasidan foydalanamiz va keyin toq sonlarni ekranga chop etamiz.

4. Oxiri.

Manba kodi

Quyida berilgan oraliqdagi barcha g'alati raqamlarni chiqarish uchun manba kodi keltirilgan. Dastur natijalari ham quyida keltirilgan.

```
lower = int(input("Pastki diapazon chgaani kiriting:"))
```

```
upper = int(input("yuqori diapason chegarasini kiriting:"))
```

```
for i in range(lower, upper+1):
```

```
if(i % 2 != 0):
```




print(i)

Dasturning qanday ishlashini tushuntirish

1. Foydalanuvchi ikkita raqamni kiritadi, ular alohida o'zgaruvchilarga yozilgan va diapazonning pastki va yuqori chegaralari.
2. for ko'chadan foydalanib, berilgan diapazondagi barcha raqamlar bo'yicha takrorlaymiz.
3. Keyinchalik, if iborasidan foydalanamiz. Ushbu operator ichidagi ifoda, qoldiq 0 ga tengligini tekshiradi.
4. Agar qoldiq 0 ga teng bo'lmasa, biz ushbu raqamni ekranda ko'rsatamiz.

Dastur natijalari

MISO 1:

DIAPAZONNING PASTKKI CHEGARASINI KIRITING :1

DIAPAZONNING YUQORI CHEGARASINI KIRITING:16

1

3

5

7

9

11

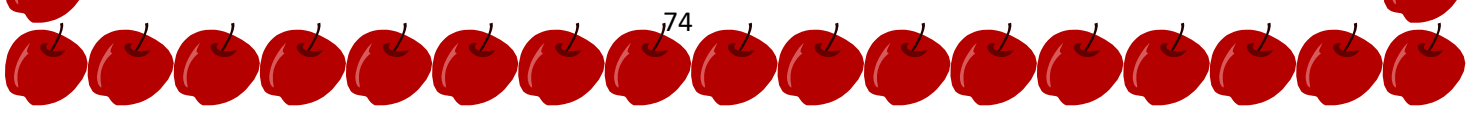
13

15



MISOL 2:
HAPAZONNING PASTKI CHEGARASININ KIRITING:150
HAPAZONNING YUQORI CHEGARASININ KIRITING:180
151
153
155
157
159
161
163
165
167
169
171
173
175
177
179

BERILGAN SONNI BARCHA RAQAMLAR YIG'INDISINI TOPISH DASTURI



Vazifaning tavsifi

Ushbu dasturda kiritilgan raqamni tashkil etadigan barcha raqamlarning yig'indisi ko'rsatilishi kerak.

Muammoning echimi

1. Kiritilgan raqamni oling va o'zgaruvchiga yozing.
2. while tsikli yordamida berilgan sondan har bir raqamni chiqarib oling va ularning yig'indisini alohida o'zgaruvchiga yozing.
3. Olingan natijani ekranga chiqaramiz.
4. Oxiri.

Manba kodi

Quyida berilgan sonni tashkil etuvchi barcha raqamlarning yig'indisini chiqarish uchun manba kodi keltirilgan. Dastur natijalari ham quyida keltirilgan.

```
n = int(input("Sonni kiriting:"))
```

```
tot = 0
```

```
while(n > 0):
```

```
    dig = n % 10
```

```
    tot = tot + dig
```

```
    n = n // 10
```

```
print("Raqamlar yig'indisi:", tot)
```

Dasturning qanday ishlashini tushuntirish

1. Foydalanuvchi o'zgaruvchiga yozilgan raqamni kiritadi.
2. while loopidan foydalanamiz. Raqamlar sonidan boshlab modul operatori yordamida raqamlardan olinadi.
3. Keyin, biz bu raqamni alohida o'zgaruvchiga yozamiz va keyin tsikl davomiyatiga qolgan raqamlarni qo'shamiz.
4. Summa yig'ilgandan so'ng, raqam butun bo'linish operatori yordamida "yo'q qilinadi".

Raqam 0 ga teng bo'lganda (barcha raqamlar qayta ishlangan va "yo'q qilingan"), tsikl tugaydi.

3. Shundan so'ng olingan summa ekranda aks etadi.

Dastur natijalari

Misol 1:

Sonni kiriting: 1892

Raqamlar yig'indisi : 20

Misol 2:

Sonni kiriting: 157

Raqamlar yig'indisi: 13

Butun sonni eng kichik bo'luvchisini topish uchun dastur

vazifaning tavsifi

Ushbu dastur kirish sifatida butun sonni oladi va uning eng kichik bo'linmasini chiqaradi.

Alqammoning echimi

Biz kirish sifatida foydalanuvchi tomonidan kiritilgan butun sonni qabul qilamiz va uni o'zgaruvchiga yozamiz.

1. Qiymatlari 2 dan berilgan butun songacha bo'lgan for loopidan foydalanamiz.

2. Agar ushbu diapazondagi raqam berilgan sonni qoldiqsiz ajratsa, u maxsus berilgan ro'yxatga qo'shiladi.

Keyin ro'yxat saralanadi va undan minimal raqam ko'rsatiladi.

3. Oxiri.

Manba kodi

Quyida eng kichik bo'luvchini topish va chiqarish uchun manba kodi keltirilgan.

Dastur natijalari ham quyida keltirilgan.

```
n = int(input("Butun sonni kiriting:"))
```

```
a = []
```

```
for i in range(2, n+1):
```

```
if(n % i == 0):
```

```
a.append(i)
```

```
a.sort()
```

```
print("Eng kichik bo'luvchi:", a[0])
```

Dasturning qanday ishlashini tushuntirish

1. Foydalanuvchi butun sonni kiritadi, u o'zgaruvchiga yoziladi.
2. for loopini kiritilgan raqamga 2 dan o'rnating.
3. Agar bo'linishning qoldig'i 0 bo'lsa, demak, bu raqam bo'luvchi hisoblanadi. Ushbu tekshirish% modulli operator yordamida amalga oshiriladi.
4. Shundan so'ng, ushbu bo'luvchi ro'yxatga qo'shiladi.
5. Keyin ro'yxat saralanadi va uning minimal elementi ko'rsatiladi.

Dastur natijalari

Misol 1:

Butun sonni kiriting:75

Eng kichik bo'luvchi teng: 3

Misol 2:

Butun sonni kiriting:64

Eng kichik bo'luvchi teng: 2

Sondagi raqamlar sonini topish dasturi

Vazifaning tavsifi

Dastur raqamni oladi va undagi raqamlar sonini chiqaradi.



1. Muammoning echimi

1. Biz butun sonning qiymatini olamiz va uni o'zgaruvchiga yozamiz.

2. Biz while tsiklidan foydalanamiz va sonni bo'linish operatoridan foydalanib sonning har bir raqamini oxirgisidan boshlaymiz va tsiklning har bir takrorlanishida biz maxsus yaratilgan o'zgaruvchini ko'paytiramiz (tsikl hisoblagichi deb ataladi). bittadan. Boshida kiritilgan raqam 0 ga teng bo'lgandan so'ng, tsikl o'z ishini to'xtatadi.

3. Biz ushbu hisoblagichning qiymatini ekranda aks ettiramiz.

4. Oxiri.

Manba kodi

Quyida berilgan sondagi raqamlar sonini hisoblash uchun manba kodi keltirilgan. Ushbu natijalari ham quyida keltirilgan.

```
n = int(input("Sonni kiriting:"))
```

```
count = 0
```

```
while(n > 0):
```

```
    count = count + 1
```

```
    n = n // 10
```

```
print("Raqamlar soni teng=", count)
```

Пример 1:

Введите число: 123

Количество цифр равно: 3

2. Masurning qanday ishlashini tushuntirish

1. Biz foydalanuvchi tomonidan kiritilgan raqamni n o'zgaruvchiga yozamiz.

2. O'zgaruvchilar sonini o'rnating va 0 qiymati bilan boshlang.

3. Biz while tsiklidan foydalanamiz va sonning har bir raqamini oxiridan boshlab "o'q qilish" uchun butun bo'linish operatoridan foydalanamiz.

4. Tsiklning har bir takrorlanishida o'zgaruvchilar soni 1 ga ko'paytiriladi.



5. Raqam tugashi va n soni 0 ga tenglashishi bilan tsikl o'z ishini to'xtatadi.

6. O'zgaruvchilar sonini ekranda ko'rsatish.

Dastur natijalari

Misol 2:

Sonni kiriting: 1892

Raqamlar soni: 4

Примечание переводчика

Надо заметить, что опять эту задачу можно решить намного короче, если использовать методы строк.

```
print("Количество цифр равно:", len(input("Введите число:")))
```

Здесь введенное число принимается как строка и мы просто выводим ее длину.

1 va 50 oralig'ida 2 va 3 ga bo'linmaydigan barcha butun sonlarni chiqaradigan dastur

ISHLANGAN ALGORITMALAR

Vazifaning tavsifi

Ushbu dastur 1 dan 50 gacha bo'lgan 2 yoki 3 ga bo'linmaydigan barcha butun sonlarni chop etishi kerak.

Muammoning echimi

1. Biz for for loopini 1 dan 51 gacha ishlatamiz.
2. Loop tanasida if operatoridan foydalanib, diapazondagi har bir raqam 2 va 3 ga bo'linishini tekshiramiz.
3. Agar raqam 2 yoki 3 ga bo'linmasa, ya'ni bizning shartlarimizga javob bersa, unda biz uni ekranda namoyish qilamiz.
4. Oxiri.



Manba kodi

Quyida 2 va 3 ga bo'linmaydigan 1 dan 50 gacha bo'lgan barcha butun sonlarni aniqlash uchun manba kodi keltirilgan. Dastur natijalari ham quyida keltirilgan.

```
for i in range(1, 51):
    if i % 2 != 0 & i % 3 != 0:
        print(i)
```

Dasturning qanday ishlashini tushuntirish

Tsikl (1, 51) oralig'ida ko'rsatilgan, chunki 51 qo'shilmagan.

if ifodasi tanasidagi ifoda, bo'linishning qoldig'ini hisoblaydigan modul operatori yordamida sonning 2 yoki 3 ga bo'linishini aniqlaydi.

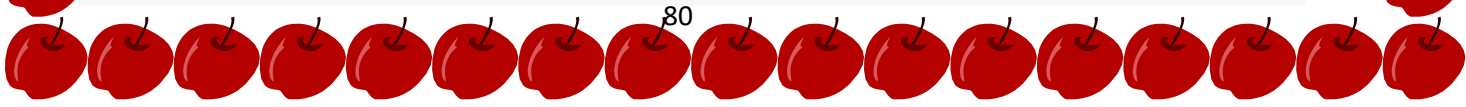
Agar 2 va 3 ga bo'linishdan keyin qoldiqlar 0 ga teng bo'lmasa, unda raqam moslashda 2 va 3 ga bo'linmaydi.

Bunday holda biz raqamni ekranda ko'rsatamiz.

Dastur natijalari

Misol 1:

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
```



29

31

35

37

41

43

47

49

DASTUR N SONNI O'QIB N TA SONNIYIG'INDISINI HISOBLAYDI: $1 + 2 +$

$+ \dots + n$ Vazifaning tavsifi

Dastur kirish uchun n raqamini oladi, $1 + 2 + 3 + \dots + n$ qatorlarining yig'indisini hisoblab chiqadi va barchasini ekranda aks ettiradi.

Muammoning echimi

1. Biz butun sonning qiymatini o'qiyamiz va uni n o'zgaruvchiga yozamiz.
2. Bo'sh ro'yxat yarating.
3. Biz for loopdan foydalanamiz, unda i loop o'zgaruvchisi 1 dan n gacha bo'lgan qiymatlarni qabul qiladi.
4. Tsikl ishlayotganda, i qiymatlari va "+" belgisini ko'rsating, shuningdek, avval yaratilgan ro'yxatga i qiymatlarini qo'shing.
5. Keyin biz ro'yxatning barcha elementlari yig'indisini hisoblaymiz.
6. "=" belgisini, so'ngra hisoblangan miqdorni ko'rsating.
7. Tugatish.

Manba kodi

Quyida n raqamini qabul qiladigan, $1 + 2 + 3 + \dots + n$ qatorlarining yig'indisini hisoblaydigan va barchasini ekranda ko'rsatadigan dasturning manba kodi keltirilgan. Dastur natijalari ham quyida keltiriladi.



```
n = int(input("Sonni iiting: "))
```

```
a = []
```

```
for i in range(1, n+1):
```

```
    print(i, sep=" ", end=" ")
```

```
    if i < n:
```

```
        print("+", sep=" ", end=" ")
```

```
        a.append(i)
```

```
print("=", sum(a))
```

asturning qanday ishlashini tushuntirish

1. Foydalanuvchi n o'zgaruvchiga yozilgan raqamni kiritadi.

2. Bo'sh ro'yxat tuziladi a.

3. for loopida i tsikli o'zgaruvchisi 1 dan n gacha o'zgaradi (n + 1 qo'shilmaydi).

4. Tsiklning har bir takrorlanishida i qiymati '+' belgisi bilan birga ko'rsatiladi.

5. Shuningdek, ro'yxatga a qiymati qo'shiladi.

6. Tsikl tugagandan so'ng '=' belgisi va ro'yxat yig'indisi ko'rsatiladi, bu sum ()

funksiyasi yordamida hisoblanadi.

astur natijalari

Misol 1:

```
sonni iiting: 4
```

```
1 + 2 + 3 + 4 = 10
```

Miso 2:

```
sonni iiting: 5
```

```
1 + 2 + 3 + 4 + 5 = 150
```

TRATOSFEN G'ALVIRI

vazifaning tavsifi



Ushbu dastur "Eratosfen elagi" algoritmi yordamida berilgan (0 dan n gacha) oralig'idagi barcha tub sonlarni chiqarishi kerak.

Muammoning echimi

1. Biz diapazonning yuqori chegarasini belgilaydigan qiymatni qabul qilamiz va uni n o'zgaruvchiga yozamiz.
2. Biz o'zgaruvchan elakni ("elak") 2 dan n gacha bo'lgan raqamlar to'plami bilan boshlaymiz.
3. Biz elak to'plami bo'shshganda tugaydigan, while ko'chadan foydalanamiz.
4. Ushbu to'plamdagi minimal son (birinchi takrorlashda u 2 bo'ladi) har doim haqiqiy tub ekanligini hisobga olaylik.
5. Biz ushbu raqamni ekranda namoyish etamiz.
6. Keyin, ushbu raqamni ko'paytiradigan barcha raqamlar bilan birga o'chirib tashlaymiz (belgilangan diapazonda).
7. Elak to'plami bo'sh bo'lguncha shunday qilishni davom eting.
8. Tugatish

```
n = int(input("Введите верхнюю границу диапазона: "))
```

```
sieve = set(range(2, n+1))
```

```
while sieve:
```

```
    prime = min(sieve)
```

```
    print(prime, end = "\t")
```

```
    sieve -= set(range(prime, n+1, prime))
```

Manba kodi

Quyida "Eratosfen elagi" deb nomlangan algoritm yordamida ma'lum bir diapazon oralig'idagi barcha tub sonlarni chiqarish uchun manba kodi keltirilgan. Dasturning natijalari ham quyida keltirilgan. Dasturning qanday ishlashini tushuntirish

1. Foydalanuvchi diapazonning yuqori chegarasiga kiradi va u n o'zgaruvchiga yoziladi.

Elak o'zgaruvchisini 2 dan n gacha bo'lgan barcha sonlar to'plami bilan boshlang. "To'plam" turi o'rnatilgan funktsiya bilan belgilanadi va diapazondagi barcha raqamlar diapazon funktsiyasi yordamida belgilanadi.

while tsikli elak to'plami bo'sh bo'lguncha ishlaydi.

Prime o'zgaruvchisi elak to'plamidan eng kichik qiymatgacha boshlanadi. Ultimos, bu har doim ham asosiy raqam bo'lib qolishini unutmang. Va bu asosiy raqam ekranda ko'rsatiladi.

Keyin elak to'plamidan bu raqam va uning ko'paytmasi bo'lgan barcha raqamlar chiriladi.

1 va 5-elementlar elak to'plami bo'sh bo'lguncha takrorlanadi, ya'ni undagi elementlar soni 0 ga teng bo'ladi.

astur natijalari

misol:

10 oralig'ining yuqori chegarasini kiriting

2 3 5 7

misol:

oralashmaning yuqori chegarasini kiriting: 15

2 3 5 7 11 13

BOB. TOSHBAQA PERSONAJLI PYTHONDA RASM CHIZISH.

Birinchi turtle dasturi

Python. Рисование.

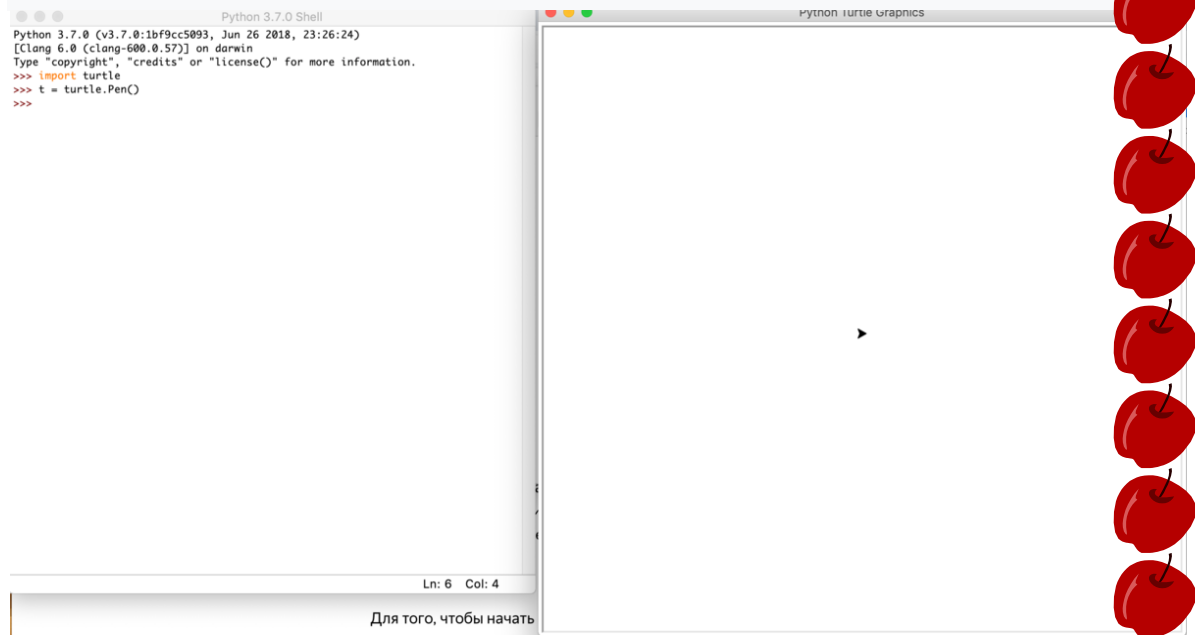


1.

Python dasturlash muhitida rasm chizish uchun toshbaqa moduli ishlatiladi.
Modul - bu foydali kodni boshqa dasturga ulash usuli, boshqa narsalar qatori, modullarda odatda funktsiyalar mavjud. Modulni ulash juda oson:

```
>>> toshbaqani import qilish
```

Chizishni boshlash uchun siz Pen funksiyasidan foydalanishingiz kerak.
Pen funktsiyasi chaqirilgandan so'ng, avtomatik ravishda tuval yaratiladi.



```
Python 3.7.0 Shell
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 26 2018, 23:26:24)
[Clang 6.0 (clang-600.0.57)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> import turtle
>>> t = turtle.Pen()
>>>
```

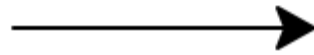
Deraza ichidagi o'qqa e'tibor bering. Bu toshbaqa.

Endi biz ekranda rasm chizadigan buyruqlar bera olamiz. Buning uchun maxsus buyruqlar mavjud. Masalan, toshbaqani oldinga siljitish uchun oldinga buyruqi ishlatiladi. qadamlar soni piksel bilan ko'rsatilgan va qavs ichida ko'rsatilgan.

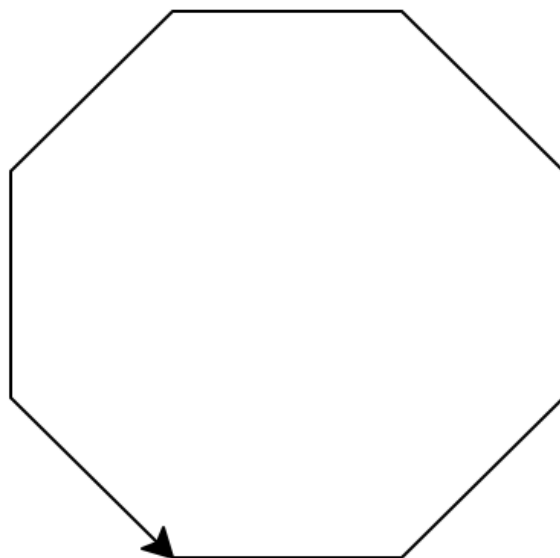
```
...
```

```
>>> t.forward(75)
```

```
>>>
```



5 piksel oldinga siljish natijasi. Piksel - bitta ekran nuqtasi. Keyin, bizning shbaqani 45 darajaga burang.



Kvadratni o'zingiz chizib oling.

Tuvalni tozalash uchun `t.reset ()` buyrug'ini yoki `t.clear ()` buyrug'ini kiriting. Shuningdek, toshbaqa `t.right` buyrug'i yordamida o'ngga burilib, `t.backward ()` buyrug'i bilan orqaga qarab harakatlanishi mumkin. `T.up ()` buyrug'i ham bor, u ruchkani tuvaldan olib tashlaydi va `t.down ()` - ruchkani tuvalga qaytaradi.

Biz quyidagi jamoalar bilan ishlaymiz:

`t.forward ()` - oldinga qadam

`t.backward ()` - orqaga qadam

`t.right ()` - o'ngga buriling

`t.left ()` - chapga buriling

`t.up ()` - qalamni ko'taring

`t.down ()` - qalamni pastga tushirish

Chizishga harakat qiling:

To'rtburchak

Ikkita to'rtburchaklar

Uchburchak

Ikki uchburchak

Toshbaqaga doir dasturlar

Python 3 (toshbaqa) yordamida oddiy grafikalarini qanday chizish mumkin

...

Men Python-ni o'rganishni boshladim va siz ajoyib chizilgan rasmlar chizishingiz mumkin bo'lgan ajoyib o'rnatilgan kutubxona toshbaqasini (toshbaqa ko'rdim, u bilan ishlash Paint-da chizishingizga juda o'xshaydi.

Biroz tushuntirib, chizishingiz mumkin bo'lgan narsalarga bir nechta misol keltirmoqchiman.

Quyidagi usullardan foydalanilgan:

`turtle.color (color)` - Kursor rangini belgilang;

`turtle.penup ()` - kursorni ko'taring, shunda kursor harakatlantirilganda chiziqlar chizilmaydi;

`turtle.pendown ()` - Biz kursorni tushiramiz, shunda kursor harakatlanganda chiziqlar chiziladi.

`turtle.pendown ()` - Biz kursorni tushiramiz, shunda kursor harakatlanganda chiziqlar chiziladi;

`turtle.goto (x, y)` - koordinatalari x va y bo'lgan nuqtaga o'ting;

`turtle.circle (radius)` - radiusi = radiusi bo'lgan aylana chizish;

`turtle.right (gradus)` - Kursorni daraja bo'yicha o'ng tomonga burang;

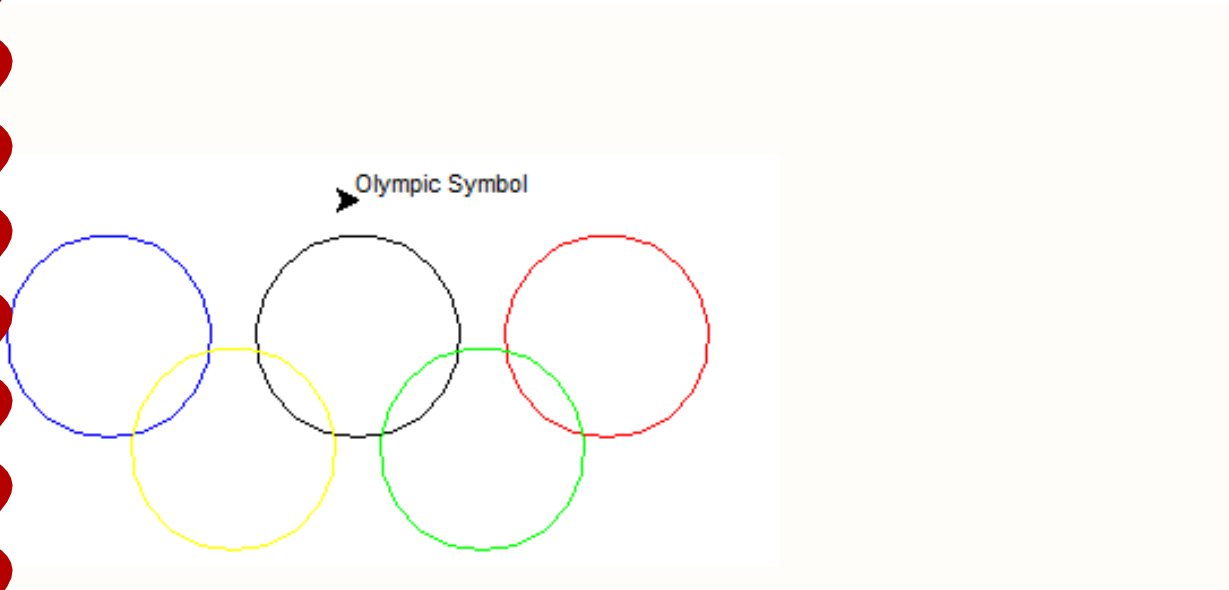
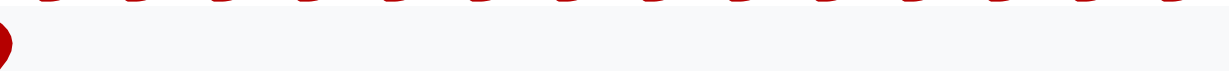
`turtle.left (graduslar)` - kursorni graduslar bo'yicha chapga burang;

`turtle.forward (uzunlik)` - uzunlik = uzunlikdagi chiziqlar chizish;

`turtle.done ()` - Ushbu usul toshbaqa yordamida barcha dasturlarni bekor qilishi kerak.

Dastlab, olimpiada ramzini chizamiz, oxirida biz quyidagilarni bilib olamiz:

`turtle.penup ()` - kursorni ko'taring, shunda kursor siljiganida hech qanday chizil chizilmaydi;



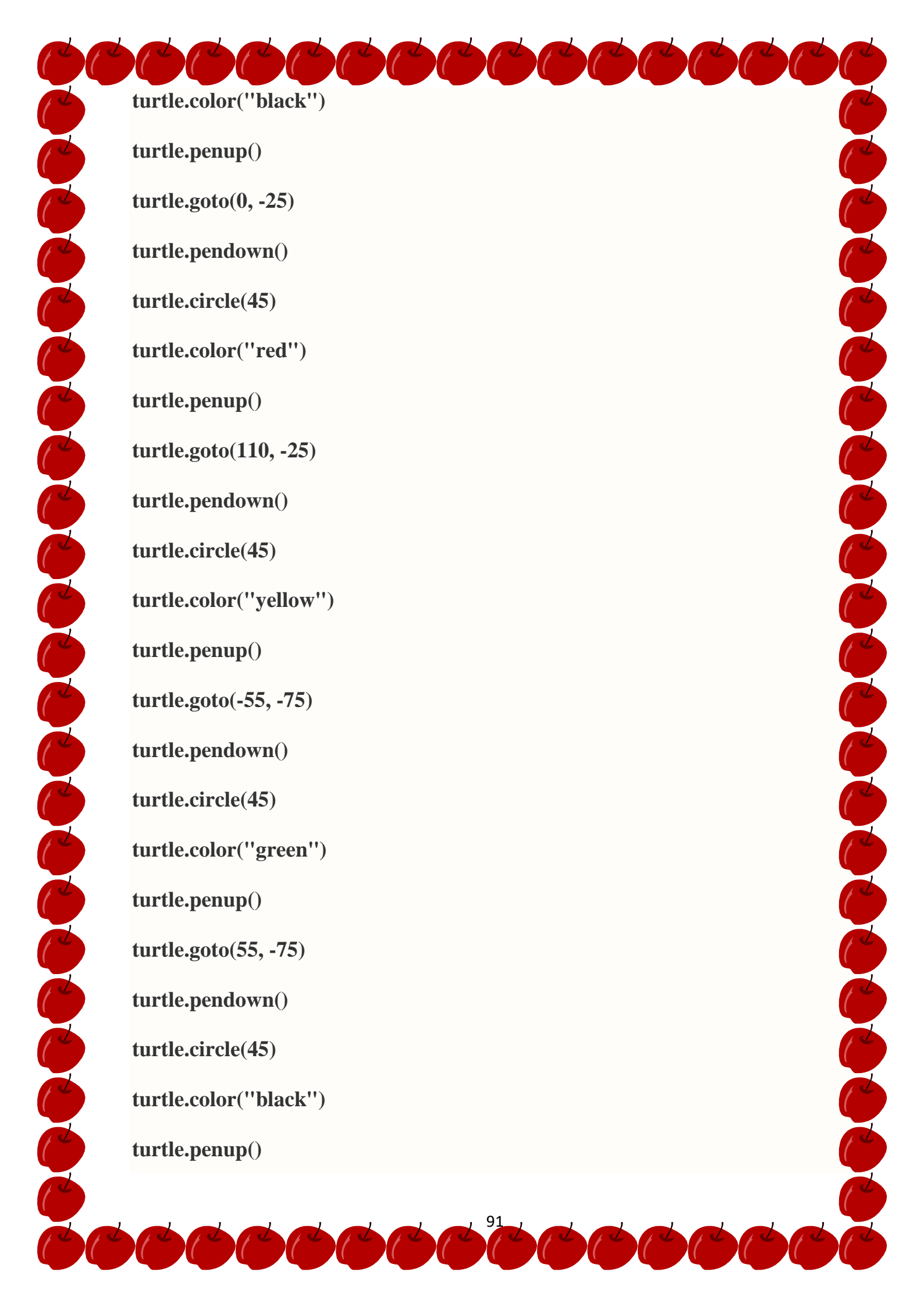
Chizish algoritmi juda oddiy:

1. Bo'yashimiz kerak bo'lgan rangni o'rnatish;
 2. Toshbaqani (kursorni) ko'taring, shunda o'tish paytida chiziqlar tortilmaydi;
 3. Bizni qiziqtirgan x va y koordinatalariga o'ting;
 4. Toshbaqani pastga tushiring (cursor);
 5. Radiusi 45 ga teng doira chizish.
- shuning uchun biz har bir doira uchun ijro etamiz. Dastur uchun kichik manba

Kodi:

```
import turtle # turtle kutubxonani bibliotekaga import qilamiz
turtle.color("blue") # toshbaqa rangini o'rnatamiz
turtle.penup() # kursorni ko'taramiz
turtle.goto(-110, -25) # kerakli koordinatalar bo'yicha yuramiz
turtle.pendown() # kursorni quyib yuboramiz
turtle.circle(45) # Рисуем круг с радиусом 45
```





```
turtle.color("black")
turtle.penup()
turtle.goto(0, -25)
turtle.pendown()
turtle.circle(45)
turtle.color("red")
turtle.penup()
turtle.goto(110, -25)
turtle.pendown()
turtle.circle(45)
turtle.color("yellow")
turtle.penup()
turtle.goto(-55, -75)
turtle.pendown()
turtle.circle(45)
turtle.color("green")
turtle.penup()
turtle.goto(55, -75)
turtle.pendown()
turtle.circle(45)
turtle.color("black")
turtle.penup()
```



```
turtle.goto(0, 80)
```

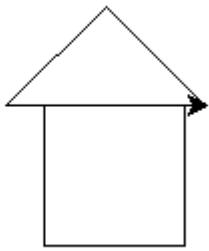
```
turtle.pendown()
```

```
turtle.write("Olympic Symbol") # Вместоещеодногоокругавыводимнадпись
```

```
"Olympic Symbol"
```

```
turtle.done()
```

aychani chizaylik juda zo'r bo'lmasada harakat qilamiz :)):



Bu yerda yana bir nechta funktsiyalar mavjud:

```
import turtle
```

```
turtle.penup()
```

```
turtle.goto(35, 35)
```

```
turtle.right(90) # o'nga 0 gradusa kursorni buramiz
```

```
turtle.pendown()
```

```
turtle.forward(70) # oldinga 70 masofani o'tamiz agarda kursor tushirilan bo'lsa,
```

urish izidan chiziq chizilgan bo'ladi

```
turtle.right(90)
```

```
turtle.forward(70)
```

```
turtle.right(90)
```





```
turtle.forward(70)
```

```
turtle.right(90)
```

```
turtle.forward(70)
```

```
turtle.penup()
```

```
turtle.goto(45, 35)
```

```
turtle.right(225)
```

```
turtle.pendown()
```

```
turtle.forward(70)
```

```
turtle.left(90)
```

```
turtle.forward(70)
```

```
turtle.left(135) # kursorni chapga 135 gradusga buringlar
```

```
turtle.forward(100)
```

```
turtle.done()
```

python haqida juda ham zo'r hujjatlar mavjud.

Pythonda konsolda rasm chizamiz:

Rasmlarni konsolda chizish uchun quyidagi [asciimatics](#) ([galereyani](#)) qo'llanish mumkin:

Bu ASCII simvoldan rasmlarni yaratadi. Unda ranglar, kursor joylashuvi va boshqalar bilan terminal imkoniyatlarini qo'llaydi. Linuxda framebuffer e.g. [fbf](#) qo'llab rasmlarni ko'rsatish mumkin

```
from colorama import init, Fore, Back, Style
```

```
init()
```



```
def console_picture():
```

```
print(Style.BRIGHT + Fore.YELLOW)
```

```
print("      **  ** ***** **      **      **  ")
```

```
print("      **  ** ***** **      **      **  **  ")
```

```
print("      ***** **      **      **      **  **  ")
```

```
print("      ***** ***** **      **      **  **  ")
```

```
print("      **  ** *      *      *      **  **  ")
```

```
print("      **  ** ***** ***** ***** **  **  ")
```

```
print("      **  ** ***** ***** ***** **  **  ")
```

```
console_picture()
```

```
input()
```

```
from colorama import init, Fore, Back, Style
```

```
init()
```

```
def console_picture():
```

```
print(Style.BRIGHT + Fore.YELLOW)
```

```
print("      **  ** ***** **      **      **  ")
```

```
print("      **  ** ***** **      **      **  **  ")
```

```
print("      ***** **      **      **      **  **  ")
```

```
print("      ***** ***** **      **      **  **  ")
```

```
print("      **  ** *      *      *      **  **  ")
```

```
print("      **  ** ***** ***** ***** **  **  ")
```

```
print("      **  ** ***** ***** ***** **  **  ")
```

```
console_picture()
```

```
input()
```

Python-dagi grafika.

Python-dagi grafikalar yordamida siz shakllar va rasmlar chizishingiz, animatsiyalar yaratishingiz va Python-da matematik hisob-kitoblarni tasavvur qilishingiz mumkin. Python dasturlarida siz grafik o'yinlardan kompyuter o'yinlarida foydalanishingiz mumkin.

Python-da grafikalar bilan ishlash uchun graphics.py modulini import qilish kerak. Python-da grafik modulni qanday o'rnatish.

Python-da grafikani boshlash uchun siz grafik oynani yaratishingiz kerak

Grafika ob'ekti = GraphWin ("Grafika uchun oyna sarlavhasi", grafika uchun oynaning kengligi, grafika uchun oynaning balandligi)

GraphWin - bu grafik ob'ektlar ko'rsatiladigan grafik maydon oynasini belgilaydigan kalit so'z.

GraphWin - bu grafik ob'ektlar ko'rsatiladigan grafik maydon oynasini belgilaydigan kalit so'z.

Ushbu funktsiyaning parametrlari - grafikalar uchun derazalarning nomi, piksellardagi derazalarning kengligi va balandligi.

Dasturni ishga tushirgandan so'ng, grafik ob'ektlar ko'rsatiladigan grafikalar uchun oyna ochiladi.

Grafika bilan barcha ishlarni biz grafik ob'ektlar orqali amalga oshiramiz.

Python-dagi grafik ob'ektlar bilan ishlashning umumiy tuzilishi

```
Graphic_Object.Call_command ()
```

Python-dagi grafik dasturning umumiy tuzilishi.

```
# grafik kutubxonani import qilish
```

```
grafik importdan *
```

```
# grafika uchun oyna yaratish
```

```
win = GraphWin ("Grafika uchun oyna", 400, 400)
```

```
# ... barcha moslamalarni chizish ...
```

```
win.getMouse () # sichqoncha tugmasi bosilishini kuting
```

```
win.close () # grafika uchun oynani yoping
```

Ushbu dasturda biz win grafik oynasi ob'ektini aniqladik va uni 400 dan 400 pikselgacha o'lchamlari bilan ochdik.

win.getMouse () buyrug'i yutish oynasi maydonida sichqonchanning har qanday tugmachasi bosilishini kutadi.

win.close () win grafikasi uchun oynani yopadi.

Python dasturlaridagi graphics.py moduli yordamida nuqta, chiziq, doira, to'rtburchak, ellips va ko'pburchakni aks ettirishingiz va matnni ekranda ko'rsatishingiz mumkin.

Ob'ektini Python grafik oynasida joylashtirish uchun uning koordinatalarini Python koordinatalar tizimida belgilash kerak.

Berilgan grafik oynada grafik chizishdan oldin, ularga aniqlik kiritishingiz kerak.

Python-ga nuqta qo'yish uchun Point (x, y) funksiyasidan foydalaning

obj = nuqta (x, y)

x, y - nuqtaning koordinatalari.

Grafik oynasida nuqta o'rnatadigan va ko'rsatadigan Python dasturi misoli.

Grafik importdan * # grafik kutubxonani import qilish

```
win = GraphWin ("Grafika uchun oyna", 400, 400) # 400 x 400 piksel o'lchamdagi grafikalar uchun oyna yarating
```

```
obj = Point (50, 50) # (50, 50) koordinatalarida nuqta yarating
```

```
obj.draw (win) # grafika uchun oynadagi nuqtani ko'rsatadi
```

```
win.getMouse () # sichqoncha tugmasi bosilishini kuting
```

```
win.close () # grafika uchun oynani yoping
```

Python-da chiziq segmentini aniqlash uchun Line funksiyasidan foydalaning

Ob'ekt birinchi uchining nuqtasi, ob'ekt ikkinchi uchining nuqtasi)

```
obj = Line (nuqta (x1, y1), nuqta (x2, y2))
```

x1, y1 - chiziq segmentining boshlanish koordinatalari,

x2, y2 - chiziq bo'lagi oxirining koordinatalari.

Berilgan grafik oynada grafika chizishdan oldin ularni aniqlash kerak.

Python-ga nuqta o'rnatish uchun Point (x, y) funksiyasidan foydalaning

```
obj = Point(x, y)
```

x, y – nuqta koordinatalari.

Pythonda dasturga misol. Nuqtani grafik oynada ifodalovchi va akslantiruvchi jarayonga misol.

```
from graphics import * # graphics kutubxonani import qilamiz
```

```
win = GraphWin("grafika uchun oyna", 400, 400) # 400 ga 400 piksel oynali
```

```
grafik uchun oynani yaratamiz obj = Point(50, 50) # (50, 50) nuqtada nuqtani
```

```
yaratamiz
```

```
obj.draw(win) # grafika uchun oynada nuqtani akslantiramiz
```

```
win.getMouse() # sichqon knopkasini bosilishini kutamiz
```

```
win.close() # grafika uchun oynani berkitamiz
```

Pythonda kesmani berish uchun funksiya Line(birinchi uchi nuqtasi, ikkinchi uchi nuqtasi) qo'llaniladi

```
obj = Line(Point(x1, y1), Point(x2, y2))
```

x1, y1 – chiziq kesmasining boshi koordinatasi,

x2, y2 – chiziq kesmasining ohirining koordinatasi.

Pythonda chiziqlar rangini berish uchun quyidagi buyruq qo'llaniladi

```
obj.setOutline("цвет")
```

Пример программы на Pythonда dasturga misol u chiziqni grafik oynada akslantirdi.

```
from graphics import *
```

```
win = GraphWin("grafika uchun oyna", 400, 400)
```

```
obj = Line(Point(50, 50), Point(350, 350))
```

```
obj.setOutline("blue")
```

```
obj.draw(win)
```

```
win.getMouse()
```

```
win.close()
```

Pythonda aylanani akslantirish uchun quyidagi opratoridan foydalaniladi

```
obj = Circle(Point(x, y), R)
```

x – aylana markazi koordinatalari,

R – aylana rdiusi.

Pythonda dasturga misol, u aylananani grafik oynada akslantiradi.

```
from graphics import *
```

```
win = GraphWin("Окно для графики", 400, 400)
```

```
obj = Circle(Point(200, 200), 50)
```

```
obj.draw(win)
```

```
win.getMouse()
```

```
win.close()
```

Pythonda to'rtburchakni akslantirish uchun

rotsedura `obj = Rectangle(Point(x1, y1), Point(x2, y2))` qo'llaniladi

```
obj = Rectangle(Point(x1, y1), Point(x2, y2))
```

x_1, y_1 – to'rtburchakning yuqori chap burchagi koordinatalari,

x_2, y_2 – to'rtburchakning yuqori o'ng burchagi koordinatalari

Pythonda to'rtburchakni grafik oynada akslantiruvchi dasturga misol

```
from graphics import *
```

```
win = GraphWin("Окно для графики", 300, 300)
```

```
obj = Rectangle(Point(50, 50), Point(200, 250))
```

```
obj.draw(win)
```

```
win.getMouse()
```

```
win.close()
```



Pythonda ellipsni akslantirish uchun quyidagi prosdura qo'llaniladi

```
obj = Oval(Point(x1, y1), Point(x2, y2))
```

x1, y1 – ellipsning birinchi fokusi koordinatalari,

x2, y2 – ellipsning ikkinchi koordintalari.

Pythonda grafik oynada ekllipsni aks ettirish uchun.

```
from graphics import *
```

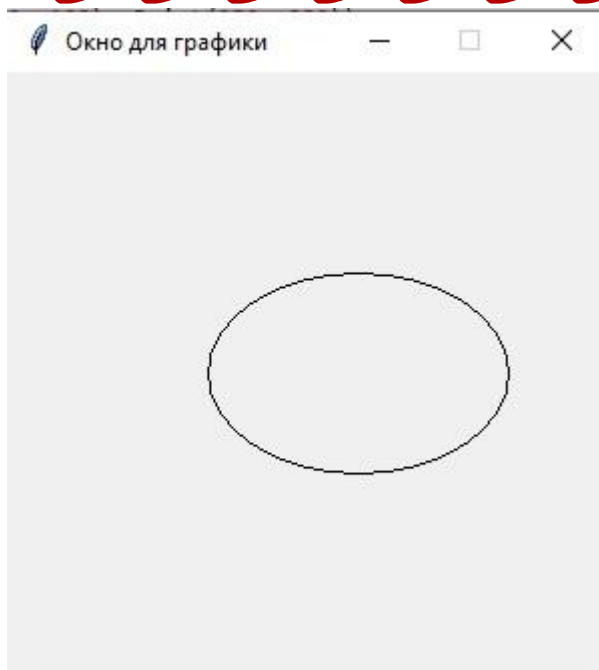
```
win = GraphWin("Окно для графики", 300, 300)
```

```
obj = Oval(Point(100, 100), Point(250, 200))
```

```
obj.draw(win)
```

```
win.getMouse()
```

```
win.close()
```



4.

Pythonda ko'p burchakni akslantirish uchun quyidagi protsedura `obj = Polygon(Point(x1, y1), Point(x2, y2),..., Point(xn, yn))` protsedura qo'llaniladi

$x_1, y_1, x_2, y_2, \dots, x_n, y_n$ – ko'pburchakning uchlari koordinatalari.

Pythonda beshburchakni grafik oynada akslantiruvchi

```
from graphics import *
```

```
win = GraphWin("Окно для графики", 400, 400)
```

```
obj = Polygon(Point(10, 10), Point(300, 50), Point(200, 300), Point(150, 150), Point(70, 70))
```

```
obj.draw(win)
```

```
win.getMouse()
```

```
win.close()
```

Pythonda grafik ob'ektni bo'yashning rangini aniqlash

Pythonda grafik ob'ektni bo'yash quyidagi `obj.setFill("rang")` buyruq qo'llaniladi

Bo'yalgan ko'k aylanani chizuvchi pythondagi dasturga misol.

```
from graphics import *
```

```
win = GraphWin("Окно для графики", 400, 400)
```

```
obj = Circle(Point(200, 200), 50)
```

```
obj.setFill("blue")
```

```
obj.draw(win)
```

```
win.getMouse()
```

```
win.close()
```

Pythonda ob'ektlar chegaralarinin tahrirlash uchun quyidagi

prtseduradan foydalaniladi `setOutline("chegara rangi")` va

`setWidth(chegara kengligi)`.

`obj.setOutline("blue")` – obj ob'ekt chegaraga bilan ko'k rangda akslanadi.

`obj.setWidth(5)` – объект obj ob'ekt 5 piksel chegaraga kengligi bilan akslanadi.

avtomatik tarzda По умолчанию графический объект в Pythonda ob'ekt qora rangli chegaralar bilan 1 piksel kengligida akslanadi. **Pythonda grafik oynada**

ko'k rangli chegarali figurani aslantirishga misol.

```
from graphics import *
```

```
win = GraphWin("grafika uchun oyna", 310, 310)
```

```
obj = Polygon(Point(10, 10), Point(300, 50), Point(200, 300), Point(150, 150),  
Point(70, 70))
```

```
obj.setOutline("blue")
```

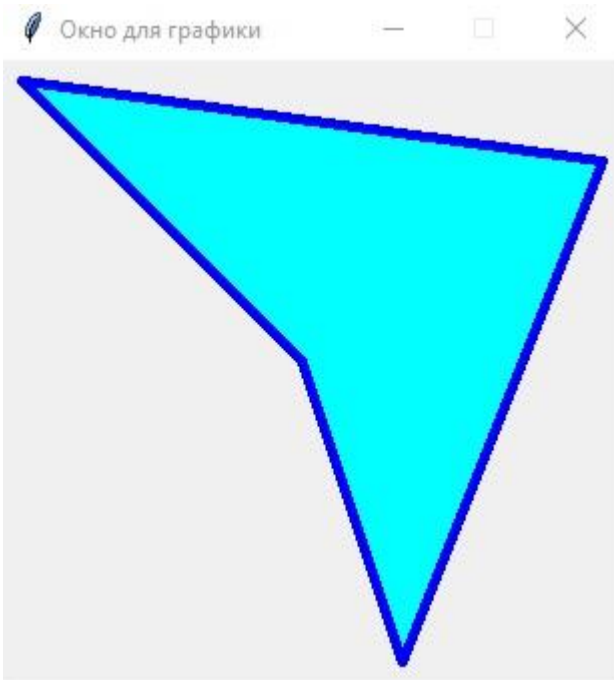
```
obj.setWidth(5)
```

```
obj.setFill("cyan")
```

```
obj.draw(win)
```

```
win.getMouse()
```

```
win.close()
```



Pythonga grafik ob'ektni kuchirsak, quyidagi protsedura

qo'llaniladi: `move(dx, dy)`, u o'z nabatida ob'ektni dx pikselga o'nga va dy pikselga pastga suradi.

`obj.move(50, 50)` obj ob'ektni 50 pikselga o'nga va 50 pikselga pastga suradi.

Ob'ektlarni klonlashtirish uchun quyidagi `clone()` protsedurasi qo'llaniladi

```
newObj = obj.clone()
```

bu buyruq orqali yangi grafik ob'ekt yaratiladi `newObj`, u o'z navbatida `obj` ob'ektiga o'hshash.

ekrandan figuralarni uchirish uchun `undraw()` protsedurasi qo'llaniladi. ob'ekt grafik ynadan uchiriladi lekin hotiradan olib tashlanmaydi.

```
obj.undraw()
```

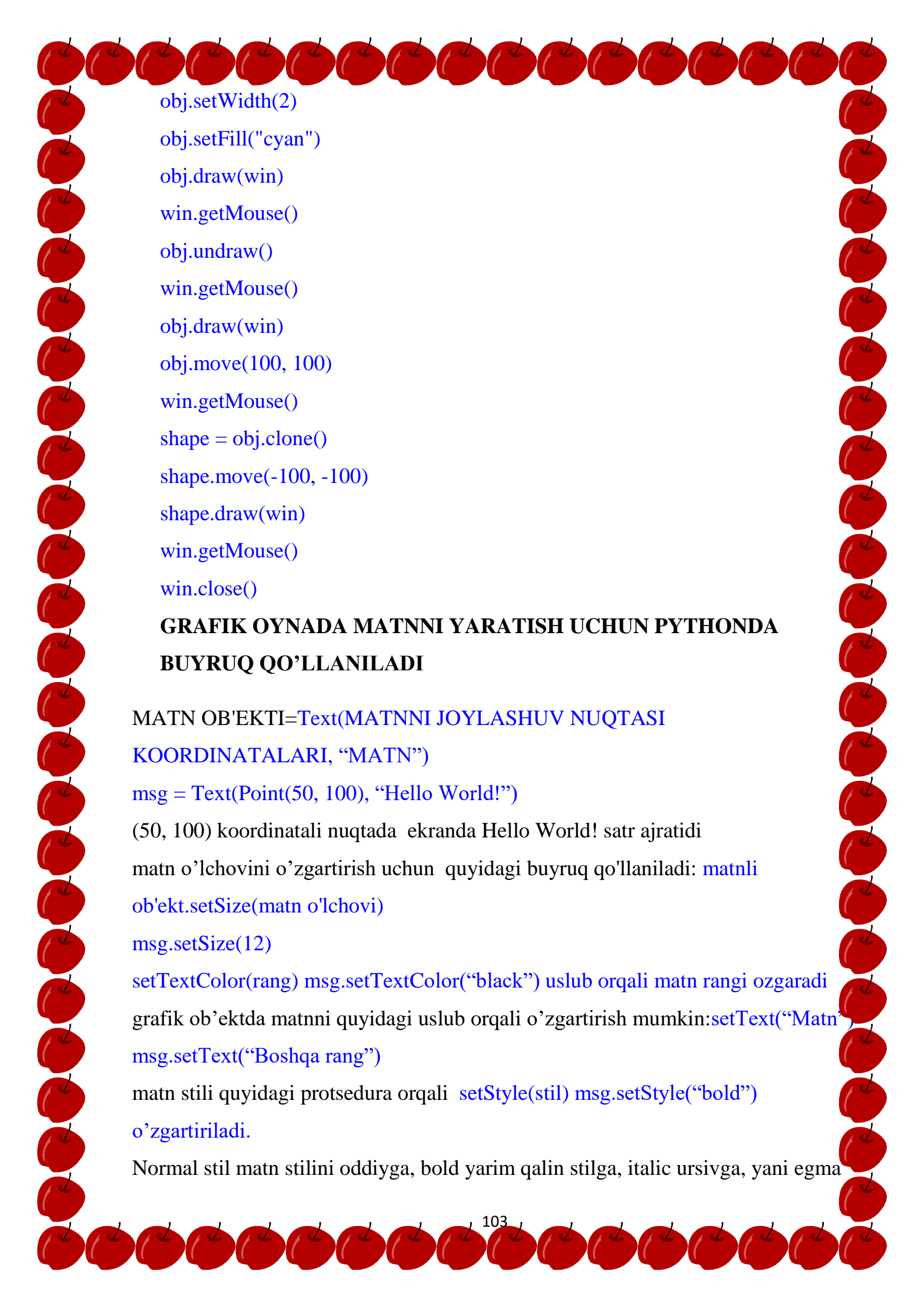
Pythonda grafik oynada ob'ektni uchirish kuchirish va nusxa qilishga misollar.

```
from graphics import *
```

```
win = GraphWin("Окно для графики", 400, 400)
```

```
obj = Polygon(Point(30, 10), Point(30, 50), Point(20, 30), Point(15, 30),  
Point(7, 7))
```

```
obj.setOutline("blue")
```



```
obj.setWidth(2)
obj.setFill("cyan")
obj.draw(win)
win.getMouse()
obj.undraw()
win.getMouse()
obj.draw(win)
obj.move(100, 100)
win.getMouse()
shape = obj.clone()
shape.move(-100, -100)
shape.draw(win)
win.getMouse()
win.close()
```

GRAFIK OYNADA MATNNI YARATISH UCHUN PYTHONDA BUYRUQ QO'LLANILADI

MATN OB'EKTI=`Text(MATNNI JOYLASHUV NUQTASI
KOORDINATALARI, "MATN")`

```
msg = Text(Point(50, 100), "Hello World!")
```

(50, 100) koordinatali nuqtada ekranda Hello World! satr ajratidi

matn o'lchovini o'zgartirish uchun quyidagi buyruq qo'llaniladi: `matnli`

```
ob'ekt.setSize(matn o'lchovi)
```

```
msg.setSize(12)
```

`setTextColor(rang)` `msg.setTextColor("black")` uslub orqali matn rangi ozgaradi
grafik ob'ektda matnni quyidagi uslub orqali o'zgartirish mumkin: `setText("Matn")`

```
msg.setText("Boshqa rang")
```

matn stili quyidagi protsedura orqali `setStyle(stil)` `msg.setStyle("bold")`
o'zgartiriladi.

Normal stil matn stilini oddiyga, bold yarim qalin stilga, italic ursivga, yani egma

shriftga o'zgartiradi. Bold italic esa matn stilini yarim qalinlashgan kursivga

o'zgartiradi.

Pythonda matnni grafik oynada ko'rsatuvchi dasturga misol.

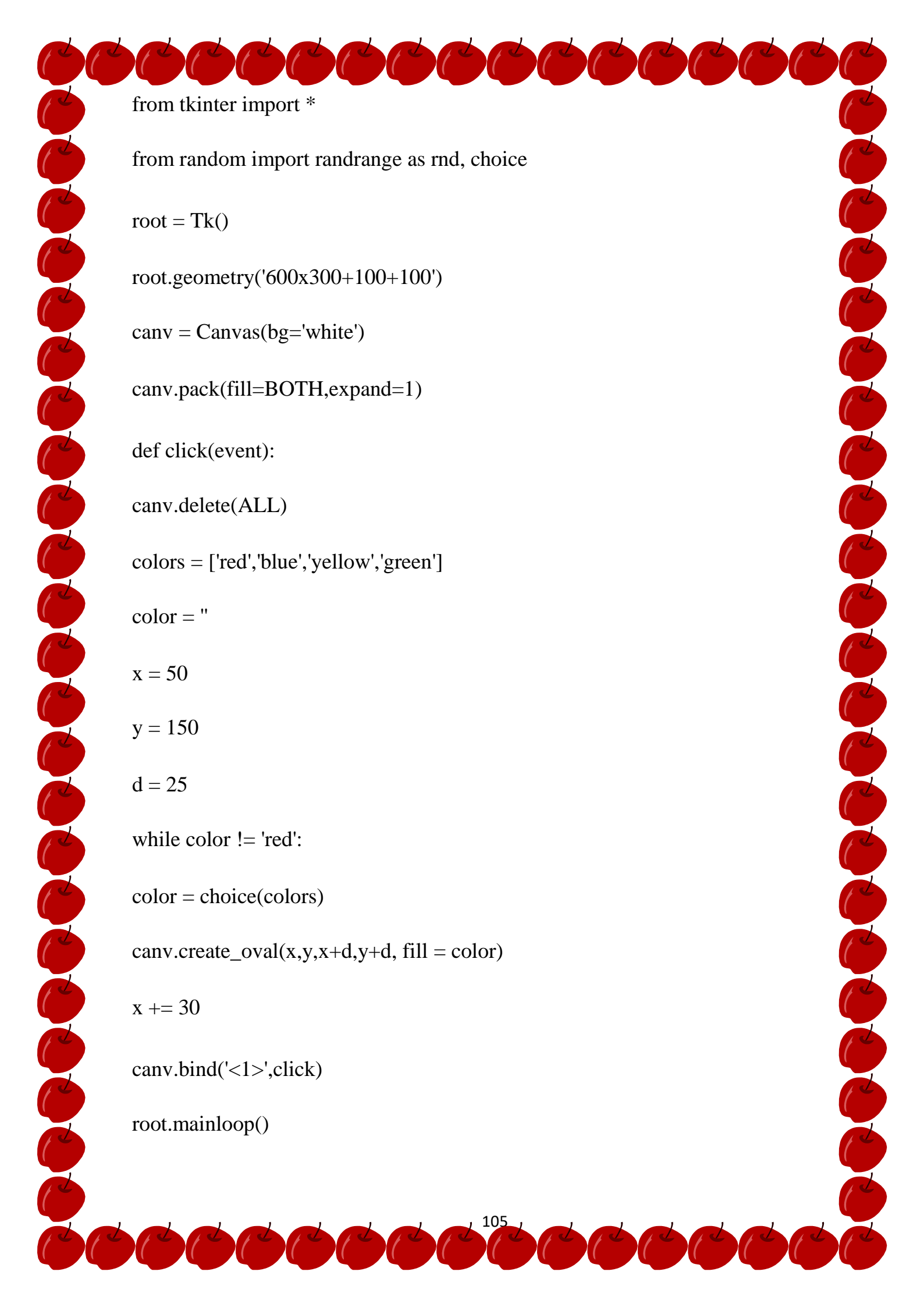
```
from graphics import *  
win = GraphWin("Окно для графики", 400, 400)  
obj = Polygon(Point(10, 10), Point(300, 50), Point(200, 300), Point(150, 150),  
Point(70, 70))  
obj.setOutline("blue")  
obj.setWidth(5)  
obj.setFill("cyan")  
obj.draw(win)  
win.getMouse()  
obj.undraw()  
msg = Text(Point(200, 200), "EKRANDAN FIGURA UCHIRILDI")  
msg.setSize(12)  
msg.setTextColor("black")  
msg.setStyle("bold italic")  
msg.draw(win)  
win.getMouse()  
win.close()
```

Dasturlshga doir misollar.

Amalga oshguncha bajarish. While tsikli

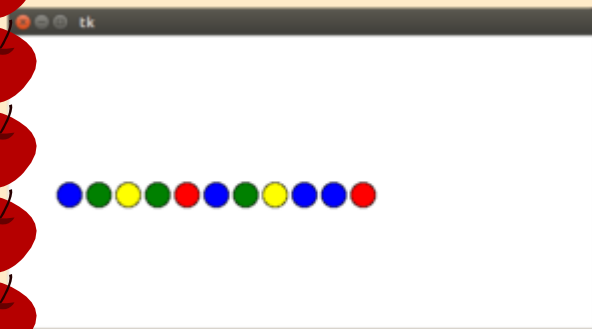
Dizil chizilguncha doira, rasm chizish:





```
from tkinter import *  
  
from random import randrange as rnd, choice  
  
root = Tk()  
  
root.geometry('600x300+100+100')  
  
canv = Canvas(bg='white')  
  
canv.pack(fill=BOTH,expand=1)  
  
def click(event):  
  
    canv.delete(ALL)  
  
    colors = ['red','blue','yellow','green']  
  
    color = "  
  
    x = 50  
  
    y = 150  
  
    d = 25  
  
    while color != 'red':  
  
        color = choice(colors)  
  
        canv.create_oval(x,y,x+d,y+d, fill = color)  
  
        x += 30  
  
    canv.bind('<1>',click)  
  
    root.mainloop()
```

ikkitaqizichiziguncachizish



Qizilvako'kchizilguncharasmchizish

β. Chertgacho'shayerdaaylanachizishkerak, toki 3

shartaixtiyoriydevrgaurilmagunchatsodifiyyo'nalishdaharakatqiluvchidoiranichizis

Создать игру «sharchani tutib oling» o'yininin yaratish, 3ta sharcha tutilganda
gaish va umumiy sonda 10 ta sharcha paydo bo'lguncha tugatish.

"SHILLIQ QURT" MSALASI YECHIM

SHART

H metrli vertical ustun stidan shiiq qut sudraib yurmoqda, u bir kunda a metrga ko'tariladi, tunda esa b metrga pasayadi. Nechanchi kunga shilliq qurt stunning eng yuqori chuqqisiga sudralib yetadi?

Dastur kiritish uchun quyidai **h, a, b** sonlarni qo'llaydi.

Dastur bitta natural sonni chiqarishi kera bo'ladi. **a>b** ligi **garantiylashtiriladi**.

Yechish

```
h =int(input())
```

```
a =int(input())
```

```
b =int(input())
```

```
print(int((h -a -1) //(a -b) +2))
```

Yoki Dasturni bunday variantini taklif etamiz:

```
h = float(input())
a = float(input())
b = float(input())
print(int(1 + (h - b - 1) / (a - b)))
```

Shilliq qurtning rasmini chizuchi dastuini pascaldan python tiliga o'tkazing, uni Pythonda chizish dasturini loyiha ustida ishah sifatida amalga oshiring:

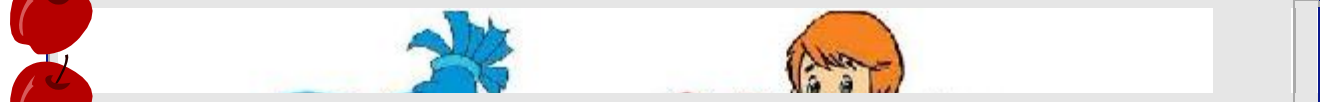
Yechilishi

Pascal

```
1 uses graphABC;
2 var x,y:integer;
3 begin
4   setwindowsize(400,400);
5   centerwindow;
6   x:=windowwidth div 2-40;
7   y:=x+40;
8   setpencolor(clBlack);
9   setpenwidth(2);
10  setbrushcolor(clLtGray);
11  ellipse(x-55,y-45,x+55,y+43);
12  setpencolor(clWhite);
13  setbrushcolor(clBlack);
14  circle(x,y+8,30);
15  setpencolor(clBlack);
16  line(x-50,y+42,x+64,y+42);
17  line(x-55,y+48,x+60,y+48);
```



```
18 line(x-50,y+42,x-55,y+48);
19 line(x+50,y+42,x+85,y-40);
20 line(x+60,y+48,x+92,y-30);
21 floodfill(x,y+45,clLtGray);
22 setbrushcolor(clLtGray);
23 circle(x+105,y-45,20);
24 floodfill(x+62,y+38,clLtGray);
25 setpenwidth(3);
26 line(x+95,y-64,x+85,y-80);
27 line(x+115,y-64,x+125,y-85);
28 setpencolor(clWhite);
29 setbrushcolor(clWhite);
30 circle(x+98,y-45,4);
31 circle(x+112,y-45,4);
32 setpencolor(clBlack);
33 setbrushcolor(clBlack);
34 circle(x+98,y-45,3);
35 circle(x+112,y-45,3);
36 arc(x+105,y-40,10,220,320);
37 end.
```



IY.PYTHONDA TAKRORLANISH.

Dasturingizni takrorlanuvchi operator for yordamida tuzing. Python dunyosida ikki xil tsikl mavjud:

- for uchun
- while tsikl

For tsikli ikkinchisiga qaraganda ancha mashhurligini payqadim. Biz biron bir narsani qayta-qayta bajarishimiz kerak bo'lganda, ko'chadan foydalaniladi.

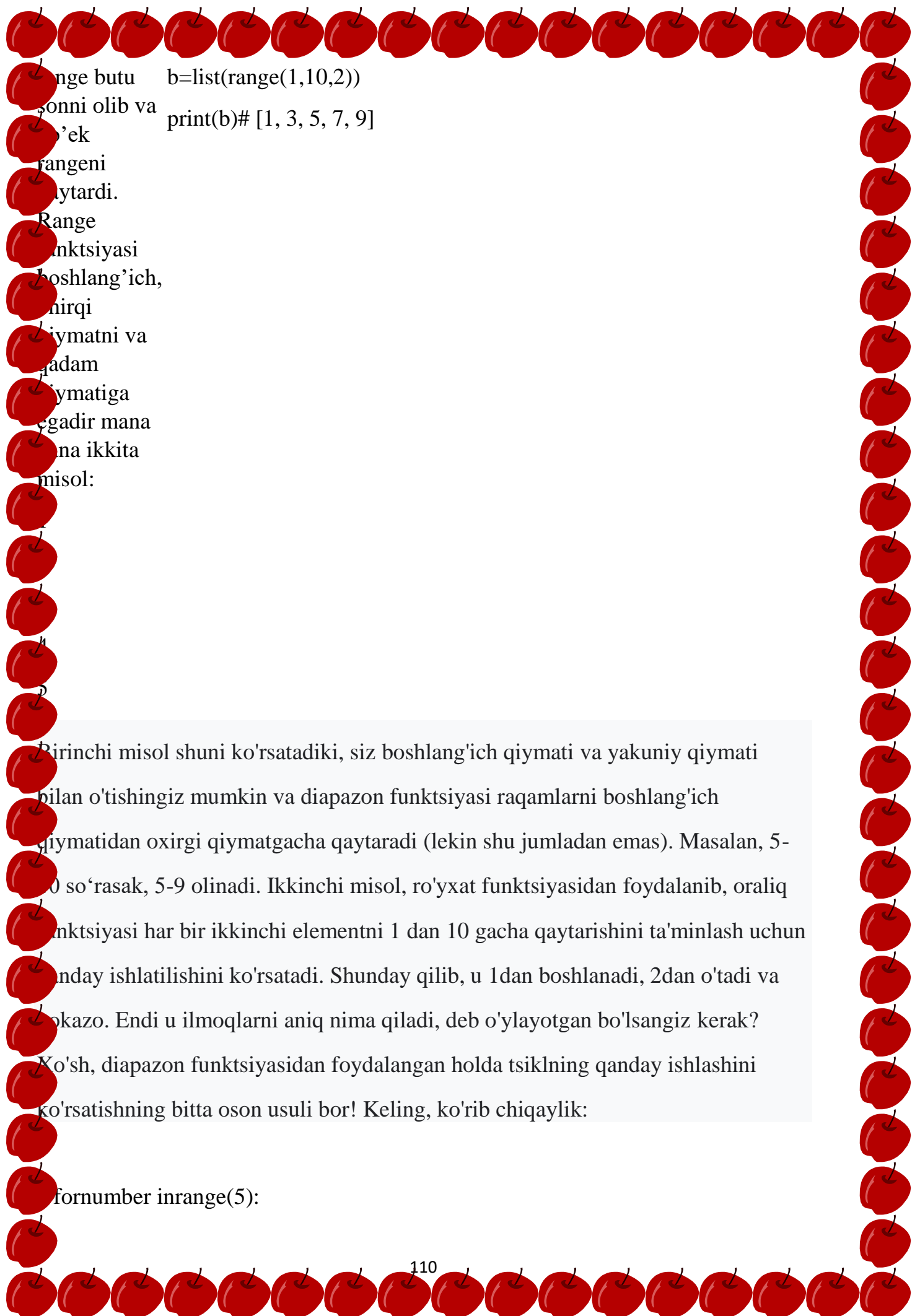
Ko'pincha, biron bir ma'lumot ustida bir necha marta (yoki bir qator operatsiyalarni) bajarishingiz kerak bo'ladi. Bu erda tsikllar o'ynaydi. Ularning yordami bilan ushbu masalani iloji boricha soddalashtirish mumkin bo'ladi. Keling ushbu tuzilmalar qanday ishlashini batafsil ko'rib chiqamiz!

Tsikl for.

Avval aytib o'tganimizdek, biron bir narsani n marta takrorlash kerak bo'lganda biz loopdan foydalanamiz. Agar siz misolga qarasangiz, buni tushunish osonroq. Biz Python-ning o'rnatilgan diapazonidan foydalanmoqdamiz. Intervalli funktsiya "n" uzun elementlarning ro'yxatini yaratadi. Python 2.X versiyasi xrange deb nomlangan yana bir funktsiyaga ega, bu raqamlarni ishlab chiqaruvchi va diapazon kabi resurslarni talab qilmaydi. Ilgari, ishlab chiquvchilar xrange oralig'ini Python 3-ga o'zgartirdilar. Mana bir misol:

```
1 print(range(5))# oTBeT: range(0, 5)
```

```
Korinib a=range(5,10)
turibdiki, print(a)# range(5, 10)
range
funktsiyasi
```



```
b=range(1,10,2)  
print(b)#[1, 3, 5, 7, 9]
```

Range
funksiyasi

boshlang'ich,
oxirgi

qiymatni va
qadam
qiymatiga
tegadir mana
na ikkita
misol:

Birinchi misol shuni ko'rsatadiki, siz boshlang'ich qiymati va yakuniy qiymati bilan o'tishingiz mumkin va diapazon funksiyasi raqamlarni boshlang'ich qiymatidan oxirgi qiymatgacha qaytaradi (lekin shu jumladan emas). Masalan, 5-10 so'rasak, 5-9 olinadi. Ikkinchi misol, ro'yxat funksiyasidan foydalanib, oraliq funksiyasi har bir ikkinchi elementni 1 dan 10 gacha qaytarishini ta'minlash uchun shunday ishlatilishini ko'rsatadi. Shunday qilib, u 1dan boshlanadi, 2dan o'tadi va shunday davom etadi. Endi u ilmoqlarni aniq nima qiladi, deb o'ylayotgan bo'lsangiz kerak? Xo'sh, diapazon funksiyasidan foydalangan holda tsiklning qanday ishlashini ko'rsatishning bitta oson usuli bor! Keling, ko'rib chiqaylik:

```
for number in range(5):
```

```
2 print(number)
```

Bu erda nima bo'ldi? Buni tushunish uchun chapdan o'ngga o'qiyamiz. 5 oralig'ida har bir raqam uchun biz raqamni kiritamiz. Bilamizki, 5 qiymatli diapazonga qo'ng'iroq qilsak, 5 ta element ro'yxati olinadi. Shunday qilib, har safar ko'chadan o'tib, har bir elementni bosib chiqaradi. Yuqorida ko'rsatilgan for loopni quyidagilarga teng bo'lishi mumkin:

```
1 for number in [0,1,2,3,4]:
```

```
2     print(number)
```

Diapazon funksiyasi natijani faqat biroz kichikroq qiladi. For loop har qanday Python iteratoridan o'tishi mumkin. Ro'yxat bilan qanday ishlashini biz allaqachon aniq bilib oldik. Keling, lug'at bilan takrorlash mumkinligini ko'rib chiqaylik

```
1 a_dict={"one":1,"two":2,"three":3}
```

```
2
```

```
3 for key in a_dict:
```

```
4     print(key)
```

Lug'atdan foydalanganda, u avtomatik ravishda tugmachalar atrofida takrorlanishini ko'rasiz. `A_dict.keys()` da for kalitini ko'rsatishingiz shart emas (ammo bu ham ishlaydi). Python faqat kerakli narsalarni qiladi. Siz nima uchun tugmachalar lug'atda ko'rsatilganidan boshqacha tartibda ko'rsatilishini qiziqtirgan bo'lishingiz mumkin? Tegishli maqoladan ma'lumki, lug'atlarga buyurtma berilmagan, shuning uchun biz ularni takrorlashimiz mumkin va kalitlar har qanday tartibda bo'lishi mumkin. Endi, kalitlarni saralash mumkinligini bilib, ularni takrorlashdan oldin saralashingiz mumkin. Keling, bu qanday ishlashini ko'rish uchun so'z boyligini biroz o'zgartirib ko'raylik.



```
a_dict={1:"one",2:"two",3:"three"}
```

```
keys=a_dict.keys()
```

```
keys=sorted(keys)
```

```
forkey inkeys:
```

```
    print(key)
```

Keling, to'xtab, ushbu kod nimani anglatishini aniqlaymiz. Birinchidan, biz lug'at yaratdik, unda klavishlar qatorlar o'rniga butun son sifatida ishlaydi. Keyinchalik, biz lug'atdan kalitlarni chiqarib tashladik. Keys () usulini har safar chaqirganingizda, u tartibsiz tugmachalar ro'yxatini qaytaradi. Agar siz ularni bosib chiqarsangiz va ular ortib boruvchi tartibda ekanliklarini ko'rsangiz, demak bu shunchaki tasodif. Endi biz kalit so'zlari deb nomlanadigan o'zgaruvchida saqlanadigan lug'at tugmachalariga kirish huquqiga egamiz. Biz o'z ro'yxatimizni saralaymiz va keyin for for loopidan foydalanamiz. Biz endi narsalarni biroz qiziqroq qilishga tayyormiz. Biz intervalli funktsiyasini ko'rib chiqishga harakat qilamiz, ammo biz faqat butun sonlarni chop etishimiz kerak. Buning uchun biz qadam parametrlari oralig'i o'rniga shartli bayonotdan foydalanishimiz kerak. Buni quyidagicha bajarish mumkin:

```
python
```

```
for number in range(10):
```

```
    if number % 2 == 0:
```

```
        print(number)
```

результат:

```
python
```

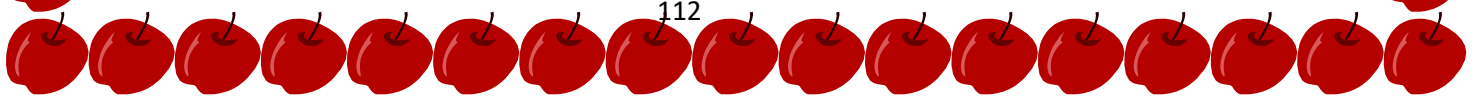
```
0
```

```
2
```

```
4
```

```
6
```

```
8
```



2 2

3 4

4 6

5

8 For uchun

For tsikli, parametr tsikli deb ham ataladi, Python-da funktsiyalarga boy. For tsikli o'zgaruvchini va o'zgaruvchiga o'tadigan qiymatlar to'plamini belgilaydi.

Qiymatlar to'plami ro'yxat, katakcha, satr yoki diapazon sifatida ko'rsatilishi mumkin.

Bu erda pastadirni ishlatishning eng oddiy misoli keltirilgan, bu yerda kortej qiymatlar to'plami sifatida ishlatiladi:

:

```
i = 1
for color in 'red', 'orange', 'yellow', 'green', 'cyan', 'blue', 'violet':
    print('#', i, ' color of rainbow is ', color, sep = " ")
    i += 1
```

Ushbu misolda o'zgaruvchan rang ketma-ket ravishda "qizil", "to'q sariq" va boshqalarni oladi. Tsiklning tanasida rang nomi, ya'ni rang o'zgaruvchisi qiymati, shuningdek tsiklning takrorlanish raqami, avval 1 ga teng bo'lgan raqam, keyin esa xabar ko'rsatiladi bittaga ko'paytirildi (ko'chadan har bir o'tish bilan $i += 1$ ko'rsatmasi bilan).

$i += 1$, $i = i + 1$ ga teng (bu shunchaki stenografiya). Ushbu stenografiyadan barcha arifmetik amallar uchun foydalanish mumkin: $* =$, $- =$, $/ =$, $\% =$...

Qadriyatlar ro'yxatida har xil turdagi iboralar bo'lishi mumkin, masalan

:

```
for i in 1, 2, 3, 'one', 'two', 'three':
    print(i)
```

tsiklning dastlabki uchta takrorlanishi davomida i o'zgaruvchisi int tipidagi qiymatga ega bo'ladi; keyingi uchta takrorlash davomida u str tipida bo'ladi.

Diapazon funktsiyasi range

Datda, tsikllar uchun harakatlar ketma-ketligini belgilangan miqdordagi takrorlash yoki tsikldagi o'zgaruvchining qiymatini ba'zi bir boshlang'ich qiymatdan yakuniy qiymatga o'zgartirish uchun foydalaniladi.

orni berilgan sonda n marta takrorlash uchun for funksiyasidan range bilan foydalanish mumkin:

4

```
for i in range(4): # for i in 0, 1, 2, 3 for instruktsiyasi bilan teng kuchli:
    # bu yrd tsiklik amallarni bajarish mumkin
    print(i)
    print(i ** 2)
tsikl tugadi, chunki otstuplioldidan qoldirish bilan 0 blok tugadi
print('tsikl ohiri')
```

N raqamli doimiy, o'zgaruvchan yoki o'zboshimchalik bilan arifmetik ifoda bo'lishi mumkin (masalan, $2 ** 10$). Agar n qiymati nolga yoki manfiy bo'lsa, u holda tsikl anasi bir marta ham bajarilmaydi.

Diapazon funktsiyasi ham bitta emas, balki ikkita parametрни qabul qilishi

mumkin. (A, b) diapazoniga qo'ng'iroq qilish indeks o'zgaruvchisi a dan b - 1

nacha qiymatlarni qabul qilishini anglatadi, ya'ni ikkita parametr bilan chaqirilgan

Diapazon funktsiyasining birinchi parametri indeks o'zgaruvchisining boshlang'ich

qiymatini o'rnatadi, ikkinchisi parametr birinchi qiymatni belgilaydi, ya'ni indeks

o'zgaruvchisi qabul qilmaydi. Agar $a \geq b$ bo'lsa, u holda tsikl bir marta ham

bajarilmaydi. Masalan, 1 dan n gacha bo'lgan sonlarning qiymatlarini yig'ish uchun

uz quyidagi dasturdan foydalanishingiz mumkin:

6

```
m = 0
```

```
n = 5
for i in range(1, n + 1):
    sum += i
print(sum)
```

Ushbu misolda i o'zgaruvchisi 1, 2, ..., n qiymatlarini qabul qiladi va o'zgaruvchining yig'indisi qiymati belgilangan qiymatlar bo'yicha ketma-ket oshiriladi.

Va nihoyat, indeks o'zgaruvchisi kamayadigan tsiklni tashkil qilish uchun uch parametrli range diapazon funksiyasidan foydalanish kerak.

Birinchi parametr indeks o'zgaruvchisining boshlang'ich qiymatini belgilaydi, ikkinchi parametr indeks o'zgaruvchisi o'zgaradigan qiymat (uni hisobga olmagan!), Uchinchi parametr esa indeks o'zgaruvchisining o'zgarish miqdori.

Masalan, (1, 100, 2) funksiyasi yordamida 1 dan 99 gacha bo'lgan barcha g'alati raqamlar bo'ylab va (100, 0, -1) diapazon yordamida 100 dan 1 gacha bo'lgan barcha raqamlar bo'ylab tsikl qilishingiz mumkin. Keyinchalik rasmiy ravishda, $d > 0$ oralig'idagi (a, b, d) tsikl uchun $i = a$, $i = a + d$, $i = a + 2 * d$ va shunga o'xshash indeks o'zgaruvchilarining qiymatlari o'rnatiladi $i < b$ bo'lgan barcha qiymatlar.

Agar $d < 0$ bo'lsa, u holda tsikl o'zgaruvchisi barcha $i > b$ qiymatlarini qabul qiladi.

3. print () funksiyasini sozlash

Odatiy bo'lib, print () funksiyasi bir nechta argumentlarni oladi, ularni bo'shliq bilan ajratib chiqaradi va keyin chiziqli lentani qo'shadi. Ushbu xatti-harakatni sep (separator) va end (end) nomlangan parametrlari yordamida o'zgartirish mumkin.

qadam-baqadam yugurish

```
print(1, 2, 3)
print(4, 5, 6)
print(1, 2, 3, sep=', ', end='. ')
print(4, 5, 6, sep=', ', end='. ')
print()
print(1, 2, 3, sep="", end=' -- ')
```

```
print(4, 5, 6, sep=' * ', end='.')
```

Bu erda nima bo'layotganini qiziqiryapsizmi? Yana foiz belgisi nima? Pythonda% modul operatori deb nomlanadi. Modul operatoridan foydalanganda, u qoldiqni ayrtaradi. Agar butun sonni ikkiga ajratganda, raqamni qoldiqsiz olasiz, shuning uchun biz bu raqamlarni chop etamiz. Kelajakda siz modul operatoridan tez-tez foydalanishni xohlamasligingiz mumkin, ammo mening ishimda bu ko'pincha yordam beradi. Endi biz while loopini ko'rib chiqamiz.

1. O'yin shaklida while bilan ishlash.

Informatika darsida ishbilarmonlik o'yini: "Tarmoqlanish, tsikl operatorlarini o'qitish. Massiv" m

Dars shakli: Ishbilarmonlik o'yini. (2 soat)

Dars turi: o'rganilgan materialni konsolidatsiya qilish.

Askunalar: kompyuter, kartalar, savollar, pul "aylanma. birliklari y.e.",

millimetrluk qog'oz, marker.

Darsning maqsadi:

Biling: Tarmoqlanish operatori, tsikl, massiv.

Bunga qodir: masalalarni Yechishda tarmoqlanish, tsikl va massiv

operatorlaridan foydalanish, tsikl operatorlaridan foydalanish: masalalarni

Yechishda while, takrorlash, uchun, massiv.

Dars borishi

1. Dars boshida sinf 5-6 kishidan iborat guruhlariga bo'linadi, o'qituvchi

kompaniyalar tashkil qilishni taklif qiladi. Har bir firmaning o'z dasturchisi,

hukumatliyachisi, menejeri va prezidenti va firma nomi bor. Yigitlar kompaniyaning

har bir aributlarini o'zlarini o'ylab topishadi.

O'qituvchi o'quvchilarni dars mavzusiga e'tibor berishga taklif qiladi va savol

beradi: Biz ushbu mavzu haqida nimalarni bilamiz? Biz nima qila olamiz?

Talabalardan bugungi darsning maqsadini shakllantirishni so'raydi. Darsning maqsadi belgilanadi, ya'ni. ishbilarmonlik o'yinining maqsadi. O'qituvchining so'zlariga ko'ra, bizning o'yinimizni boshlash uchun talabalar blits savollarga javob berish orqali dastlabki kapitalni to'plashlari va tez va aniq javob berishlari kerak.

III. Har bir firma uchun 10 ta savol beriladi, har bir javobning narxi 100 dollarni tashkil qiladi. javob qabul qilinadi yoki qabul qilinmaydi (eng yaxshi talabalar orasida "Mutaxassislar" kuzatiladi). Keling, o'yinni boshlaymiz:

1 firma uchun savollar:

1. Tarmoqlash buyrug'i to'liq shaklda qanday bajariladi? (Agar u holda. Keyin operator1 boshqa operator2 bo'lsa)
2. Tarmoqlanuvchi buyruqni loyihalash qoidalari qanday? (oldin vergul qo'yilmaydi)
3. Qanday algoritmlar tsiklik deyiladi? (Takroriy harakatlar ketma-ketligi)
4. Tsikl operatorlaridan qaysi biri eng ko'p qirrali hisoblanadi? (while-toki)
5. Repeat siklini boshqarish uchun qaysi funktsiyalardan foydalanish qulay? (succ yoki pred, ins, dec)
6. FOR tsikl operatori qachon ishlatiladi? (Xuddi shu harakatlarni qayta-qayta takrorlang)
7. Boshlanish va tugatish qiymatlari qaysi turga tegishli? (Konstantalar, o'zgaruvchilar, ifodalar bo'lishi mumkin va bir xil turdagi bo'lishi kerak)
8. Massiv nima? (Indeksalar bo'yicha buyurtma qilingan bitta turdagi cheklangan ma'lumotlar to'plami yoki bitta turdagi cheklangan miqdordagi ma'lumotlar to'plami)
9. Dasturda ishlatiladigan massivni qaerda tasvirlash kerak? (Yoki o'zgaruvchilarni e'lon qilish bo'limida yoki turlarni e'lon qilish bo'limida)
10. Indeksning qaysi turi bo'lishi kerak? (Tavsifda ko'rsatilgan turdagi)

Firma uchun savollar:

1. Tugallanmagan tarmoqlanish buyrug'i qanday bajariladi? (if.....else operator)

2. Tarmoqlanish buyrug'ida qanday mantiqiy operatsiyalar qo'llaniladi? (vand-va, or-yoki, yo'q-emas-not)

3. while tsikli qanday formatlanadi? (do tsiklini bajarish sharti bilan)

4. while tsikli qachon tugaydi? (Shart to'g'ri bo'lgan taqdirda takrorlanadi)

5. while bir nechta bayonotlarning bajarilishini qanday tashkil qilish kerak?

{.....})

Firma uchun savollar:

1. Tanlash buyrug'i qanday yoziladi va bajariladi? (variant1: bayonot1;

variantN: bayonotN; tugatish;)

2. Shu so'z uchun while va takrorlash so'zlaridan keyin yozilgan shartlar o'rtasidagi

farq nima? (While so'zidan keyin tsikl ko'rsatmalarining bajarilishini davom

eltirish sharti yoziladi, bu farq)

3. Hisoblagich kamayib boruvchi tsiklni qanday tashkil qilish kerak? (hisoblagich

uchun: = boshlang'ich qiymati pastga tushadigan qiymatgacha)

4. Boshlanish va tugatish qiymatlari qaysi turga tegishli? (Konstantalar,

o'zgaruvchilar, ifodalar bo'lishi mumkin va bir xil turdagi bo'lishi kerak)

5. Bir nechta bayonotlarni for tsiklda bajarilishi uchun qanday yozish kerak? ({ ...

return 0)

6. Bir o'lchovli massiv qanday tavsiflanadi? (var array name: array

lowerBoundary..UpperBoundary] element turi)

7. Ikki o'lchovli massiv? ...

8. Indeks sifatida nimadan foydalanish mumkin? (Ifoda, doimiy, o'zgaruvchan)

9. Indeks turi aslida nimani bildiradi? (Qatorlar, ustunlar soni)

10. Indeks turi qaysi turga kirishi mumkin? (Har qanday oddiy turi)

So'rov blitsi natijalari umumlashtirildi. "Mutaxassislar" ishlab topilgan pulni e'lon qiladi, har bir firmaning moliyachilari odatdagi bo'linmalarni olib, jurnalga yozadilar.

Bankning shaxsiy hisobvarag'idagi mablag'larni hisobga olish jurnali (kompaniya moliyachisi uchun)

kirim	Chiqim	Qoldiq
		Jami:

O'qituvchi ishbilarmon o'yin qoidalarini e'lon qildi (qoidalar jadvallarida bosilgan).

Ishbilarmonlik o'yinlari qoidalari:

1. Dasturiy mahsulotingizni varaqqa yozing: dasturni yozing, ko'rsatmalar bilan ta'minlang.

Firmaning vazifalari: Dasturning mohiyatini tushuntirib bermaslik.

2. O'qituvchi ham dasturning buyurtmachisi, ham buyurtmachisidir. Har bir topshiriqqa ma'lumot vaqtini taqdim etish, shundan tashqari barcha firmalar topshiriqlarini bajarish bo'yicha ishlashni amalga oshirdilar.



1. Kompyuter yozilgan dasturlarni disk raskadrovka qilish uchun ishlash.

2. Muhokama paytida dasturlarni tuzish - ishlash muddatgacha kompyuter,

3. birgalikda. vaqt tugashidan oldin, ishlatilmaydi!

4. Vaqt tugashi bilan (buyurtma muddati) o'qituvchi (buyurtmachi) dasturiy

5. mahsulotlarni ko'rib chiqish tanlovini e'lon qildi.

6. Kompaniya o'z mahsulotlarini "sotuvchisi" dir va o'z dasturini taklif qiladi, ya'ni.

7. Echimni doskada tushuntiradi, shu bilan birga kompaniyaning "dasturchisi" uni

8. kompyuterga yozadi, natijani aks ettiradi.

9. Xost-firma dasturni sinchkovlik bilan tekshiradi, mohiyati bo'yicha savollar

10. beradi, dasturning ayrim qismlariga oydinlik kiritishni so'raydi.

11. Dasturni amalga oshirmagan har bir kompaniya tayyor dasturiy mahsulotni sotib

12. olishga majburdir.

13. Dasturiy mahsulotning shartli qiymati 100 ta an'anaviy birlikni tashkil etadi,

14. ammo quyidagi mezonlarni hisobga olsak, narx oshishi mumkin: original echim,

15. tsional echim.

16. Shartli pullar shartli bankda to'planadi. (Dastlabki kapital blits-turnirda

17. to'plangan birliklardan iborat).

18. Agar barcha firmalar topshiriqni bajargan bo'lsa. Dasturning mohiyatini

19. tushuntirish o'qituvchiga bog'liq.

20. Vazifalar berilgan (darsning Ilovasiga qarang), har bir topshiriq uchun ma'lum

21. vaqt beriladi, vaqt tugagandan so'ng ish ishbilarmonlik o'yin qoidasiga binoan

22. qovom etadi.

23. Ishbilarmonlik o'yinining natijasi: shartli pul shartli idishda hisoblanadi.

24. "Mutaxassislar" firmalar tomonidan har bir muammo echimidan so'ng o'qituvchi

25. bilan birgalikda tinglashadi, natijani kompyuterda ko'rishadi, boshqa firmalarni

26. tinglashadi, muammolarni baholaydilar va har bir muammoni umumlashtir

Vii. O'yinning yakuniy natijasi: shartli pullar hisoblanadi, eng yaxshi kompaniya e'lon qilinadi; Prezident har bir xodimning bahosini baholash varag'iga kiritadi, ekspertlar har bir prezidentning ishini baholaydilar.

"Mutaxassislar" butun o'yin haqida o'z fikrlarini bildiradilar, eng yaxshi firmalarni belgilaydilar, sharhlar bilan baho beradilar:

A) original eritma;

B) ratsional qaror.

"Mutaxassislar" xulosalar jadvalini to'ldiradilar (darsning Ilovasiga qarang).

VIII. Butun o'yin oxirida o'qituvchi o'quvchilarga so'z beradi, aks ettirish amalga oshiriladi.



0Dasturlashtirishda tsikllar. While tsikli

Tsikllar ham shartli ravishda tuzilgan dasturlashning muhim qismidir. Looplar yordamida siz kod qismlarining bajarilishini takrorlashni tashkil qilishingiz mumkin. Bunga ehtiyoj ko'pincha paydo bo'ladi. Masalan, foydalanuvchi ketma-ket raqamlarni kiritadi va ularning har biri umumiy songa qo'shilishi kerak. Yoki bir qator natural sonlar kvadratlarini va shunga o'xshash masalalarni aks ettirishingiz kerak.



Tsikllar ham shartli ravishda tuzilgan dasturlashning muhim qismidir. Tsikllar ordamida siz kod qismlarining bajarilishini takrorlashni tashkil qilishingiz mumkin. Bunga ehtiyoj ko'pincha paydo bo'ladi. Masalan, foydalanuvchi ketma-ket raqamlarni kiritadi va ularning har biri umumiy songa qo'shilishi kerak. Yoki bir qator natural sonlar kvadratlarini va shunga o'xshash masalalarni aks ettirishingiz kerak.

Tsikllar ham shartli ravishda tuzilgan dasturlashning muhim qismidir. Tsikllar ordamida siz kod qismlarining bajarilishini takrorlashni tashkil qilishingiz mumkin. Halqa paytida "While" inglizchadan "toki" deb tarjima qilingan. Ammo "toki" ma'nosida emas, balki "toki shar bajarilguncha shuni qilamiz" ma'nosida.

Siz universal deb ayta olasiz. U Pythonni o'z ichiga olgan tuzilgan dasturlashni ko'llab-quvvatlaydigan barcha tillarda mavjud. Uning sintaksisini barcha tillar uchun quyidagicha umumlashtirish mumkin:

```
while mantiqiy_ifoda {  
    ifoda 1;  
    ...  
    ifoda n;
```

U xuddi shartli if ifodasiga o'xshaydi. Biroq, tsiklik operatorlarga nisbatan ularning tanalari bir necha bor bajarilishi mumkin. Agar shunday bo'lsa, agar sharlavhadagi mantiqiy ifoda to'g'ri bo'lsa, tana bir marta bajariladi. Shundan so'ng, dastur oqimi asosiy tarmoqqa qaytadi va shartli bayonotning butun konstruksiyasi ostida quyidagi bayonotlarni bajaradi.

Agar to'g'ri bo'lsa, uning tanasi bajarilgandan so'ng, ip halqa boshiga qaytadi va shartni yana tekshiradi. Agar mantiqiy ifoda to'g'ri bo'lsa, tana yana bajariladi. Keyin sharlavhaga qaytamiz va hokazo.

Faqatgina sarlavhadagi mantiqiy ifoda noto'g'ri qiymatini qaytarganda tsikl o'z ishini tugatadi, ya'ni tsiklni bajarish sharti endi bajarilmaydi. Shundan so'ng, ijro oqimi butun tsikl ostidagi ifodalarga o'tadi. Ular "tsikldan chiqish bor" deyishadi.

Tsiklning while blok-sxemasini ko'rib chiqing.

Unda porloq ko'k to'rtburchaklar dasturning asosiy tarmog'ini, rombni - mantiqiy ifodali tsikl sarlavhasini, turkuaz to'rtburchakni - tsiklning tanasini bildiradi.

While tsiklida ikkita istisno mavjud:

:

- Agar tsiklga birinchi kirish paytida mantiqiy ifoda **False**-ni qaytarsa, u holda tsikl tanasi bir marta ham bajarilmaydi. Ushbu holatni normal deb hisoblash mumkin, chunki ma'lum bir sharoitda dastur mantig'ida tsikl tanasining ifodalarini bajarishga hojat yo'q deb taxmin qilish mumkin.
- Agar sarlavhadagi mantiqiy ifoda hech qachon "**False**" ni qaytarmasa, lekin har doim "**True**" ga teng bo'lib qolsa, unda uning tanasi ko'chadan (break) majburiy chiqishni o'z ichiga olmasa yoki dasturdan funktsiyalardan chiqishga chaqirmasa, tsikl hech qachon tugamaydi. Python uchun (), exit () Agar tsikl cheksiz ko'p marta takrorlansa va takrorlansa, u holda dastur ko'chiriladi. Ayni paytda u muzlaydi va o'z-o'zidan tugata olmaydi.
- Istisnolar haqidagi darsdagi misolimizni eslaylik. Foydalanuvchi butun son kiritishi kerak. **Input** () funktsiyasi mag'lubiyatni qaytarganligi sababli, dastur kodi **int** () funktsiyasi yordamida kirishni butun son turiga o'zgartirishi kerak. Ammo, agar raqamli tartibda tartibda bo'lsa, **ValueError** istisno holati tashlanadi, u **except** shohi bian ishanadi, bu holat bundan mustasno. Dastur shu bilan yakunlanadi.
- Boshqacha qilib aytadigan bo'lsak, agar dastur raqam bilan keyingi harakatlarni o'z zimmasiga olgan bo'lsa (masalan, tenglikni tekshirish) va u

uni olmagan bo'lsa, unda dasturning qilishi mumkin bo'lgan yagona narsa - bu o'z ishini muddatidan oldin tugatish.


- Lekin siz foydalanuvchidan raqamni kiritmaguncha uni to'g'ri kiritishni so'rashingiz va so'rashingiz mumkin. Buni amalga oshiruvchi kod quyidagicha ko'rinishi mumkin.

```
n=input("Butun sonni kiriting: ")
while type(n)!=int:
    print("Noto'g'ri kiritildi!")
    n=input("Butun sonni kiriting: ")
if n % 2==0:
    print("MISBAT")
else:
    print("TOQ")
```

Zoh 1. Python dasturlash tilida murakkab ko'rsatmalar sarlavhalari oxiriga ko'p qatnashda qo'yilishini unutmang.

Zoh 2. $(n)!$ = Int ifoda turida n (n) o'zgaruvchisi turi () funktsiyasi yordamida tekshiriladi. Agar u int ga teng bo'lmasa, ya'ni n qiymati butun son bo'lmasa, lekin bu holda satr bo'lsa, u holda ifoda haqiqiy bo'ladi. Agar n turi int bo'lsa, u holda bu mantiqiy ifoda noto'g'ri bo'ladi.

Zoh 2. $(n)!$ = Int ifoda turida n (n) o'zgaruvchisi turi () funktsiyasi yordamida tekshiriladi. Agar u int ga teng bo'lmasa, ya'ni n qiymati butun son bo'lmasa, lekin u holda satr bo'lsa, u holda ifoda haqiqiy bo'ladi. Agar n turi int bo'lsa, u holda bu mantiqiy ifoda noto'g'ri bo'ladi.



Izoh 3. Python'dagi % operatori bo'linmaning qolgan qismini topish uchun ishlatiladi. Shunday qilib, agar raqam juft bo'lsa, u holda qoldiqsiz 2 ga bo'linadi, ya'ni qoldiq nolga teng bo'ladi. Agar raqam g'alati bo'lsa, qolgan qismi biriga teng bo'ladi.

.

.

Keling, ushbu kodning bajarilish algoritmini kuzatamiz. Foydalanuvchi ma'lumotlarni kiritadi, ular qator turiga kiradi va n o'zgaruvchiga beriladi. While sarlavhasi n turini tekshiradi. Dastlab tsiklga kirganingizda n har doim mag'lubiyatga ega bo'ladi, ya'ni int ga teng emas. Shuning uchun, mantiqiy ifoda to'g'ri keladi, bu sizga loop tanasiga o'tishga imkon beradi.

Bu erda, sinash filialida, mag'lubiyatni butun son turiga o'tkazishga harakat qilinadi. Agar u muvaffaqiyatli bo'lsa, unda bundan tashqari band o'tkazib yuboriladi va ijro oqimi while sarlavhasiga qaytadi.

Keling, ushbu kodning bajarilish algoritmini kuzatamiz. Foydalanuvchi ma'lumotlarni kiritadi, ular qator turiga kiradi va n o'zgaruvchiga beriladi. While sarlavhasi n turini tekshiradi. Dastlab tsiklga kirganingizda n har doim mag'lubiyatga ega bo'ladi, ya'ni int ga teng emas. Shuning uchun, mantiqiy ifoda to'g'ri keladi, bu sizga TSIKL tanasiga o'tishga imkon beradi.

Bu erda, sinash filialida, mag'lubiyatni butun son turiga o'tkazishga harakat qilinadi. Agar u muvaffaqiyatli bo'lsa, unda bundan tashqari band o'tkazib yuboriladi va oqim shu vaqt sarlavhasiga qaytadi.

Bu erda, try shoxida sinash filialida, mag'lubiyatni butun son turiga o'tkazishga harakat qilinadi. Agar u muvaffaqiyatli bo'lsa, unda bundan tashqari band o'tkazib



yuboriladi va oqim shu vaqt sarlavhasiga qaytadi. Endi n butun son bilan bog'langan, shuning uchun uning turi `int` bo'lib, uni `int` ga tenglashtirib bo'lmaydi. U unga tengdir. Shunday qilib, mantiqiy ifoda turi $(n)! = \text{Int False}$ ni qaytaradi va butun tsikl tugaydi. Bundan tashqari, ijro oqimi dasturning asosiy qismida joylashgan `if-else` operatoriga o'tadi. Bu erda hamma narsa bo'lishi mumkin, shartli ravishda emas.

Endi n butun son bilan bog'langan, shuning uchun uning turi `int` bo'lib, uni `int` ga tenglashtirib bo'lmaydi. U unga tengdir. Shunday qilib, mantiqiy ifoda turi $(n)! = \text{Int False}$ ni qaytaradi va butun tsikl tugaydi. Bundan tashqari, ijro oqimi dasturning asosiy qismida joylashgan `if-else` operatoriga o'tadi. Bu erda hamma narsa bo'lishi mumkin, shartli ravishda emas.

Qaytamiz. Agar `sinash` tanasida raqamga o'g'irishga urinish muvaffaqiyatsiz tugagan bo'lsa va `ValueError` istisno qilingan bo'lsa, u holda dastur oqimi bundan mustasno filialga yuboriladi va bu erda topilgan ifodalarni bajaradi, oxirgi o'zgaruvchidan yana ma'lumotlarni kiritishni so'raydi. `N` o'zgaruvchisi endi yangi qiymatga ega bo'ldi.

Istisno tugagandan so'ng, tsikl boshidagi mantiqiy ifoda yana tekshiriladi. Bu `True` ga qaytadi, chunki n hali ham satrdir.

Qaytaylik. Agar `sinash` tanasida raqamga o'g'irishga urinish muvaffaqiyatsiz tugagan bo'lsa va `ValueError` istisno qilingan bo'lsa, u holda dasturni amalga oshirishi kerak va bundan mustasno filialga yuboriladi va erda topilgan ifodalangan boladi `N` o'zgaruvchisi endi yangi narxga ega bo'ldi.

Istisno tugagandan so'ng, tsikl boshidagi mantiqiy ifoda yana tekshiriladi. Bu `True` ga qaytadi, chunki n hali ham mag'lubiyatdir.

`N`-ning qiymati raqamga muvaffaqiyatli aylantirilgandagina tsikldan chiqish mumkin.

Quyidagi misolni ko'rib chiqing

:

```
total =100
```

```
i =0
```

```
while i <5:
```

```
    n =int(input())
```

```
    total = total - n
```

```
    i = i + 1
```

```
print("Осталось", total)
```

Ushbu dasturda tsikl "tsikl" necha marta bo'ladi, ya'ni qancha takrorlanadi? Javob:

1. Birinchidan, i o'zgaruvchisi 0 ga teng. Tsikl sarlavhasida i <5 sharti tekshiriladi va u to'g'ri. Loop tanasi bajariladi. I qiymatini unga qo'shish orqali o'zgartiradi.

2. Endi i 1 ga teng. Bu beshdan kam va tsiklning tanasi ikkinchi marta bajariladi.

Unda i o'zgaradi, uning yangi qiymati 2 ga teng.

3. Ikki - beshdan kam. Loop tanasi uchinchi marta bajariladi. I qiymati uchta bo'ladi.

4. Uchtasi beshdan kam. Ushbu takrorlashda menga 4 beriladi.

5. To'rtlik hali ham beshdan kam. Bittasi i ga qo'shiladi, endi uning qiymati be

Keyinchalik, tsiklning oltinchi takrorlanishi boshlanadi. I <5 sharti

tekshiriladi. Lekin endi u false qiymatini qaytaradi, tsiklning bajarilishi to'xtatiladi va uning tanasi bajarilmaydi.

Ushbu tsiklning "semantik yuki" bu umumiy o'zgaruvchidan kiritilgan raqamlarni

ketma-ket olib tashlashdir. I o'zgaruvchisi bu holda faqat tsikl takrorlash

hisoblagichining rolini o'ynaydi. Boshqa dasturlash tillarida bunday holatlar uchun

for loop taqdim etiladi, bu "qarshi tsikl" deb nomlanadi. Uning afzalligi shundaki,

hisoblagich o'zgaruvchini tsikl tanasida o'zgartirishga hojat yo'q, uning qiymati fa

sarlavhasida avtomatik ravishda o'zgaradi



Python-da for loopi mavjud. Ammo bu qarshi halqa emas. Python-da, u ketma-ketlik elementlari va boshqa murakkab ob'ektlar bo'ylab takrorlash uchun qo'ljallangan. Ushbu tsikl va ketma-ketliklar keyingi darslarda o'rganiladi.

Hisoblagich bir muddat ixtiyoriy. Jami noldan katta bo'lgan holda raqamlarni kiritish kerakligini tasavvur qiling. Keyin kod quyidagicha bo'ladi:

```
total = 100
while total > 0:
    n = int(input())
    total = total - n
    print("RESUSRS NIHOYASIGA YETDI")
```

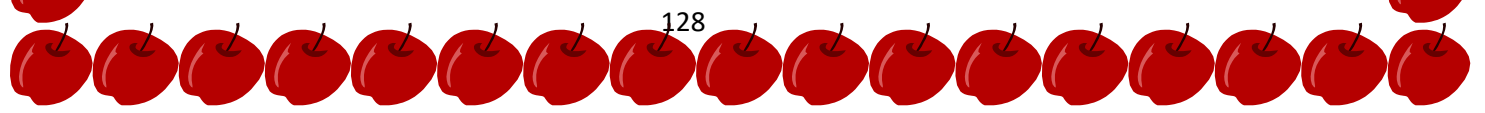
U yerda tsikl necha marta bajariladi? Noma'lum, barchasi kiritilgan qiymatlarga bog'liq. Shuning uchun hisoblagichli tsikl takrorlanish sonini biladi, ammo hisoblagichsiz tsikl bilmaydi.

While tsikli uchun eng muhimi shundaki, uning sarlavhasida tekshirilgan o'zgaruvchilarning qiymatlari tanasida o'zgaradi va hech bo'lmaganda biron kun osilib sarlavhadagi mantiqiy ifoda False-ni qaytaradigan holat bo'ladi. Aks holda, tsikl osilib qoladi.

Yoh 1. Total = total - n va i = i + 1 ifodalarida bir xil o'zgaruvchini takrorlash shart emas. Pythonda bunday iboralarni yozishning stenografik usuli qabul qilinadi: total -= n va i += 1.

Yoh 2. Hisoblagichni ishlatishda uning bittaga ko'payishi shart emas, lekin istalgan o'zgaruvchida istalgan qiymatga o'zgarishi mumkin. Masalan, agar siz 100 dan 0 gacha bo'lgan beshga ko'paytirilgan raqamlarni ko'rsatishingiz kerak bo'lsa, u holda hisoblagichning o'zgarishi i = i - 5 yoki i -= 5 bo'ladi.

Yoh 3. Hisoblagich uchun i identifikatoriga ega o'zgaruvchini ishlatish shart emas. Hisoblagich o'zgaruvchisini xohlaganingizcha nomlashingiz mumkin. Biroq,



dasturlashda shunday qabul qilinganki, hisoblagichlar i va j nomlari bilan belgilanadi (ba'zida ikkita hisoblagich bir vaqtning o'zida talab qilinadi).

Amaliy ish

1. Umumiy o'zgaruvchi manfiy bo'lmasligi uchun darsdagi so'nggi kodni o'zgartiring. Masalan, oldingi olib tashlanishlardan so'ng uning qiymati 25 ga teng bo'ldi. Foydalanuvchi 30 raqamini kiritadi. Ammo dastur ayirboshlashni amalga oshirmaydi, aksincha operatsiya yaroqsiz degan xabarni chiqaradi va keyin tsikldan chiqadi.
2. Vaqt siklidan foydalanib, 2 raqami uchun uning kuchini 0 dan 20 gacha ko'rsating. Pythonda ko'rsatkichlar `**` bilan belgilanadi. Chiqarilgan qism

```
...  
32  
64  
128  
256  
512  
1024  
...
```

Ro'yhatelementlari yig'indisini hisoblash gamisol

```
# Ro'yhat elementlari yig'indisini hisoblash
```

```
# Berilgan ro'yhat
```

```
T = [2.8, 3.5, 4.9, 0.01, 2.34]
```

```
s=0
```

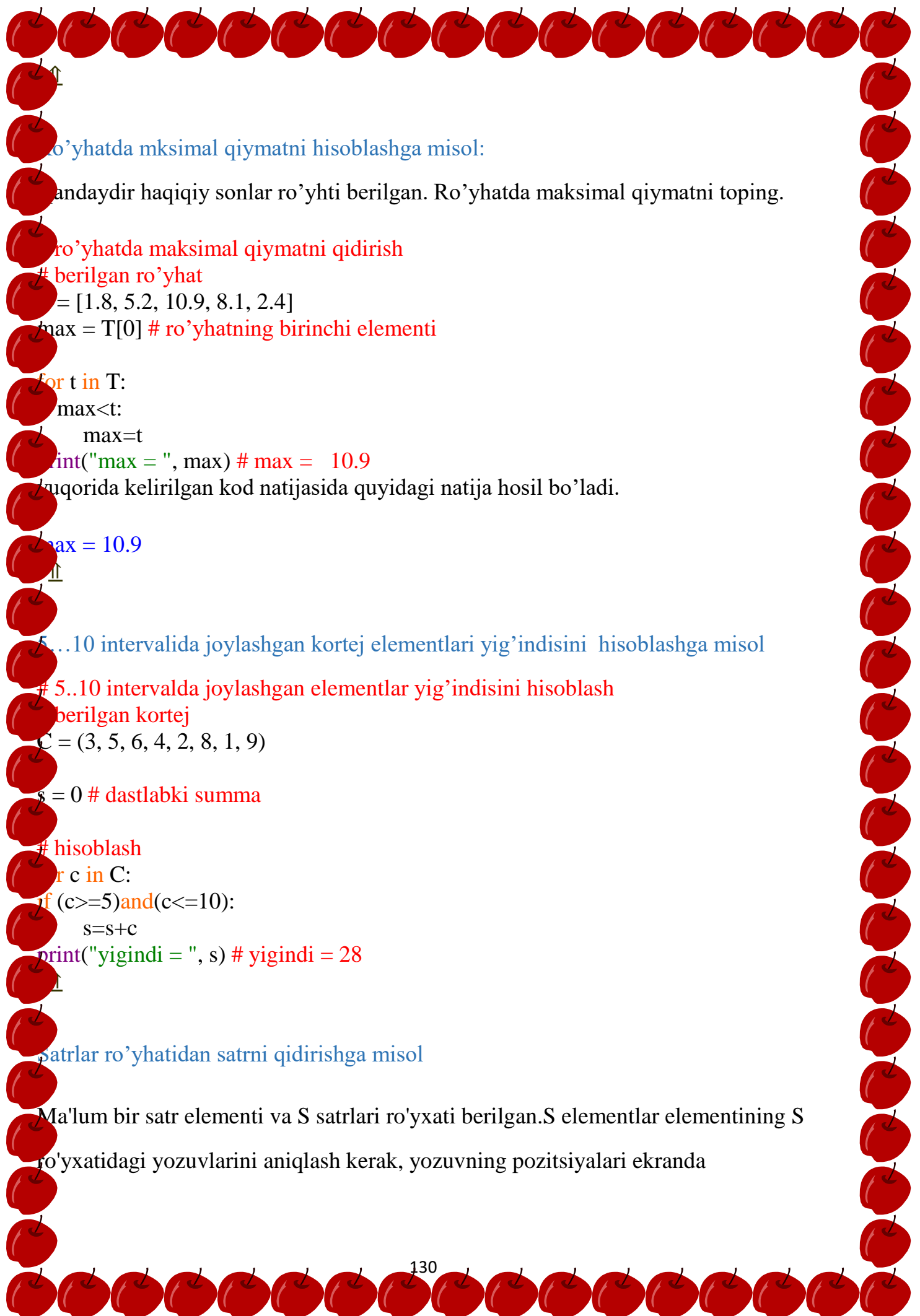
```
for t in T:
```

```
    s = s + t
```

```
print("s = ", s) # s = 13.549999999999999
```

Yuqorida keltirilgan kodni bajarilish natijasida keying natij aks etadi:

```
s = 13.549999999999999
```



Ro'yhatda mksimal qiymatni hisoblashga misol:

Andaydir haqiqiy sonlar ro'yhti berilgan. Ro'yhatda maksimal qiymatni toping.

Ro'yhatda maksimal qiymatni qidirish

berilgan ro'yhat

T = [1.8, 5.2, 10.9, 8.1, 2.4]

max = T[0] # ro'yhatning birinchi elementi

for t in T:

max < t:

max = t

print("max = ", max) # max = 10.9

Yuqorida kelirilgan kod natijasida quyidagi natija hosil bo'ladi.

max = 10.9

5..10 intervalida joylashgan kortej elementlari yig'indisini hisoblashga misol

5..10 intervalda joylashgan elementlar yig'indisini hisoblash

berilgan kortej

C = (3, 5, 6, 4, 2, 8, 1, 9)

s = 0 # dastlabki summa

hisoblash

for c in C:

if (c >= 5) and (c <= 10):

s = s + c

print("yigindi = ", s) # yigindi = 28

Satrlar ro'yhatidan satrni qidirishga misol

Ma'lum bir satr elementi va S satrlari ro'yxati berilgan. S elementlar elementining S

ro'yxatidagi yozuvlarini aniqlash kerak, yozuvning pozitsiyalari ekranda

ko'rsatilishi kerak. Agar element S ro'yxatida hech qachon uchramasa, unda tegishli xabarni chiqaring.

Ushbu muammoni for loop yordamida hal qilish, masalan, quyidagilar bo'lishi mumkin:

```
# satrlar ro'yhatidan satrni qidirish
```

```
# berilgan ro'yhat
```

```
S = ['abc', 'def', 'hij', 'abc', 'fhg', 'jkl', 'jpr']
```

```
item = str(input("toppish kerak bo'lgan elementni kiiting: "))
```

```
P=[] # S ro'yhatga item elementini Kirish pozitsiyalari
```

```
# hisoblash
```

```
i=0
```

```
for s in S:
```

```
if s==item:
```

```
    P=P+[i]
```

```
i=i+1
```

```
if len(P)>0:
```

```
print("elemntni kirish pozitsiyalari ", item, " ro'yhatiga")
```

```
print(P)
```

```
else:
```

```
print("elemnti ", item, " ro'yhatda yo'q")
```

yuqoridagi kodning natijasi quyidagicha bo'lishi mumkin:

Topish kerak bo'lgan elementni toping: abc

abc elementni ro'yhatga [0, 3] kirish pozitsiyasi

↑.

Else blokidan iborat for operatorining qo'llanilishi

Misolda kortejning berilgan elementi mos ma'lumotni akslanishi bilan aniqlanadi

```
# kortejdagi berilgan elementning mavjudligining aniqlanishi
```

```
# berilgan kortej
```

```
C = (3, 5, 6, 4, 2, 8, 1, 9)
```

```
# elementni kiritish
```

```
item = int(input("navbatdagi elemntni kiiting: "))
```

```
# hisoblash
```

```
for c in C:
```


For va while operatorlari va ularga dasturlar

Faktorialni hisoblash.

Faktorial 1 dan boshlab, berilgan songa qadar sonlarning ko'patmasini aniqlaydi. Misol uchun 5 faktorial $1 * 2 * 3 * 4 * 5 = 120$ ga teng. Faktorial formulasini $n! = 1 * 2 * \dots * n$ yoki $n! = 1 * \dots * (n-2) * (n-1) * n$ ko'rinishdagi formulalar orqali aniqlash mumkin.

Faktorialni aniqlash dasturini tuzish uchun takrorlash operatorlaridan fodalanish mumkin.

While operatori orqali:

```
*12.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/12.py (3.5.1)*
File Edit Format Run Options Window Help
n = int(input())

factorial = 1
while n > 1:
    factorial *= n
    n -= 1

print(factorial)
Ln: 9 Col: 0
```

For operatori orqali:

```
*12.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/12.py (3.5.1)*
File Edit Format Run Options Window Help
n = int(input())

factorial = 1

for i in range(2, n+1):
    factorial *= i

print(factorial)
Ln: 9 Col: 0
```

Qaytish operatori orqali:

```
*12.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/12.py (3.5.1)*
File Edit Format Run Options Window Help
def fac(n):
    if n == 0:
        return 1
    return fac(n-1) * n

print(fac(5))
Ln: 8 Col: 0
```



Python dasturlash tili faktorialni hisoblash uchun yuqorida ko'rsatilgan dastur kodlarini tuzishni qisqartirish uchun **math** modulidagi **factorial()** funksiyasini ishlatib chiqqan.

```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
Python 3.5.1 (v3.5.1:37a07cee5969, Dec 6 2015, 01:54:25) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> import math
>>> math.factorial(8)
40320
Ln: 6 Col: 4
```



Fibonachi sonlarini hisoblash.

Fibonachi sonlari shunday sonlar qatoriki o'zidan oldingi kelgan ikki sonning yig'indisidan tashkil topadi. Misol uchun 0,1,1,2,3,5,8,13,21,...

Fibonachi sonlarini topish uchun quyidagi formuladan foydalanamiz:

$$F_1 = 1$$
$$F_2 = 1$$
$$F_n = F_{n-1} + F_{n-2}$$

Fibonachi sonlarining yig'indisini topish dasturi quyidagi ko'rinishda tuzish mumkin:

```
*13.py - C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/13.py (3.5.1)*
File Edit Format Run Options Window Help
fib1 = 1
fib2 = 1

n = input("Fibonachi raqamlar chegarasini kiriting: ")
n = int(n)

i = 0
while i < n - 2:
    fib_sum = fib1 + fib2
    fib1 = fib2
    fib2 = fib_sum
    i = i + 1

print(fib2)
Ln: 15 Col: 0
```

Natija:



```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
Python 3.5.1 (v3.5.1:37a07cee5969, Dec 6 2015, 01:54:25) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:/Users/DILSHOD/AppData/Local/Programs/Python/Python35/13.py ====
Fibonacci raqamlar chegarasini kiriting: 9
34
Ln: 7 Col: 4
```

2. Spiral.

Spirallarni chizish jarayoni va tayyor rasmlarni olish nafaqat o'quvchilarda balki maktabgacha yoshdagi bolalarda ham qiziqish o'rgatadi. O'quvchilarga esa spirallarni ko'rsatib, keyin kodlarni terib, ularni namoyish qilish orqali ham dasturlashtirishga, kod yozishga qiziqish o'rgatish eng qiziqarli va ahamiyatli uslublardan hisoblanadi.

Python toshbaqasi bilan spiral shaklida chizmalarni chizishni o'rganaylik. bunga doir kodlarni yozamiz va ishga kirishamiz.

Mening toshbaqa Angie kodimda nima yomon? Men uning Bredning to'rtburchagiga aylanishini istayman.

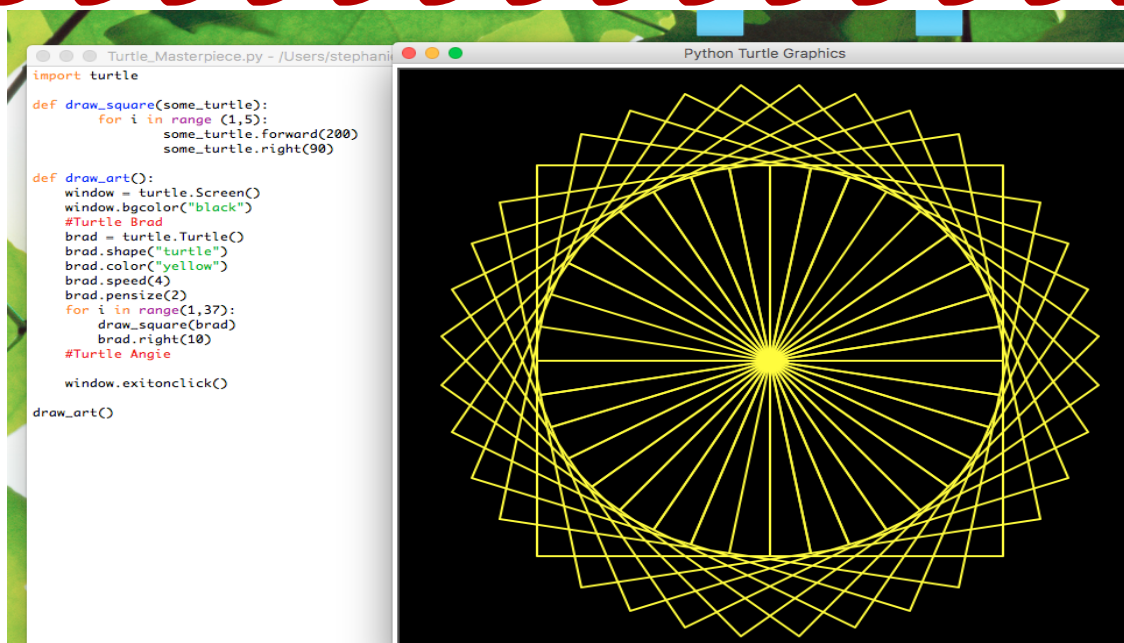
Mening kodim:

```
...
import turtle

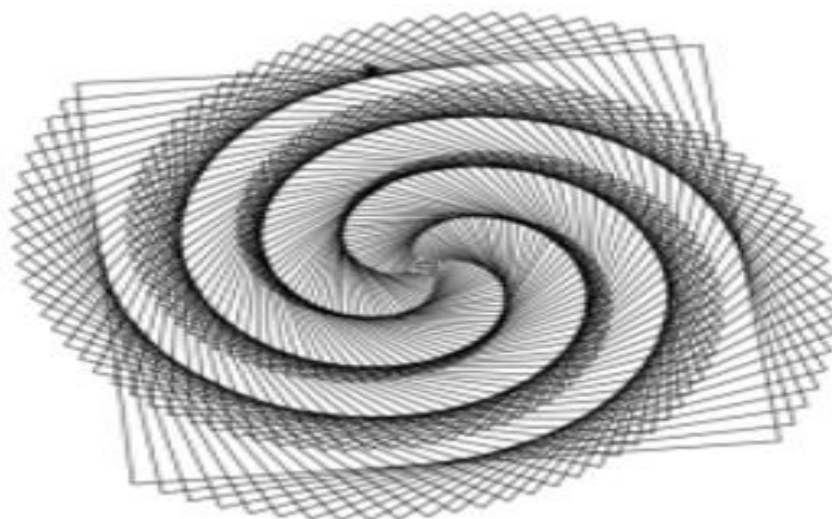
def draw_square(some_turtle):
3.
4. for i in range (1,5):
5.     some_turtle.forward(200)
6.     some_turtle.right(90)
7.
8. def draw_art():
9.     window = turtle.Screen()
10.    window.bgcolor("black")
11. #Turtle Brad
```

```
12. brad = turtle.Turtle()
13. brad.shape("turtle")
14. brad.color("yellow")
15. brad.speed(6)
16. brad.pensize(2)
17. for i in range(1,37):
18.     draw_square(brad)
19.     brad.right(10)
20. #Turtle Angie
21. angie = turtle.Screen()
22. angie.shape("turtle")
23. angie.color("blue")
24. angie.speed(5)
25. angie.pensize(2)
26. size=1
27. while(True):
28.     angie.forward(size)
29.     angie.right(91)
30.     size = size +1
31.
32. window.exitonclick()
33.
34. draw_art()
```

Mana, u qanday ko'rinishini xohlaganim haqidagi fotosuratlar. Men Bredning
yashqi tomoni ko'rinishini, so'ngra ichidagi aylana spiralni o'z ichiga olishni
xohlayman. U birlashtirilgan spiral tasvirga o'xshash bo'lishi kerak. Rahmat!
spiral kodi va natijaning namoyishi.

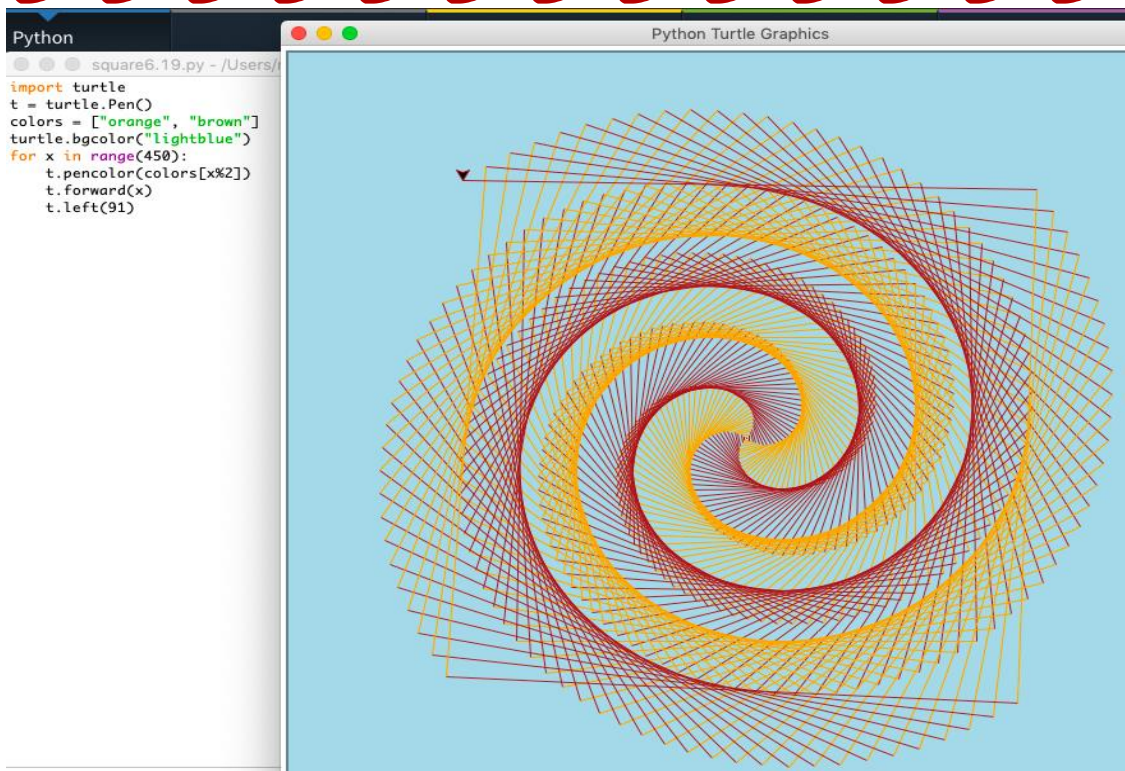


2spiral va uning kodi, natijaning tasviri.

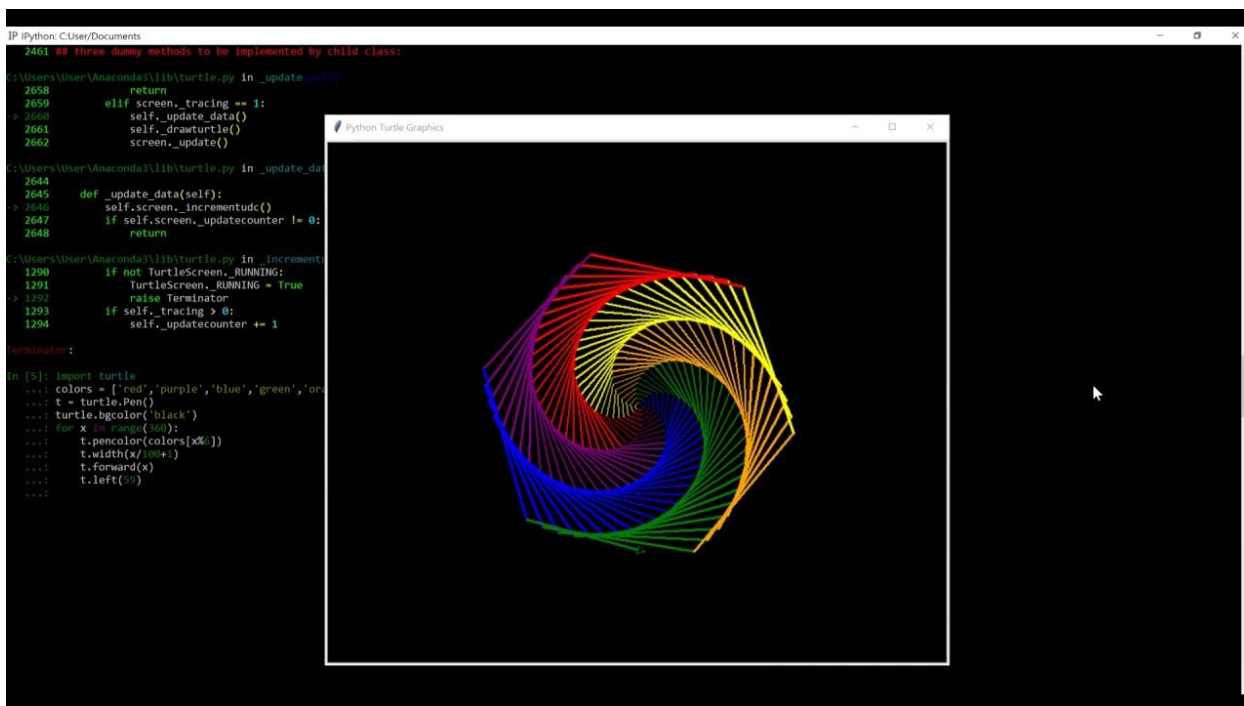


3. Dastrlashga misol va masalalar.

Quyidagi kodni terib, uning natijasida spiralni tasvirini hosil qilish muumkin.



Yana bitta chiroyli spiral manzarasinin namoyish etish uchun ajoyib kodni keltitamiz.

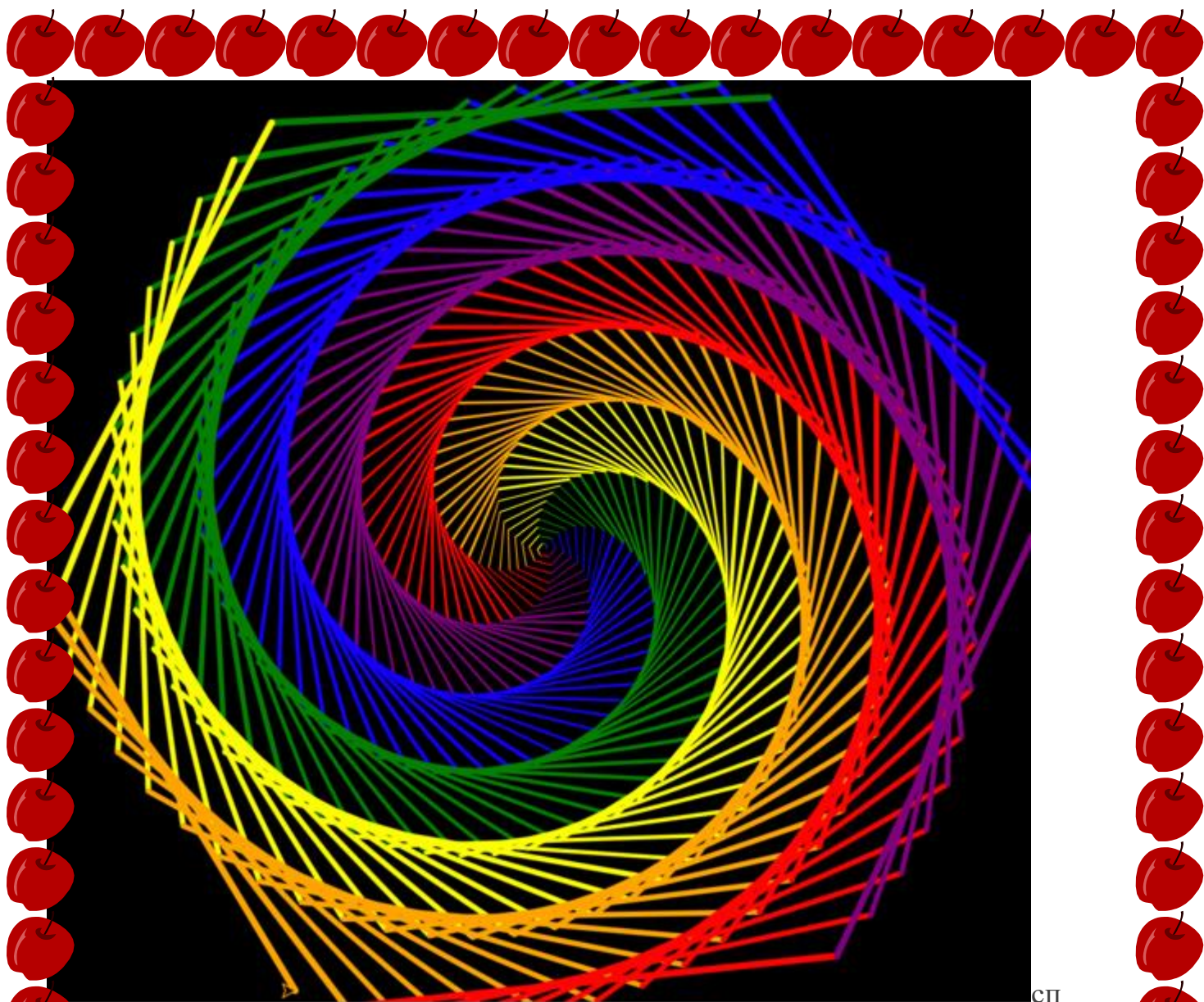


Yuqorida spiallarni chizishga doir materiallar va ularga doir dasturlarning kodlarining birida biz toshbaqa moduli bilan tanishdik, u bilan siz turli xil

geometrik narsalarni chizishingiz mumkin. Ushbu bo'limda biz modulning imkoniyatlari bilan tanishamiz. Biz naqshlar chizamiz va ranglar bilan tajriba o'tkazamiz.



1. Spiralsimon naqshlar

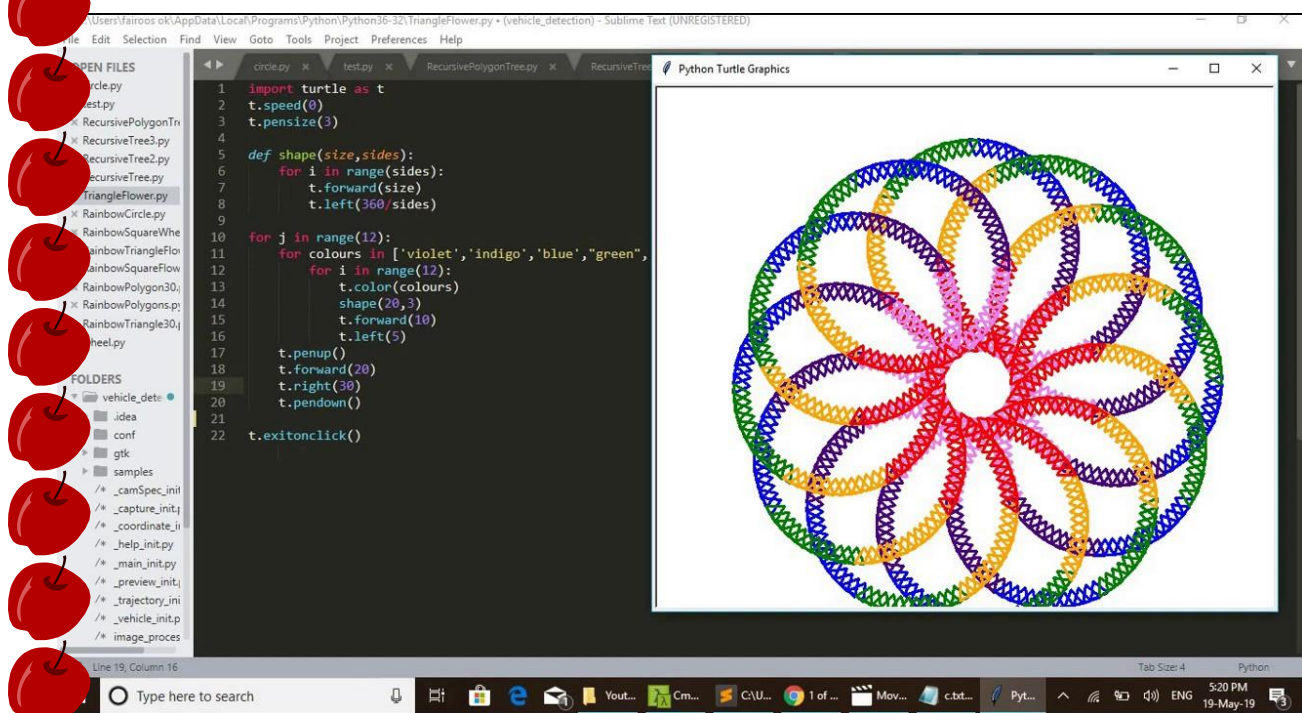
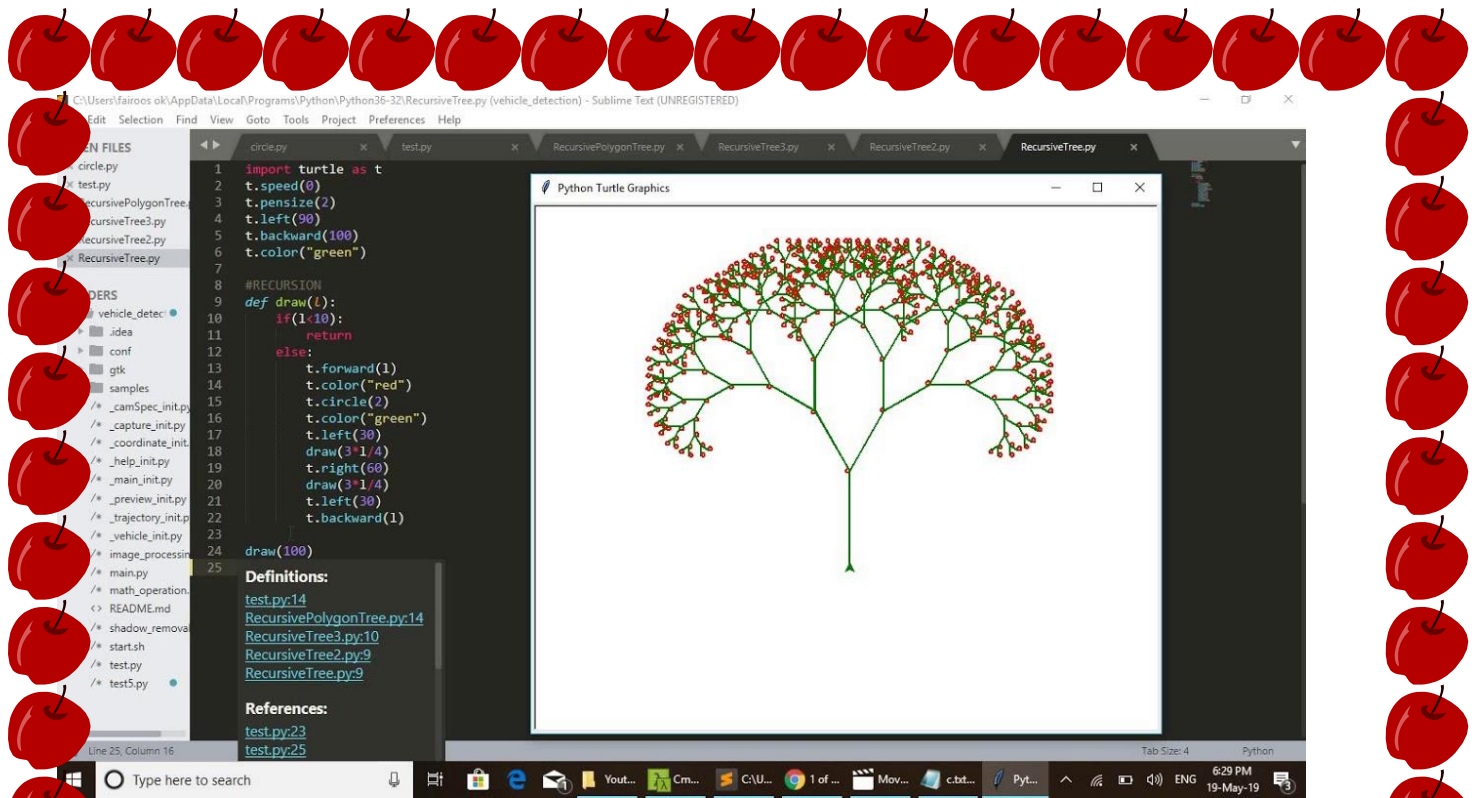


СИ

SPIRALSIMON NAQSH KODINI TERIB SPIRALNING NAMOYISHINI
TOMOSHA QILING:

```
import turtle
colors=['red', 'purple', 'blue', 'green', 'yellow', 'orange']
t=turtle.Pen()
turtle.bgcolor('black')
for x in range(360):
    t.pencolor(colors[x%6])
    t.width(x/100+1)
    t.forward(x)
    t.left(59)
```





KODNI KIRITGANINGIZDAN SO'NG, F5 TUGMASININ BOSING.

```

colors=['red', 'purple', 'blue', 'green', 'yellow', 'orange']
turtle.Pen()
turtle.bgcolor('black')
for x in range(360):
    t.pencolor(colors[x%6])
    t.width(x/100+1)
    t.forward(x)
    t.left(59)

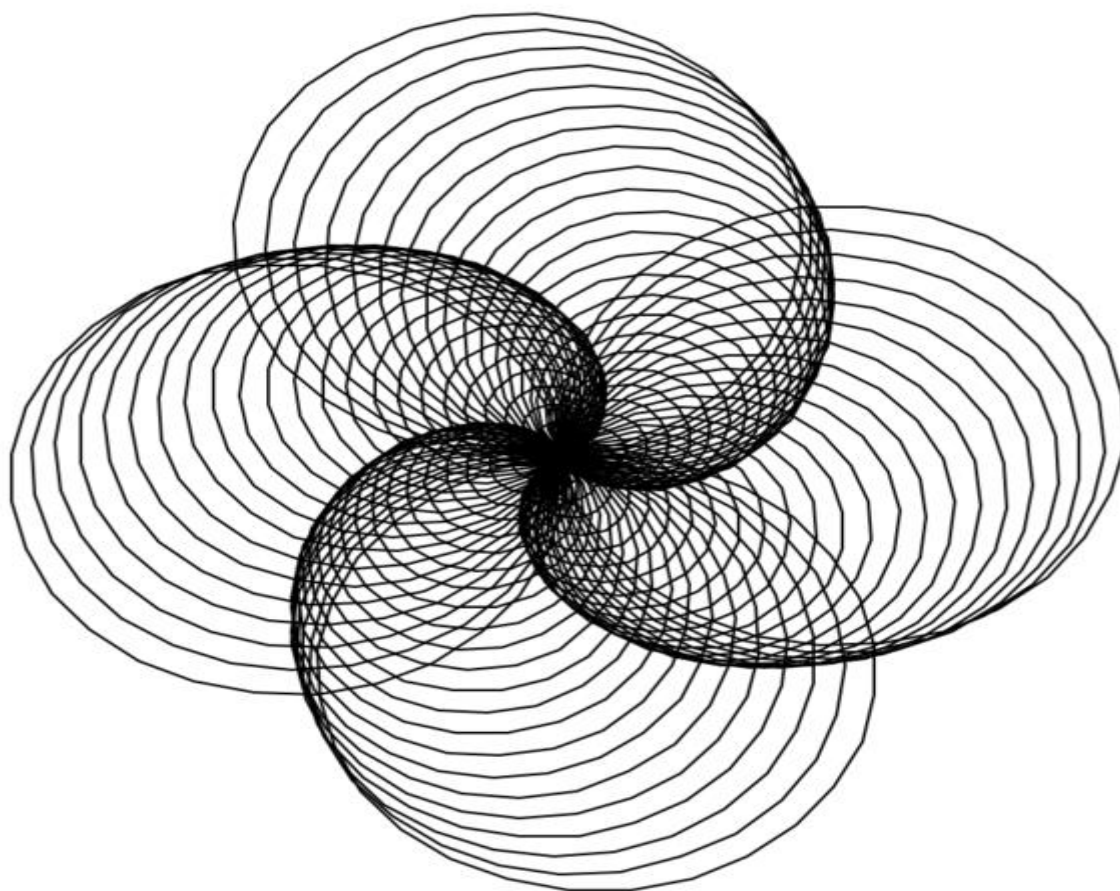
```

Kodni sizga qulay bo'lsin deb nusxa qilindi. F5 tugmasi bilan uni ihga tushiring.qulaylik uchun kodni nusxa qilindi. Uni f5 tugmasi bilan ishga tushiring. Funktsiyalar ichida parametrlarni o'zgartirib, savollarga javob bering:

Qaysiparametrga javob beradi:

- t.pencilcolor? spiraldarangni qanday o'zgartirish mumkin?
- t.width?
- t.forward?

2. SPIRALSIMON AYLANALAR



```
import turtle
t=turtle.Pen()
for x in range(100):
    t.circle(x)
    t.left(45)
```



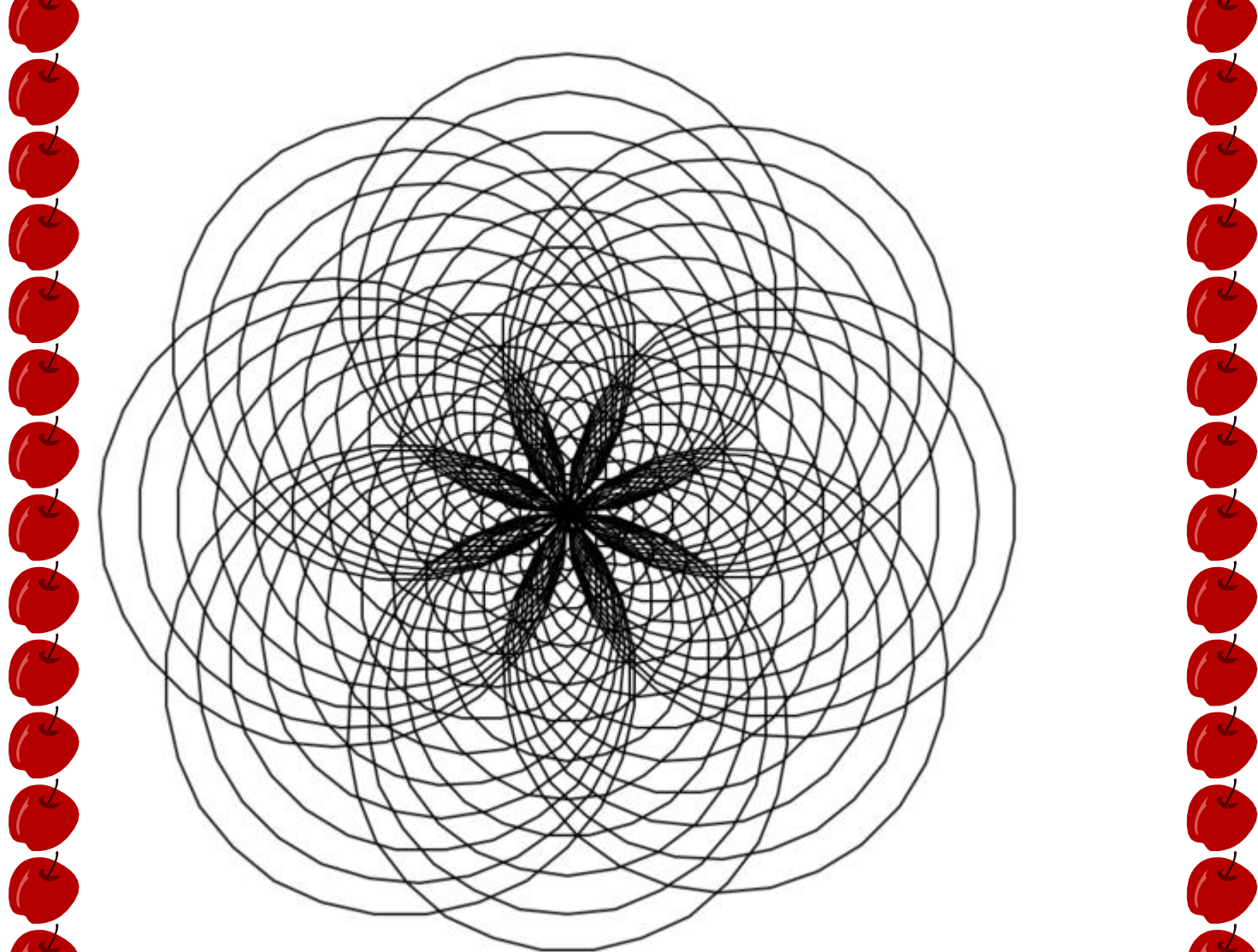
Ushbu misolda men tuval fonini o'zgartirmadim va turli xil ranglarni qo'llamadim.

Ushbu misolda doiralalar 91 daraja ofset bilan chizilgan kod amalga

qo'yiladi.

Vazifa:

1. Nima uchun ofset 90 emas, 91 daraja?
2. Kodni 8 ta aylana bo'ladigan qilib o'zgartiring.



SHARTLAR

1. if sharti.

OPERATORI



IF OPERATORINING SINTAKSISI QUYIDAGICHA:

if condition:

<indented statement 1>

<indented statement 2>

<non-indented statement>

Bayonotning birinchi satri, ya'ni if sharti: if sharti, shart esa mantiqiy ifoda bo'lib, u True yoki False-ni qaytaradi. Keyingi satr - ko'rsatmalar to'plami. Blok - bu bir yoki bir nechta ko'rsatmalar. Agar u if shartiga amal qilsa, blok if blok deb nomlanadi.

Shuni ta'kidlash kerakki, if blokidagi har bir ibora if so'zidan bir xil chuqurchaga ega. Ko'pgina tillar, masalan, C, C ++, Java va PHP, blokning boshi va oxirini ko'rsatish uchun jingalak qavslardan ({}) foydalanadi, ammo Python indentatsiyadan foydalanadi.

Har bir bayonotda bir xil sonli bo'sh joy bo'lishi kerak. Aks holda, dastur sintaksis xatosini qaytaradi. Python hujjatlari 4 bo'sh joyni ajratishni tavsiya qiladi. Ushbu tavsiya va buning uchun dolzarbdir.

U qanday ishlaydi:

If ifodasi bajarilganda shart tekshiriladi. Agar shart to'g'ri bo'lsa, unda if blokidagi barcha gaplar bajariladi. Ammo agar shart noto'g'ri bo'lib chiqsa, ushbu blokdagi barcha ko'rsatmalar o'tkazib yuboriladi.

Agar indentatsiyalanmagan if bandidan keyin keltirilgan gaplar if blokiga tegishli emas. Masalan, <non-intenden operator> if blokining tarkibiy qismi emas, shuning uchun u baribir bajariladi.

MISO UCHUN:



```
number =int(input("Sonni kiriting: "))
```

```
if number >10:
```

```
print("Son 10dan katta")
```

Birinchi natija:

Sonni kiriting: 100

Son 10 dan katta

Ikkinchi natija:

Sonni kiriting: 5

Shuni ta'kidlash kerakki, ikkinchi holatda, agar shart to'g'ri kelmasa, if blok
ichidagi bayonot o'tkazib yuboriladi. Ushbu misolda if bloki bitta gapdan iborat,
 ammo shuncha ko'p bo'lishi mumkin, asosiysi indentatsiya qilishdir.

Quyidagi kodni ko'rib chiqing:

```
number =int(input("Sonni kiriting: "))
```

```
if number >10:
```

```
print("birinchi satr")
```

```
print("ikkinchi satr")
```

```
print("uchinchi satr")
```

```
print("har safar siz dasturni ishga tushirganingizda amalga ")
```

```
print("Ohiri")
```

Birinchi natija:

Sonni kiriting: 45

Birinchi satr

Ikkinchi satr

Uchinchi satr

Har safar siz dasturni ishga tushirganingizda amalga oshiriladi

ohiri

Ikkinchi natija:

Sonni kiriting: 4

Har safar siz dasturni yuklaganingizda

ohiri

Bu erda ta'kidlash kerak bo'lgan muhim narsa shundaki, faqat 3, 4 va 5-satrlardagi bayonotlar if blokiga tegishli. Shuning uchun, ular faqat if sharti to'g'ri bo'lgan taqdirdagina bajariladi. Ammo 7 va 8-qatorlardagi ko'rsatmalar baribir bajariladi.

Python konsolida boshqaruv buyrug'larini o'sha erda ishlatganingizda boshqacha ta'sir qiladi. Shuni esda tutish kerakki, davom etish operatori (\) ifodani bir necha qatorga bo'lish uchun ishlatiladi. Ammo bu boshqarish operatorlari uchun kerak emas. Python tarjimoni if shartidan keyin Enter tugmachasini bosgan bo'lsangiz, ko'p satrli rejimni avtomatik ravishda faollashtiradi. Masalan:

```
>>>
```

```
>>> n =100
```

```
>>> if n >10:
```

```
...
```

If sharti bo'lgan qatorda Enter tugmachasini bosgandan so'ng, buyruq satri >>> da ... ga o'zgartiriladi. Python konsolida ko'p qatorli ko'rsatmalar uchun ... ko'rsatiladi.

Bu shuni anglatadiki, boshlangan ko'rsatma hali ham to'liq emas.

iborasini tugatish uchun if blokiga yana bir bayonot qo'shishingiz kerak:

```
>>
```

```
>> n = 100
```

```
>> if n > 10:
```

```
print("n > 10")
```

Python avtomatik ravishda joy qoldirmadi. Buni o'zingiz qilishingiz kerak.

ko'rsatmani kiritishni tugatgandan so'ng, ko'rsatmani bajarish uchun Enter

bosmasini ikki marta bosishingiz kerak. Shundan so'ng, konsol asl holatiga qaytadi.

```
>>
```

```
>> n = 100
```

```
>> if n > 10:
```

```
print("n esa 10" dan katta)
```

```
    esa 10 dan katta
```

```
>>
```

shbu dasturlarning barchasi to'satdan tugaydi, agar shart to'g'ri bo'lmasa, hech

narsa ko'rsatmaydi. Ammo aksariyat hollarda foydalanuvchiga hech bo'lmaganda

biror narsa ko'rsatilishi kerak. Buning uchun if-else ifoda operatori ishlatiladi.

Bul ifodalarini uchratamiz.

Mantiqiy ifodalar va mantiqiy ma'lumotlar turi

ko'pincha haqiqiy hayotda biz bayonotga qo'shilamiz yoki rad etamiz. Masalan,

sizga 3 va 5 raqamlarining yig'indisi 7 dan katta ekanligi aytilsa, siz rozi bo'lasiz:

"Ha, bu to'g'ri". Agar kimdir uch va beshning yig'indisi ettidan kam deb da'vo qilsa, siz bunday bayonotni yolg'on deb bilasiz.

Bunday iboralar faqat ikkita mumkin bo'lgan javoblarni taklif qiladi - yoki ifoda to'g'ri yoki to'g'ri deb baholanganda "ha", yoki noto'g'ri yoki yolg'on deb baholanganda "yo'q". Dasturlash va matematikada, agar ifodani baholash natijasi faqat to'g'ri yoki yolg'on bo'lishi mumkin bo'lsa, unda bunday ifoda mantiqiy deb nomlanadi.

Masalan, $4 > 5$ ifoda mantiqiy, chunki u rost yoki yolgonga baholanadi. $4 + 5$ ifodasi mantiqiy emas, chunki uning bajarilish natijasi raqamdir.

O'tgan darsda biz ma'lumotlar uch turi - tamsayilar va haqiqiy sonlar, shuningdek satrlar bilan tanishdik. Bugun biz to'rtinchisini - mantiqiy ma'lumotlar turini (bool turi) tanishtiramiz. U mantiqiy deb ham yuritiladi. Ushbu turdagi faqat ikkita mumkin bo'lgan qiymatlar mavjud: True va False.

```
>>> a = True
>>> type(a)
<class 'bool'>
>>> b = False
>>> type(b)
<class 'bool'>
```

Bu erda a o'zgaruvchiga True qiymati berilgan, so'ngra uning turi o'rnatilgan Python type () funksiyasi yordamida tekshirilgan. Tarjimon bu bool sinf o'zgaruvchisi ekanligini xabar qildi. Bu holda "sinf" va "ma'lumotlar turi" tushunchalari bir xil. B o'zgaruvchisi ham mantiqiy qiymat bilan bog'liq.

Dasturlashda False odatda nolga, True esa biriga tenglashtiriladi. Buni tekshirish uchun mantiqiy qiymatni butun son turiga o'zgartirishingiz mumkin:

```
>>> int(True)
1
>>> int(False)
0
```



eskari ham bo'lishi mumkin. Biror ifodani bul tipli ifodagao'zgartirish mumkin:

```
>> bool(3.4)
True
>> bool(-150)
True
>> bool(0)
False
>> bool(' ')
True
>> bool("")
False
```

bu yerda ham qoida amaldadir, 0 va bo'shliq bo'lmagan bo'lmagan har bir narsa haqiqatdir.

mantiqiy operatorlar

Tabiiy tilda gaplashsak (masalan, rus tilida), biz "teng", "ko'proq", "kamroq" so'zlari bilan taqqoslashni belgilaymiz. Dasturlash tillarida matematikada ishlatiladigan belgilarga o'xshash maxsus belgilar qo'llaniladi: > (katta), <(kichik), >= (katta yoki teng), <= (kichik yoki teng), == (teng), != (teng emas).

Pythonda bitta tenglik belgisi bilan belgilanadigan o'zgaruvchiga qiymat berish operatsiyasini va taqqoslash operatsiyasini (ikkita teng belgi) aralashtirmang.

Topshiriq va taqqoslash har xil operatsiyalar.

```
>> a = 10
>> b = 5
>> a + b > 14
True
>> a < 14 - b
False
>> a <= b + 5
True
>> a != b
True
>> a == b
```





```
False
```

```
>>> c = a == b
```

```
>>> a, b, c
```

```
(10, 5, False)
```

Ushbu misolda $c = a == b$ ifodasi ikkita kichik ifodadan iborat. Birinchidan, a va b o'zgaruvchilarni taqqoslash ($==$) mavjud. Shundan so'ng mantiqiy operatsiya natijasi c o'zgaruvchiga beriladi. A, b, c ifodasi o'zgaruvchilar qiymatlarini ekranga chiqaradi.

rakkab mantiqiy iboralar $KByte > = 1023$ oddiy mantiqiy ifodalar oddiy, chunki ular faqat bitta mantiqiy amalni bajaradilar. Ishlaydi, amalda davolash murakkab iboralarga yakunlanadi. Ikki oddiy iboraning natijalariga qarab sizga "Ha" yoki "Yo'q" javobini olish kerak bo'lishi mumkin. Masalan, "nazoratida qor yoki yomg'ir yog'moqda", "yangiliklar o'zgaruvchisi 12 dan katta va 20 dan kam"

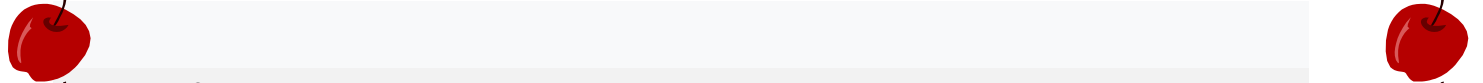
Bunday hollarda ikki yoki undan ortiq oddiy mantiqiy ifodalarni birlashtiradigan maxsus operatorlardan foydalaniladi. Ikki operator keng qo'llaniladi - mantiqiy V (va) va OR (yoki) deb nomlanadi.

Va operatoridan foydalanganda ROSTNI olish uchun, ushbu operator bog'laydigan ikkala oddiy ifodaning natijalari ham to'g'ri bo'lishi kerak. Agar hech bo'lmaganda bitta holatda natija noto'g'ri bo'lsa, unda butun kompleks ifoda Yolg'on bo'ladi.

Yoki operatoridan foydalanganda Haqiqiylikni olish uchun murakkab tarkibga kiritilgan kamida bitta oddiy ifodaning natijasi to'g'ri bo'lishi kerak. Yoki operator holatida murakkab ibora, uni tashkil etuvchi ikkala sodda ibora yolg'on bo'lgandagina yolg'on bo'ladi.

Aytaylik, x o'zgaruvchiga 8 ($x = 8$), y o'zgaruvchiga 13 ($y = 13$) qiymat berildi.

Mantiqiy ifoda $y < 15$ va $x > 8$ quyidagicha bajariladi. Birinchidan, $y < 15$ ifodasi bajariladi, natijasi True bo'ladi. Keyin $x > 8$ ifoda bajariladi, uning natijasi False bo'ladi. Bundan tashqari, ifoda True va False-ga qisqartiriladi, bu esa False-ni qaytaradi.



```
>>> x = 8
>>> y = 13
>>> y < 15 and x > 8
False
```

Agar biz ifodani quyidagicha yozgan bo'lsak: $x > 8$ va $y < 15$, u ham False qaytadi.

chunki bunga

tiyoj qolmaydi. Axir, birinchi oddiy mantiqiy ifoda ($x > 8$) allaqachon false

qiymatini qaytargan, bu va operatori holatida butun ifodani yolg'onga aylantiradi.

Yoki operatorida, ikkinchi oddiy ifoda birinchisi noto'g'ri bo'lsa tekshiriladi, agar

birinchisi allaqachon to'g'ri bo'lsa, tekshirilmaydi. Butun ifoda to'g'ri bo'lishi uchun

faqat Haqiqat etarli bo'lganligi sababli, uning qaysi tomoni yoki turgani muhim

emas.



```
>>> y < 15 or x > 8
True
```

Python-da bir xil mantiqiy bo'lmagan operator, ya'ni inkor mavjud. U haqiqatni

yolg'onga, yolg'onni esa haqiqatga aylantiradi. U unariydir, chunki u ikkilik va va

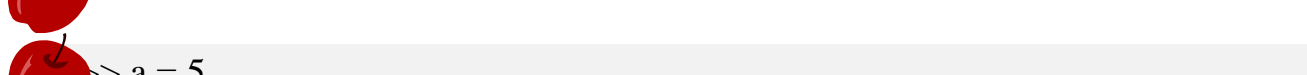
yoki holatidagi kabi uning o'ng va chap tomonlariga emas, balki undan keyingi

tta iboraga tegishli.

```
>>> not y < 15
False
```

Bu yerda `y < 15` **True** qiymatni qaytaradi. Buni inkor etib, biz **False** qiymatini

qaytarimiz.



```
>>> a = 5
>>> b = 0
>>> not a
False
>>> not b
True
```



5 soni true siatida amalda qo'llaniladi, haqiqatning inkori esa false ni
yani yolg'onni beradi. 0 **False** ga tenglashadi. **False** ning inkori esa

Tog'rini beradi.

Adabiyotlar ro'yhati:

1. **Лутц М. Изучаем Python**, 4-е издание. – Пер. с англ. – СПб.: Символ-Плюс, 2011. – 1280 с.
2. **Златопольский Д.М. Основы программирования на языке Python**. – М.: ДМК Пресс, 2017. – 284 с.
3. **Лутц М. Программирование на Python**, том I, 4-е издание. – Пер. с англ. – СПб.: Символ-Плюс, 2011. – 992 с.
4. **Лутц М. Программирование на Python**, том II, 4-е издание. – Пер. с англ. – СПб.: Символ-Плюс, 2011. – 992 с.
5. **Гэддис Т. Начинаем программировать на Python**. – 4-е изд.: Пер. с англ. – СПб.: БХВ-Петербург, 2019. – 768 с.
6. **Лучано Рамальо Python. К вершинам мастерства**. – М.: ДМК Пресс, 2016. – 768 с.
7. **Свейгарт, Эл. Автоматизация рутинных задач с помощью Python: практическое руководство для начинающих**. Пер. с англ. — М.: Вильямс, 2016. – 592 с.
8. **Рейтц К., Шлюссер Т. Автостопом по Python**. – СПб.: Питер, 2017. – 336 с.: ил. – (Серия «Бестселлеры O'Reilly»).
9. **Любанович Билл Простой Python. Современный стиль программирования**. – СПб.: Питер, 2016. – 480 с.: – (Серия «Бестселлеры O'Reilly»).
10. **Федоров, Д. Ю. Программирование на языке высокого уровня Python: учебное пособие для прикладного бакалавриата / Д. Ю. Федоров**. – 2-е изд., перераб. и доп. – Москва : Издательство Юрайт, 2019. – 161 с. – (Бакалавр. Прикладной курс). – ISBN 978-5-534-10971-9. – Текст: электронный // ЭБС Юрайт [сайт]. URL: <https://urait.ru/bcode/437489> (дата обращения: 13.02.2020).
11. **Шелудько, В. М. Основы программирования на языке высокого уровня Python: учебное пособие / В. М. Шелудько**. – Ростов-на-Дону, Таганрог: Издательство Южного федерального университета, 2017. – 146 с. – ISBN 978-5-9275-2649-9. – Текст: электронный // Электронно-библиотечная система IPR BOOKS: [сайт]. – URL: <http://www.iprbookshop.ru/87461.html> (дата обращения: 13.02.2020). – Режим доступа: для авторизир. пользователей
12. **Шелудько, В. М. Язык программирования высокого уровня Python. Функции, структуры данных, дополнительные модули: учебное пособие / В. М. Шелудько**. – Ростов-на-Дону, Таганрог: Издательство Южного федерального университета, 2017. – 107 с. – ISBN 978-5-9275-2648-2. – Текст: электронный // Электронно-библиотечная система IPR BOOKS: [сайт]. – URL: <http://www.iprbookshop.ru/87530.html> (дата обращения: 13.02.2020). – Режим доступа: для авторизир. пользователей
13. **Доусон М. Программируем на Python**. – СПб.: Питер, 2014. – 416 с.
14. **Прохоренок Н.А. Python 3 и PyQt. Разработка приложений**. – СПб.: БХВ-Петербург, 2012. – 704 с.
15. **Пилгрим Марк. Погружение в Python 3 (Dive into Python 3 на русском)**
16. **Прохоренок Н.А. Самое необходимое**. — СПб.: БХВ-Петербург, 2011. — 416 с.