

Matnli kodlash (Text Coding)
dunyasiga sayohat qilamiz!

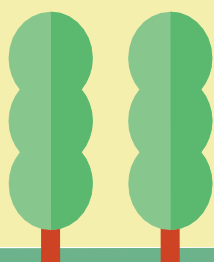
Entry-Python



RoboticsWare & UFE

Mundarija

1. Entry-Pythonni boshlash	4
2. Kompyuterda ma`lumotlarni ko'rsatish usuli	8
3. Muammoni hal qilmoqchi bo'lsangiz ma`lumotni kiritishdan boshlang!	15
4. Bir nechta ma`lumotni saqlay olaman deysizmi?	20
5. Dasturlash orqali hisoblashni oson va tez ishlash	28
6. Kompyuterjon, qachongacha takrorlamoqchisiz?	35
7. Ikkitadan bittasini albatta tanlash kerak!	40
8. Up(Yuqoriga) yoki Down(Pastga) o'yinini yasab ko'ramiz	44



Qadamma-qadam o'zlashtirib, o'rganadigan "Entry-Python" dasturlash

Bu darslik bolalar uchun mo'ljallangan dasturiy ta'minot bo'yicha ta'lim dasturi bo'lgan "Entry" dasturlash tili orqali Python dasturlash tilini yaxshilab tushunib o'rganish uchun o'zbek tiliga tarjima qilindi.

Bu kitob hamma bepul dasturiy ta'minot ta'limini olishga yordam berish uchun ishlab chiqildi. Ushbu darslikni ishlab chiqishda yaqindan yordam berganlarga o'z minnadorchiligimizni bildiramiz.

Asl nusxa: **엔트리파이선**

Nashriyot: **RoboticsWare**

Hamkorlik: **UFE**(Uzbekistan Foreign Experts group)

O'zbekiston Respublikasi Xalq ta'limi vazirligi

BARKAMOL AVLOD respublika bolalar maktabi

IT PARK



Qo'shimcha yozuvchi : JeongJun Lee

Tekshiruvchi: JeongJun Lee, EuiHo Hong, Safarova Go'zal Ramazonova

Tarjimon: Najimova Altinay

Asl mualliflik huquqi: UFE & RoboticsWare



Copyright © NAVER Connect Foundation & [RoboticsWare](https://roboticsware.com). Some Rights Reserved. Ushbu kitobning mazmuni "Creative Commons" Attribution 4.0 litsenziyasi ostida ishlatiladi.

<https://creativecommons.org/licenses/by-nc-sa/4.0/>

1-Dars. Entry-Pythonni boshlash

1

Matnli(Text) tili nima?

Dasturlash uchun qurilmaga yaqin darajali Mashina(Machine) tili, Assembly tili, C dasturlash tili yoki yuqori darajadagi dasturlash tillari (Python, Java, Javascript) kabi ishlab chiqarilgan maqsadlar yoki foydalanish usuliga har xil tillar qo'llaniladi. Ular orasida sizlar osongina yondashishingiz mumkin bo'lgan Entry va Scratch kabi tillar ta'lim dasturlash tillari deb ataladi. Ta'lim dasturlash tillari klaviatura orqali buyruqlarni to'g'ridan-to'g'ri kiritadigan boshqa dasturlash tillaridan (matnli(text) tillar deb ataladi) farqli o'laroq, ta'lim dasturlash tillari buyruq bloklarini sichqoncha bilan taxlash orqali qo'llaniladi. U ta'lim olish maqsadida ishlab chiqilganligi sababli, yuqori darajadagi dasturlash qiyin, lekin dasturlash sohaga oson kiritish xususiyatiga ega.

Agar siz Entry yoki Scratch orqali blokli tilini o'rgangan bo'lsangiz, endi matnli tilini o'rganishni boshlasangiz-chi? Agar siz blokli tili bilan allaqachon tanish bo'lsangiz, matnli tili avvaliga biroz notanish qiyin bo'lib tuyulishi mumkin. Lekin, tashvishlanmang! Entry orqali siz matnli tilidagi Python tilini o'rganishingiz mumkin.

Hozirdan boshlab Entry-Pythonni birga boshlaymizmi?

2

Keling, blokli tili va matnli tilning o'rtasidagi farqni ko'rib chiqaylik.



Blokli va matnli tillarining orasida quyidagicha xususiyatlari bor. Birgalikda ko'rib chiqamizmi?

	Blokli tili	Matnli tili
Dasturlash usuli	Sichqoncha orqali sudrab olib tashlash (Drag & Drop)	Klaviatura orqali kiritish
Maqsad	Dasturlashga kirish va dasturlash ta'lim	Haqiqiy tijorat dasturi va tizimni ishlab chiqish, amaliy muammolarni hal qilish
Afzalliklar	O'rganish oson	Xohlagan narsalarni cheksiz yaratish
Kamchiliklari	Belgilangan(chegaralangan) doira ichida ishlab chiqish	O'rganish qiyin

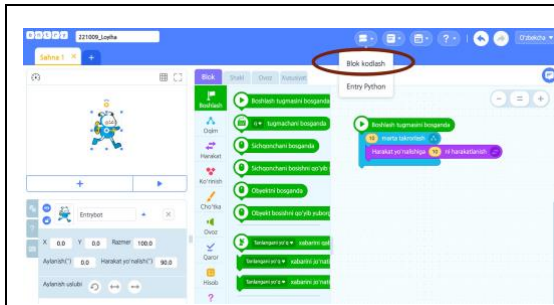
3


Entry-Python qanday til?

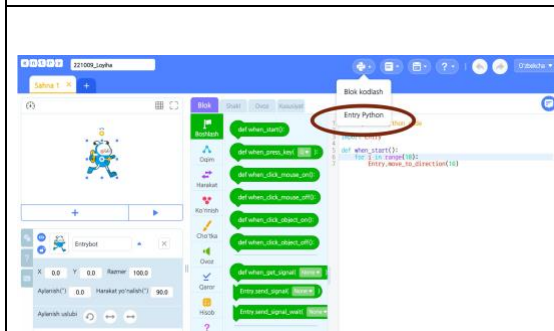
Entryda bloklari asosida dasturlash qilgan kodlarni matnli(text) tillar(Python, C++)ga almashtirish imkoniyatiga ega. Shu paytgacha qilgan loyihalaringizni text tili orqali qanday amalga oshirishni ko'rish qiziqasizmi?

Shuning uchun agar siz Entry-Pythondan foydalanib o'rganishni boshlasangiz, *Pythonning asosiy grammatika va buyruqlarini osondan o'rganishingiz mumkin.*

Avvalo, Entry-Pythonni ishga tushirib ko'ramizmi?



Entryning yuqori qismidagi menyudan  tugmasini bosib Entry-Pythonni tanlaymiz.



Blok kodlar matnli tilga almashtirilgach, biz Entry-Pythonni ishga tushirishga tayyormiz. Odatida blok kodlarni boshlash tugmasi(▶)ni bosib ishga tushirgandek Entry-Python kodlar ham ishga tushirish uchun boshlash tugmasi orqali ishlay boshlaydi.



Qo'shimcha
ma'lumotlar

Rejimni o'tganda doim quyidagi 2ta qatorlarda Python kodlar avtomatik ravishda kiritiladi va uni o'zgartirib bo'lmaydi.

Entrybot's Python code → Joriy obyekt nimaligini bildirish uchun izohlar.

Python tilida izohlarni qoldirishda '#'dan boshlash kerak.

import Entry → Entryning hamma Python funksiyalarini bir joyga yig'ib qo'ygan Entry kutubxonasini chaqirish kod.

Lekin Entry blokli dasturlashga ustunlik qo'yib, asos sifatida foydalanganligi sababli Pythonning barcha standart funksiyalaridan foydalana olmaysiz va Entry o'zining funksiyalardan ko'proq foydalanib dasturlash qilishingiz kerak.

Va siz Python tilida tushunib olmaydigan grammatika tarzda blokli dasturlash qilsangiz, Error xabari chiqadi, o'shanda Entry-Python tili rejimga o'ta olmaysiz. Keling, qanday vaziyatlar bo'lishi mumkinligini ko'rib chiqamiz.

Vaziyat	Chiqadigan error xabarlar
O'zgaruvchi nomida bo'sh joy bo'lgan vaziyatda	Ro'yxatdan o'tgan o'zgaruvchilar orasida nomida bo'sh joy(bo'shliq, probel)ga ega o'zgaruvchi mavjud bo'lsa, rejimni o'zgartirib bo'lmaydi.
Ro'yxat nomida bo'sh joy bo'lgan vaziyatda	Ro'yxatdan o'tgan ro'yxatlar orasida nomida bo'sh joy(bo'shliq, probel)ga ega ro'yxat mavjud bo'lsa, rejimni o'zgartirib bo'lmaydi.
Funksiya nomida bo'sh joy bo'lgan vaziyatda	Ro'yxatdan o'tgan funksiyalar orasida nomida bo'sh joy(bo'shliq, probel)ga ega funksiya mavjud bo'lsa, rejimni o'zgartirib bo'lmaydi.
Funksiyani yaratishda [nom] bloki 2 martadan ziyod ishlatilgan vaziyatda	Ro'yxatdan o'tgan funksiyalar orasida funksiya nomida [nom] bloki 2 marta ziyod kiritilgan bo'lsa, rejimni o'zgartirib bo'lmaydi
Funksiya yaratishda [nom] bloki [satrli/sonli qiymat] yoki [mantiqiy qiymati] blokidan keyin yozilsa	Funksiya yaratishda [nom] bloki [satrli/sonli qiymat] yoki [mantiqiy qiymati] blokidan keyin yozilsa, rejimni o'zgartirib bo'lmaydi
Funksiyani yaratish yoki tahrirlash (o'zgartirish)da	Funksiyani yaratish yoki tahrirlash(o'zgartirish)da rejimni o'zgartirib bo'lmaydi.

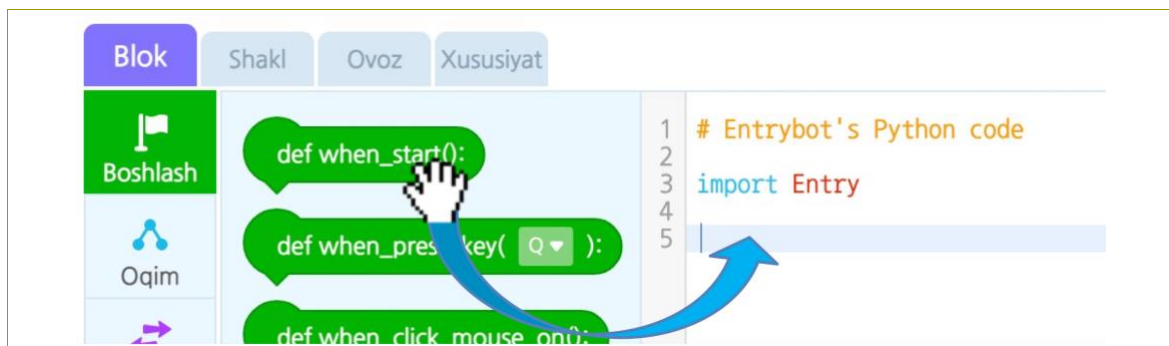
Blokli rejimdan matnli rejimga o'tishda chiqqan har bir errorga qarab shu errornining ma'nosini to'g'ri tushunib hal qilsangiz keyin muammosiz rejimga o'ta olasiz.

4

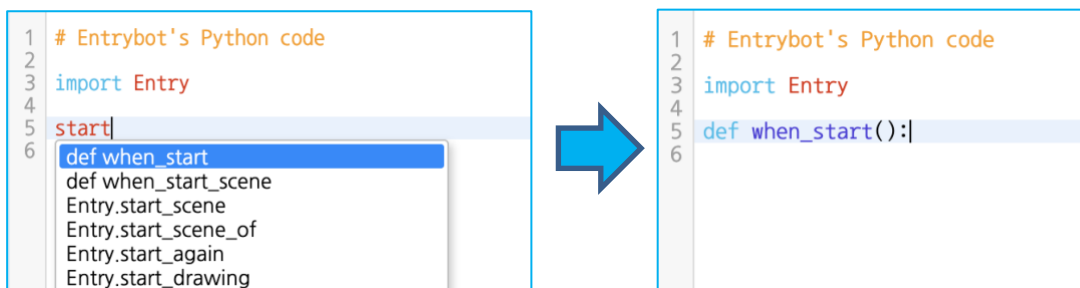
Entry-Python qanday qilib foydalanishni batafsil ko'rib chiqamiz.

Text(Matnli) tilini o'rganishda qiyin bo'lgan buyruqlarni kiritishni Entry-Pythonda ikki xil usulda oson amalga oshirish mumkin.

Avvalo, quyidagi rasmda ko'rsatilgandek, blok rejimida foydalangan usul kabi sichqonchadan buyruqlarni sudrab olib tashlash (Drag & Drop) orqali kodlarni yozish mumkin. Usha paytida buyruqlar kursor turgan joyga borib tushishligi sababli kursorni har doim buyruq kiritilmoqchi bo'lgan joyga qo'yishingiz kerak.



Buyruqlarni sudrab olib tashlash usuli qulay bo'lishi mumkin, lekin bu text tilining xususiyatlariga ega bo'lmagan dasturlashga aylanadi. Shu sababdan ham ikkinchi usulni tavsiya qilamiz. Entry bloklarini o'ylab hayolingizga kelgan kalit so'zlarni kiritsangiz tavsiya qilingan buyruqlar orasidan tegishlisini tanlab olsangiz bo'ladi.



5

Dasturlashda yordam beradigan klaviaturalar kombinatsiyalarini o'rganamiz.

Text tili xususiyatlaridan kelib chiqqan holda buyruqlar klaviatura orqali kiritiladi. Shu sababli klaviaturalar kombinatsiyalarini bilib olsangiz bimalol qulay tarzda buyruqlarni kiritishingiz mumkin. Quyidagi buyruqlarni ko'rib chiqing va dasturlashingizda har bittasiga atalgan klaviatura kombinatsiyasini sinab ko'ring.

Windows uchun klaviatura kombinatsiyalari

Funksiya	Klaviaturalar kombinatsiyasi
4 ta probelni kiritish / oldirish	Tab, Shift + Tab
Bloklar / Shakl / Ovoz / Xususiyatlar tasmali panelini tanlash	Alt + 1, Alt + 2, Alt + 3, Alt + 4
Avvalgi / Keyingi obyektini tanlash	Alt + [, Alt +]
Blokli kodlash / Entry-Python rejimiga o'zgartirish	Ctrl + [, Ctrl +]
Ishga tushirish	Ctrl + R

Mac OS uchun klaviaturalar kombinatsiyalari

Funksiya	Klaviaturalar kombinatsiyasi
4 ta probelni kiritish / oldirish	Cmd + [, Cmd +], Tab, Shift + Tab
Bloklar / Shakl / Ovoz / Xususiyatlar tasmali panelini tanlash	Alt + 1, Alt + 2, Alt + 3, Alt + 4
Avvalgi / Keyingi obyektini tanlash	Alt + [, Alt +]
Blokli kodlash / Entry-Python rejimiga o'zgartirish	Ctrl + [, Ctrl +]
Ishga tushirish	Ctrl + R

2-Dars. Kompyuterda ma'lumotni ko'rsatish usuli

1

Kompyuter ma'lumotlarni qanday ko'rsatadi?

Kompyuterlar turli xil qurilmalardan tashkil etilgan. Ular orasida natijani ko'rsatadigan qurilmani chiqarish qurilmasi deb ataladi. **Chiqarish(Output)** hozirgi sharoitni ifodalashdek turli vaziyatlarda qo'llaniladi. Odatda foydalanuvchilar murojaat qilganidan hal qilingan natijasini chiqarib ko'rsatishda ishlatiladi.

Kundalik hayotimizda ishlab chiqarishning ko'plab misollarini ko'rishimiz mumkin. Smartfonlar, televizorlar, printerlar va kolonka kabi turli xil qurilmalar mavjud va biz ushbu qurilmalardan tinglash yoki tomosha qilish orqali ma'lumot olish uchun foydalanamiz. Chiqarish o'zi dasturlashda hal qilgan natijalarni ko'rsatish yoki oraliq natijalarni ko'rib chiqishda ko'p foydalanadi.

Keling, chiqarish Entry-Pythonda qanday ishlashini birgalikda ko'rib chiqaylik!



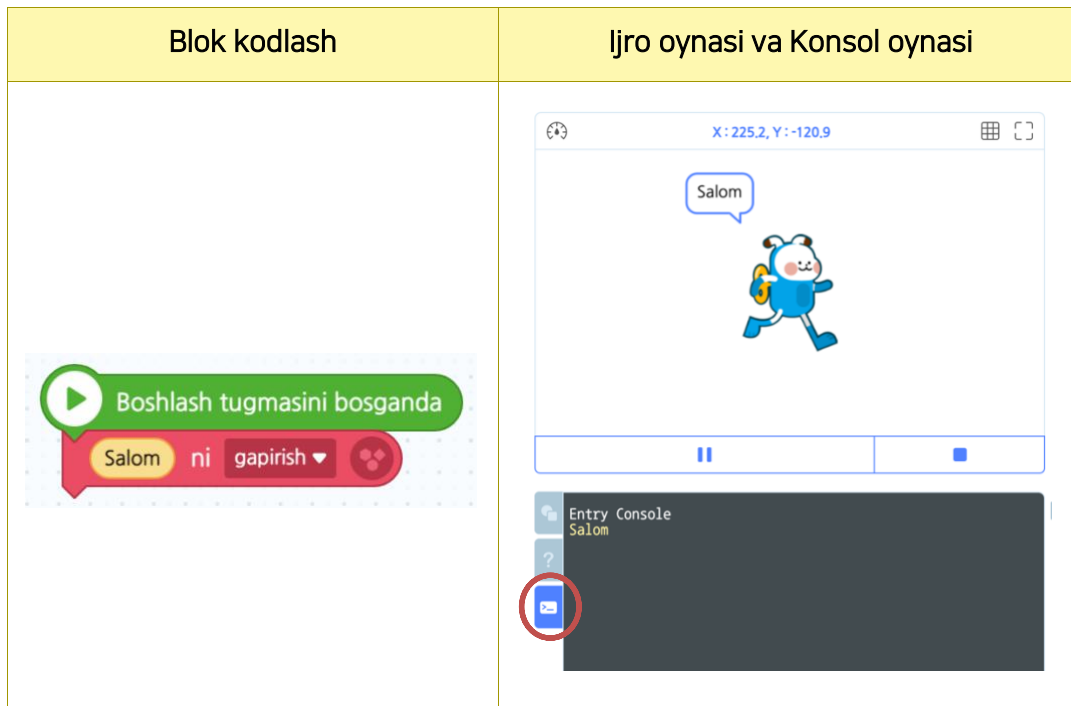
2

Keling, Entry-Pythondagi chiqarish(Output)ni o'rganamiz.

Entryning eng katta xususiyati shundaki, u turli yo'llar bilan natijalarni chiqarib berishi mumkin. Siz ijro oynasida turli obyektlarni ko'chirishingiz va gaplar yoki so'zlarni aytirishingiz mumkin.

Text tillarida ham albatta, rasm yoki grafikdan foydalanib natijalarni ifodalash mumkin. Biroq, u asosan konsol oynasida matnli ko'rinishida ko'rsatiladi.

Entryda dasturlashning natijasini ijro oynasida chiqarish mumkin. Shuningdek, natijalarni konsol oynasida chiqarib tekshirishingiz mumkin.



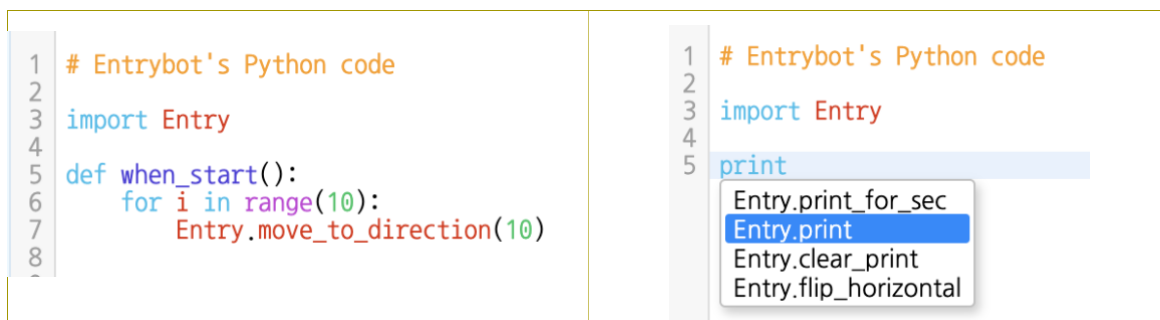
3

Dasturlash orqali chiqarish haqida bilib olamiz.

Mabodo "Hello World!" gapini qayerdadir ko'rmaganmisiz? Bu barcha dasturlashni o'rganishda eng boshida tuzadigan eng kichik dasturdan ko'rgan bo'lsa kerak. Bu dasturning maqsadi u gapni konsolga chiqarishdir. U gapi dasturlash olamiga xush kelibsiz kabi ma'noda desak ham bo'ladi Entry-Pythonda ham "Hello World!" gapni konsolga chiqarib ko'ramiz.

Entry-Python ham boshqa tillar kabi chiqarish buyrug'i(funksiyasi)ga ega. Chiqarish so'zini eshitganingizda birinchi bo'lib xayolingizga nima keladi? To'g'ri, hujjatlarni chiqarish uchun kabi ishlar printerdan foydalaniladi.

Inglizcha "Print" so'zi chop etish, chiqarishni anglatadi, shuning uchun ko'plab dasturlash tillari natijani chiqarib berishda print so'ziga o'xshash so'zdan foydalanadi. Avvalo, blok kodlash rejimidan Entry-Python rejimiga o'tamiz. Quyidagi chap tarafida ko'rsatilgan allaqachon yozilgan hamma kodlarni o'chirib o'riniga **"print"**ni yozib kiriting.



Printni yozsangiz, o'ng tarafida ko'rsatilishdek, **print**ga aloqador turli buyruqlar(funksiyalar)ni ko'rishingiz mumkin. Biz ikkinchi **Entry.print** yozuvini tanlasangiz avtomatik **Entry.print()** deb yoziladi. U ma'nosi Entry deb nomlangan kutubxonasi(Funksiyalarini bir joyga yig'ib qo'ygan to'plami) ichidagi **print()** funksiyani chaqirish bo'ladi. **print()** funksiyasi Entry kutubxonasi ichida mavjud bo'lganligi sababdan uni chaqirish uchun funksiyaning nomi oldida Entry bilan **nuqta(.)ni qo'shib Entry.** deb yozib chaqirish shart. Aslida nuqtasi(.) dasturlash tilida operatorlardan bir deb hisoblanadi. Odatda kutubxona yoki obyekt ichdagi funksiyalarni chaqirish maqsadli operator bo'ladi.

Agar funksiya o'zi nimaligini bilmasangiz oldin chop etilgan <<Qadamma-qadam o'zlashtirib o'rganadigan Entry dasturlash>> o'rtacha darajali kitobning 5-bobini o'qib tushunib olishingiz mumkin.



Qo'shimcha
ma'lumotlar

Funksiya o'zi nimaligi soddalashtirib aytilsa, u asosiy kod bo'lagiga tegishli bo'lgan kichik mustaqil kod bo'lagi deb hisoblanadi. Unga mustaqil degani da, uning ikkita ma'nosi bor. U alohida mavjud bo'lib, chaqirilgandagina ishlaydi va mustaqil ishlaydi.

Demak, bir dastur o'zi ko'p funksiyalardan foydalanib yaratiladi. Uning funksiyalari dasturchi o'zi yaratgan funksiyalar va Entry kutubxonasi ichida qo'yilgan boshqa dasturchidan yaratilgan oldin tayyor bo'lgan funksiyalardan iborat.

Funksiyasi odatda foydalanuvchidan topshirilgan qiymatlardan foydalanib o'zi ishni bajaradi. U qiymatlarini funksiyaning nomidan keyin turgan qavs() ichkariga yozib funksiyaga topshiradi. Shuning uchun funksiyaning nomi **print**dan keyin turgan qavs() ichida, ya'ni u funksiyasi chiqarish ish bajarishda kerak bo'lgan qiymati demak biz chiqartirmoqchi bo'lgan gapni <<Hello World!>>ni yozamiz. Dasturlash dunyoda mana shundaydek funksiyaga topshiriladigan qiymatlarni **parametr** deb ataladi.

Keling, endi <<Hello World!>> natijasini ko'raylik! Dasturlashganini ijro etish uchun boshlash tugmasi(▶) ni bosing!

Voyyyy! Pastki o'ng burchakdagi error xabari bilan hech narsa ishlamaydi. Muammo nimada? Pastki o'ng burchakdagi sintaksis error(grammatik error) xabari bilan kodda error borligini bildirgandi.

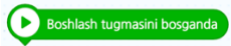
```
1 # Entrybot's Python code
2
3 import Entry
4
5 Entry.print(Hello World!)
6
7
```

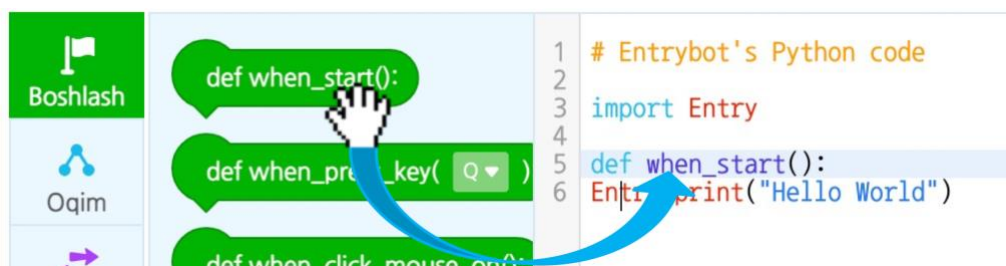
Sintaksis Error

[Token] : Sintaksis error. o'zgaruvchi nomi o'riniga ", " yoki ")" kerak. (line 3)

Ko'pgina text tillarida turli xil qiymatlar va buyruqlardan foydalaniladi. Lekin u qiymatlarning turi raqam(sonli)mi? yoki matn(satrl)mi? Oldin turi nimaligini bildirib foydalanish kerak. Shunday qilib, <<Hello World!>> gapini chiqarish uchun siz unga bu qiymati matn(satrl)ligini oldin bildirishingiz kerak. Shuning uchun **bittalik (' ') yoki ikkitalik (" ") qo'shtirnoq ichida** u gap yoki matn(satr)ni yozishingiz kerak.

Hozir <<Hello World!>>ni bir vaqtning o'zida ijro oynasini va konsolni ochgan holda va yozilgan kodni ishga tushiramiz. Qanday natija chiqayapti? Oldingidek error yo'q. Lekin hech nima bajarmagandek farqsiz ko'rinadi.

Entry blokli dasturlashda tuzgan kodlarni ishga tushirish uchun  blokini foydalanish shart edi. Entry-Pythonda ham *"Boshlash tugmasi(▶)ni bosganda"*dek kodlarni ishga tushirish uchun bir xil bo'lish kerakmi? deb taxmin qilish mumkin. Bloklar kategoriyasida Boshlashni tanlab birinchi blok(**def when_start()** :)ni sudrab olib tashlash (Drag & Drop) orqali Entry.print("Hello World!") kodining yuqorisiga qo'shamiz. Quyida ko'rsatilgandek ko'rinadi.



Uni yana ishga tushirib ko'raylik-chi? Yana ishlamay qoladi. Dasturlashda buyruqlardan to'g'ri foydalanish yoki tilning sintaksis(grammatika) qoidasiga aniq rioya qilmasak, kompyuter tushunmaydi.

Mana bu muammoni to'g'ri tushunib hal qilish uchun endi Python tilida funktsiyani qanday yaratish va qanday chaqirib foydalanishni o'rganish kerak vaqt keldi. Funktsiyani yaratish sintaksis qoidadagidek bo'ladi. Bosh qismida buni ishlatmoqchi bo'lgan foydalanuvchiga qaysi nomidan chaqirish va chaqirishda topshirish kerakli qiymatlar nimaligini bildiradi. Tana qismida esa chaqirilganda funktsiya o'zi nima ishlarni qilish haqida buyruqlarni ketma-ketlik bilan yoziladi..

Sintaksisi	Misol kodlari
<pre>def funksiya_nomi(parametrlar) :</pre> <p>[4ta probel] bajaradigan buyruqlarning to'plami</p>	<pre>def when_start() :</pre> <p> Entry.print("Hello World")</p>

[Funktsiya yaratish qoydalar]

- ✓ Dasturmizda yozilgan qismi "Funktsiya"ligini bildirish uchun def(definition) deb nomlangan e'lon qiluvchi kalit so'zdan boshlash kerak.
- ✓ Funktsiyaning nomini qaror qilishda ma'noli bo'lib qo'yish muhim. Nomi orqali nima ish bajaradigan funktsiyaligini taxmin qila oladigan darajali bo'lsa yaxshi.

- ✓ Agar funksiyaning foydalanuvchidan topshirilib funksiya o'zi ichida foydalanish kerak bo'lgan qiymatlar(parametrlar) bor bo'lsa, qavs bilan yozish kerak. Yo'q bo'lsa faqat qavsni qoldirsa bo'ladi.
- ✓ Qavsdan keyin **:** (**Ikki nuqta belgisi**)ni qoldirib e'lon qilingan funksiyaning bosh qismi qayergachaligini bildirish kerak.
- ✓ Funksiyani chaqirilganda bajaradigan amallar bo'yicha buyruqlarni tartib bilan yozish kerak. Lekin u buyruqlarning qaysi qismi funksiya tegishli ekanligini aniqlashtirish uchun har bir buyruqning oldida 4 marta probel bosib bo'sh joylarni qoldirish yoki 1 marta Tab kalitni bosib birdaniga 4ta bo'sh joylarni qoldirish mumkin.



Qo'shimcha
ma'lumotlar

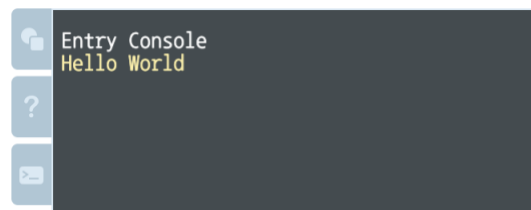
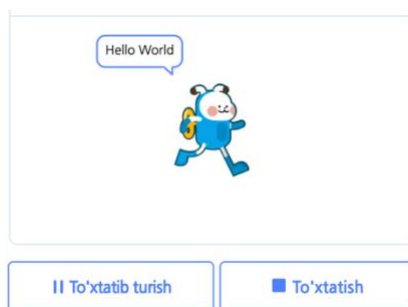
Aslida bo'sh joylar qoldirishda 1ta yoki 2ta bo'sh joylar qoldirish ham mumkin. Muhimi Python tilida bo'sh joylar orqali har bir kodlar bo'lagi(misol uchun funksiya)ning bosh va oxirini ajratib olish maqsadga muvofiq, ya'ni har bir qatordagi buyruqlari qaysi kodlar bo'lagiga tegishligini bildirish maqsadda ishlatiladi. Lekin Python tilining standarti bo'yicha 4 marta bo'sh joylarni tavsiya qilinadi.

Endi mana bu qoidalarga saqlab oldin ishlamagan funksiyamizni to'g'rilasak shunday bo'ladi.

```
1 # Entrybot's Python code
2
3 import Entry
4
5 def when_start():
6     Entry.print("Hello World")
```

```
1 # Entrybot's Python code
2
3 import Entry
4
5 def when_start():
6     Entry.print("Hello World")
```

Hozirgacha yozgan kodlarning ma'nosini aniqlab ko'rsak **when_start** deb nomlangan funksiyasini yaratdik. Lekin hozircha faqatgina bir qatorli buyruqqa ega bo'lganda ham u xatosiz mukammal funksiyadir. Foydalanuvchidan funksiyasi chaqirilsa "Hello World" satrni konsolga chiqaradi. Uni yana ishga tushirsa endi to'g'ri ishlaydi.



Sizga shunday savol tug'ilishi mumkin. Biz faqat mana bu **when_start()** funksiyasini yaratdik, lekin uni chaqirib foydalanadigan hech boshqa qo'shimcha foydalanuvchi kodini

yozmaganmiz. Biz yozgan `when_start()` funksiyasini kim chaqirgan? Chunki biz oldin funksiyaning xususiyatida o'rgangandek yaratilgan funksiyalar boshqa buyruqdan u funksiyaning nomi orqali chaqirilib foydalaniladi. Misol uchun hozir biz `when_start()` funksiyamizning ichida **`Entry.print()`** deb kodni yozib `print()` funksiyasini nomi bilan chaqirib foydalanayapmiz.

Endi kim chaqirganligi haqida bilib olamiz. mana bu `when_start()` funksiyasi oddiy funksiyalardan foydalanish emas, biz *Entry interfeysidagi "Boshlash" tugmasini bosgan paytda Entry dastur o'zi mana bu `when_start()` funksiyasini avtomatik chaqirib bergandir.* Dasturlash dunyosida shunday dastur o'zidan yoki operatsion tizimi o'zidan avtomatik chaqiriladigan funksiyalari **"callback" funksiyasi** deb ataladi.

4

Entry-Pythondagi boshqa chiqarish funksiyalarini o'rganamiz.

(1) Chiqarish uchun ishlatiladigan har xil `print()` funksiyalarni ko'rib chiqamiz.

Funksiyalar	Funksiyaning tavsifi
<code>Entry.print("Salom!")</code>	Qo'shtirnoq ichidagi topshirilgan satrli qiymati konsol oynasida va ijro etish oynadagi obyekt o'zida chiqarib beriladi.
<code>Entry.print_for_sec("Salom!", 4)</code>	Funksiyaga topshirilgan satr qiymati va sonli qiymati orqali berilgan satrni berilgan soniya davomida konsol oynasi va ijro oynasidagi obyekt o'zida ko'rsatiladi.
<code>Entry.clear_print()</code>	Obyektida chiqarilgan qiymatlarni o'chiradi. Biroq, u konsol oynasida chiqarilgan qiymatlarni o'chirmaydi.

5


Chiqarish funksiyalaridan foydalangan holda sizga qo'shimcha vazifalarni sizga topshiramiz.

(1) Oldindek `Entry.print()` funksiya orqali chiqarilganda ijro oynasi va konsol oynasi natijalari nima uchun farq qilishini o'ylab ko'ring.

1-Vazifa

<pre> 1 # Entrybot's Python code 2 3 import Entry 4 5 def when_start(): 6 Entry.print("Hello World?") 7 Entry.print("Men Entrybotman.") 8 Entry.print("Tanishganimdan xursandman.") </pre>	<div style="border: 1px solid #ccc; padding: 10px; margin-bottom: 10px;"> <p>Tanishganimdan xursandman.</p>  </div> <div style="display: flex; justify-content: space-around;"> <div style="border: 1px solid #ccc; padding: 5px; background-color: #f0f0f0;">II To'xtatib turish</div> <div style="border: 1px solid #ccc; padding: 5px; background-color: #f0f0f0;">■ To'xtatish</div> </div> <div style="border: 1px solid #ccc; padding: 10px; background-color: #333; color: #fff;"> <p>Entry Console</p> <p>Hello World?</p> <p>Men Entrybotman.</p> <p>Tanishganimdan xursandman.</p> </div>
--	--

(2) Entry.clear_print() funksiya orqali nega shunday natijani chiqarayotganini o'ylab ko'ring. Dasturda ketma-ketlik bo'yicha bajarishni o'ylasangiz javobni topish oson bo'ladi.

2-Vazifa	
<pre> 1 # Entrybot's Python code 2 3 import Entry 4 5 def when_start(): 6 Entry.print("Hello World?") 7 Entry.print("Men Entrybotman.") 8 Entry.print("Tanishganimdan xursandman.") 9 Entry.clear_print() </pre>	<div style="border: 1px solid #ccc; padding: 10px; margin-bottom: 10px;">  </div> <div style="display: flex; justify-content: space-around;"> <div style="border: 1px solid #ccc; padding: 5px; background-color: #f0f0f0;">II To'xtatib turish</div> <div style="border: 1px solid #ccc; padding: 5px; background-color: #f0f0f0;">■ To'xtatish</div> </div> <div style="border: 1px solid #ccc; padding: 10px; background-color: #333; color: #fff;"> <p>Entry Console</p> <p>Hello World?</p> <p>Men Entrybotman.</p> <p>Tanishganimdan xursandman.</p> </div>

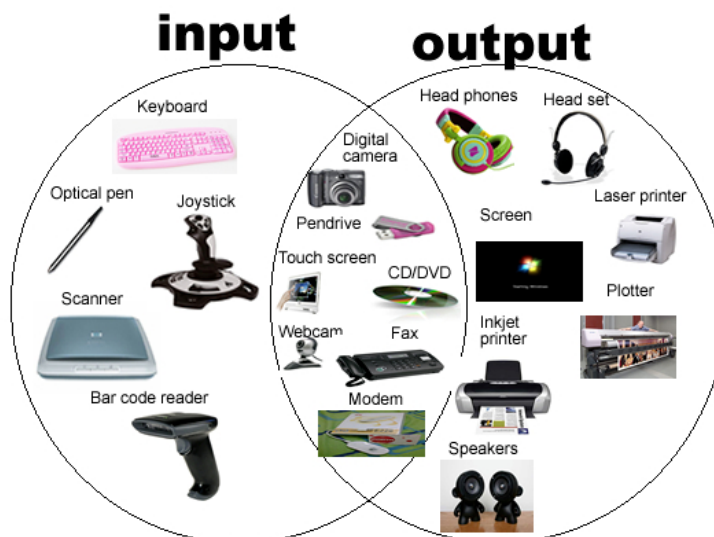
3-Dars. Muammoni hal qilmoqchi bo'lsangiz ma'lumotni kiritishdan boshlang!

1

Kompyuter bilan suhbat, kiritish/chiqarish nima?

Kiritish/chiqarish kompyuterning eng asosiy axborotni qayta ishlash jarayonidir. Klaviatura, sichqoncha, kamera, joystick kabi turli interfeyslar orqali kompyuterga axborot uzatish yoki kiritish(input) deb, oxirgi marta ko'rib chiqqan monitor yoki printer orqali ma'lumotlarni ko'rsatish esa chiqarish(output) deb ataladi.

Kundalik hayotimizda muayyan muammoni hal qilish uchun ma'lumotlarni kiritib, muammoni hal qilish jarayonidan so'ng oxirgi natijani chiqarish yoki ishlab chiqish paytda oraliq natijalarni tez-tez tekshirishda kiritish/chiqarishdan foydalanamiz.

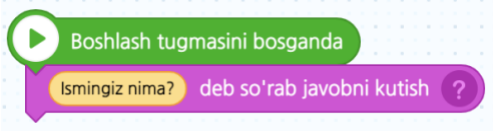
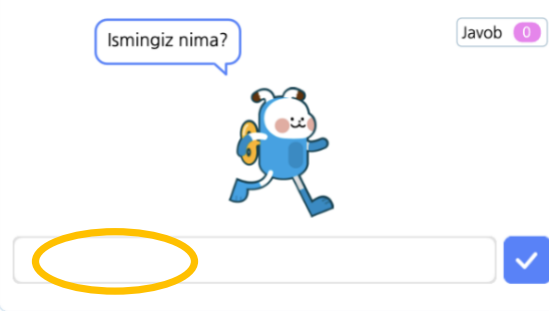
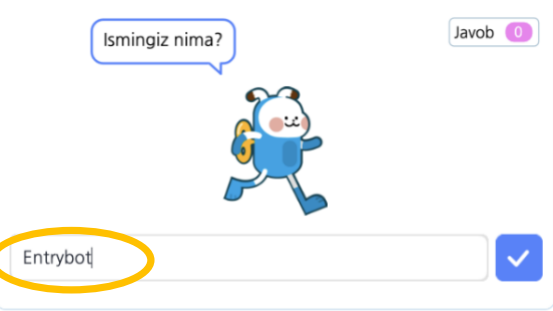



2

Klaviaturada yozish orqali ma'lumotni kompyuterga kiritish

O'tgan safar "Hello World" dek oldin tayyor bo'lgan satrni chiqarish qilib ko'rdik. Keling, bu safar foydalanuvchilardan olgan o'zi xohish so'z yoki gapni chiqarishga harakat qilaylik.

Chiqarish **print()** funksiya bo'lsa, kiritish uchun funksiyaning nomini taxmin qilib ko'raylik. Entry-Pythonda kiritish funksiyasi **input()** bo'ladi. Odatda funksiyaning nomidan chaqirib foydalanish mumkin. Lekin u funksiya Entry kutubxonasi ichida turganligi sababli doim **Entry.** bilan chaqirish mumkin bo'lib demak **Entry.input()** orqali foydalanamiz. **Input()** funksiyada bitta parametr(funksiya ish bajarishda kerakli qiymati)ni talab qiladi. Foydalanuvchilarga nima ma'lumotni kiritish kerakligini bildirish uchun satrli qiymati bilan chaqirish kerak.

Blokli kod	Entry-Python
	<pre> 1 # Entrybot's Python code 2 3 import Entry 4 5 def when_start(): 6 Entry.input("Ismingiz nima?") </pre>
Kiritishni kutish	Kiritish jarayon
 <div data-bbox="284 898 571 969"> To'xtatib turish</div> <div data-bbox="582 898 850 969">■ To'xtatish</div> <div data-bbox="284 999 850 1238"> <p>Entry Console</p> <p>Ismingiz nima?</p> <p>Ismingiz nima?</p> </div>	 <div data-bbox="866 898 1153 969"> To'xtatib turish</div> <div data-bbox="1165 898 1433 969">■ To'xtatish</div> <div data-bbox="866 999 1433 1238"> <p>Entry Console</p> <p>Ismingiz nima?</p> <p>Entrybot</p> </div>
Kiritgan natijasi	
 <div data-bbox="619 1619 858 1680"> To'xtatib turish</div> <div data-bbox="869 1619 1098 1680">■ To'xtatish</div> <div data-bbox="619 1704 1098 1904"> <p>Entry Console</p> <p>Ismingiz nima?</p> <p>Entrybot</p> </div>	

Entry ijro oynasiga ismingizni kiritib yoki dasturni qayta ishga tushirib konsol oynasiga ham ismingizni kiritib Enter tugmachani bosib ko'ring. Bir xil natija chiqaradi. Demak

Entryda ijro oynasidek visual interfeys orqali kiritish yoki konsol orqali kiritish ikkita iloji mavjud.

Kiritilgan ismingizni Entry ijro oynasidagi javob maydonida ko'rishingiz mumkin. Entry o'zi biz uchun kiritilgan qiymatni vaqtincha kompyuter xotiraning bir joyi (**javob o'zgaruvchi**)da saqlab turadi. U saqlangan qiymatini olish uchun oldin tayyor bo'lgan funksiya mavjud. **Entry.answer()** funksiya orqali u qiymatini xotiradan olishingiz mumkin.

Dasturlash dunyosida muayyan ma'lumotlarni saqlash uchun kompyuterda yaratishimiz kerak bo'lgan joyni biz nima deb atardik? Bu o'zgaruvchi. Entry kiritilgan qiymatini "javob" deb nomlangan vaqtinchalik o'zgaruvchida saqlaydigan qilib loyihalashtirilgan.

3

Ma'lumotni saqlash uchun o'zgaruvchi yaratib ko'raylik.

Entryda avvaldan tayyor "javob" degan o'zgaruvchi bo'lsa ham bizlar o'zlarimiz xohlagan o'zgaruvchini yaratib ma'lumotlarni saqlab chiqarib ko'ramiz. O'zgaruvchilar uchun quyidagi tavsifni o'qib chiqing.



Qo'shimcha
ma'lumotlar

O'zgaruvchi nima?

Satrlari va raqamli kabi turli xil ma'lumotlarni saqlashi mumkin bo'lgan kompyuterdagi xotira joyi. O'zgaruvchilarga ma'noli nom berilsa, u yerda qanday ma'lumotlar borligini bilish oson bo'lishi mumkin va bitta o'zgaruvchida faqat bitta ma'lumot saqlash mumkin. Agar boshqa ma'lumotlar saqlangan bo'lsa, oldingi ma'lumotlar o'chiriladi.

Endi o'zgaruvchi yarataylik. "Ismingiz nima?" deb savol berganimiz uchun qanday kiritilgan ma'lumot qanday bo'ladi? Albatta, bu ism! Ushbu ma'lumotlarni osongina tushunish va dasturlash uchun "ism" deb nomlangan o'zgaruvchini yaratish qulay. Pythondagi o'zgaruvchilar har doim bajarilishi kerak bo'lgan buyruq oldidan yaratilishi kerak.

```
1 # Entrybot's Python code
2
3 import Entry
4
5 ism = 0
6
7 def when_start():
8     Entry.input("Ismingiz nima?")
```

Agar siz "ism = 0" bilan o'zgaruvchi yaratsangiz, kompyuterga quyidagi buyruqni bergan bo'lasiz.

Avvalo, "ism" degan ma'lumotlarni saqlash mumkin bo'lgan bo'shliqni yaratib, bu bo'shliqqa 0 raqamli qiymatni saqla!

Endi "ism" nomli o'zgaruvchi yaratilgan bo'lsa, biz **Entry.input("Ismingiz nima?")** deb funksiya orqali foydalanuvchilardan kiritilgan qiymatni "ism" nomli o'zgaruvchida saqlashimiz kerak, to'g'rimi? "Javob" deb nomlangan vaqtinchalik saqlanadigan o'zgaruvchidagi qiymatni **Entry.answer()** dan olish mumkin. Shuning uchun, agar biz **"ism = Entry.answer()"** deb kodni yozsak, biz foydalanuvchilardan kiritilgan qiymatni "ism" deb nomlangan o'zgaruvchiga saqlashimiz mumkin.

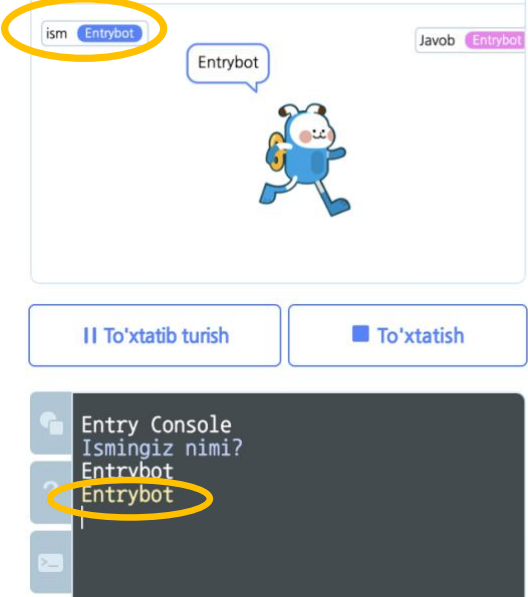
Nihoyat, **Entry.print()** funksiya yordamida natijani chiqarishingiz mumkin. Qanday ma'lumotlarni chiqarishimiz kerak? Albatta, bu "ism" deb nomlangan o'zgaruvchida saqlangan qiymat bo'lishi kerak, to'g'rimi? Shunday qilib, siz quyidagi kabi dasturlashni yakunlashingiz mumkin.

Ishga tushirib kiritilgan ism chiqarilib berilishini tekshiring.

```

1 # Entrybot's Python code
2
3 import Entry
4
5 ism = 0
6
7 def when_start():
8     Entry.input("Ismingiz nima?")
9     ism = Entry.answer()
10    Entry.print(ism)

```



4

Entry-Python funksiyalarni o'rganamiz.

(1) Kiritishda foydalaniladigan funksiyalarni ko'rib chiqamiz.

Funksiya	Funksiyaning tavsifi
<code>Entry.input("salom")</code>	Bu klaviatura yordamida kerakli ma'lumotlarni kiritish imkonini beradi.
<code>Entry.answer()</code>	<code>Entry.input("Salom")</code> ishga tushirilganda klaviatura orqali kiritilgan vaqtinchalik saqlangan qiymatni olish
<code>Entry.answer_view("hide")</code> <code>Entry.answer_view("show")</code>	Ijro oynasida vaqtinchalik "javob" o'zgaruvchisini yashiradi yoki ko'rsatadi.

5

Chiqarish vazifasini hal qilamiz.

(1) Quyidagicha dasturlashdagi muammolarni va hal qilish usullarni yozamiz.

1-Vazifa	
	<pre> 1 # Entrybot's Python code 2 3 import Entry 4 5 ism = 0 6 7 def when_start(): 8 Entry.input("Ismingiz nima?") 9 ism = Entry.answer() 10 Entry.input("Yoshingiz nechida?") 11 ism = Entry.answer() 12 Entry.input("Hobbyingiz nima?") 13 ism = Entry.answer() 14 Entry.print(ism) </pre>
Muammmo-nima?	
Qanday hal qilish mumkin?	<p>* Turli xil o'zgaruvchilar yaratish orqali o'z dasturingizni yarating. Chiqarishda <code>Entry.print_for_sec(o'zgaruvchining nomi, soniya)</code> funksiyadan foydalansa osonroq.</p>

4-Dars. Bir nechta ma'lumotni saqlay olish mumkinmi?

1

Bir nechta ma'lumotlarni bir joyda saqlay olmaymizmi?

O'tgan soatda saqlangan ma'lumotlarni bilib olish oson bo'lgan nom berib, ma'lumotlarni saqlab va ishlatib ko'rdik. Kompyuterda ma'lumotni saqlash uchun joy ajratish muhim dedik. Unda, bu narsa haqda o'ylab ko'raylik. Har kuni qancha ma'lumotdan foydalanishimizni hisoblang. Agar hisoblash juda qiyin bo'lsa, ma'lumotlar turlarini sanab ko'ring.

Kundalik hayotimizda foydalanadigan va saqlaydigan ma'lumotlar miqdori behisob. Va bu ma'lumotlardan to'g'ri foydalanish qobiliyati sizning kelajakdagi jamiyatdagi raqobatbardoshligingizdan biri bo'lishi mumkin. Xo'sh, qanday qilib biz juda ko'p ma'lumotlarni saqlab boshqarishimiz kerak?

Albatta, ma'lumotni saqlash uchun oxirgi marta o'rgangan o'zgaruvchilardan ham foydalanishingiz mumkin, ammo bu safar ko'plab ma'lumotlarni saqlash va tartibga solish uchun yangi joy, demak ro'yxatni ishlatamiz. Keling, Entry-Pythonda ro'yxatni qanday yaratish va undan foydalanishni birgalikda ko'rib chiqamiz..



2

Bir necha ma'lumotlarni saqlaydigan va boshqaradigan ro'yxatlar haqida o'rganamiz.

Oxirgi soatdagi barcha topshiriqlarni hal qildingizmi? O'zgaruvchilar yordamida turli xil ma'lumotlarni kiritishga harakat qilgan edik. Agar siz o'zgaruvchilarga turli xil ma'lumotlarni kiritib va ulardan foydalana olsangiz, menimcha, boshqa usullar ham kerakmi? O'ylashingiz mumkin. Biroq, unga kamchilik bor. Bitta o'zgaruvchida faqat bitta ma'lumot saqlanishi mumkin.


Ko'p ma'lumot bo'lmasa, kerakli darajada ko'p o'zgaruvchilar yaratib ishlatish mumkin. Ammo ma'lumotlar soni 1000, 10000 va undan ko'p bo'lsa-chi? Bir biriga o'xshash ma'lumotlarni bir joyga to'plab va joylashuvini birgalikda ifodalasak, uni ancha samarali boshqarish mumkin bo'ladi. Hozirda sizga kerak bo'lgan narsa **"Ro'yxat(List)"** dir. Ro'yxati ma'lumotlarni saqlanadigan joy bilan bir qatorda bir nechta ma'lumotlarni saqlashi va ulardan foydalanishi mumkin.

ismlar

0	Javlon
1	Baxtiyor
2	Entrybot
3	Durdona

yoshlar

0	12
1	13
2	200
3	14



Entrybot

II To'xtatib turish
■ To'xtatish

```

1 # Entrybot's Python code
2
3 import Entry
4
5 ismlar = ["Javlon", "Baxtiyor", "Entrybot", "Durdona"]
6 yoshlar = [12, 13, 200, 14]
7
8 def when_start():
9     Entry.print(ismlar[2])

```

3

Ro'yxat(List)ni yaratib ma'lumotlarni saqlang.

Ro'yxatlar(List) xuddi o'zgaruvchilar kabi nomlanishi mumkin va ularda juda ko'p ma'lumotlarni saqlashi mumkin. Ma'lumotlarni ro'yxatda saqlash paytida u ma'lumotning turi nimaligi demak matn, raqam yoki boshqa ma'lumot ekanligini ajratib berib kiritish muhimdir.

Biz kompyuterga sonli emas, gap yoki matndek satrli ma'lumotni qanday qilib ajratish mumkinligini o'rgangan edik? To'g'ri. " " yoki ' ' (qo'shtirnoq belgilari) dan foydalandik. Ro'yxatda bir nechta ma'lumotlar saqlanganligi sababli, ma'lumotlarni ajratish uchun, (vergul) dan foydalaning. Endi ro'yxat yaratib matnli ma'lumotlarni kiritamiz.

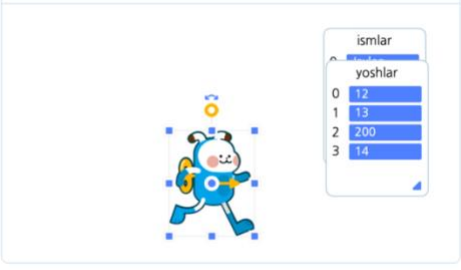
Ro'yxat	O'zgaruvchi
Ro'yxat nomi= [ma`lumot1, ma`lumot2, ma`lumot3]	O'zgaruvchi nomi = ma`lumot
ismlar = ["Javlon", "Baxtiyor", "Entrybot", "Durdona"]	ism = "Entrybot"
O'zgaruvchilar singari, ro'yxat nomlari ham ro'yxatdagi ma'lumotlarni osongina tanib oladigandek qilib nomlanishi kerak. Ro'yxat nomi qanchalik yaxshi bo'lsa, ma'lumotlarni boshqarish shunchalik samarali va qulay bo'lishi mumkin.	

Buyruqni ishga tushirishdan oldin quyidagi tarzda ikkita ro'yxat yaratamiz. Ro'yxat nomiga ma'lumotni saqlamoqchi bo'lgan ro'yxat nomini qo'ying va keyin saqlanadigan ma'lumotni to'rtburchak qavs ichida qo'ying. Agar siz shunday ro'yxat yaratib boshlasangiz, ijro oynasida ikkita ro'yxat paydo bo'ladi,

```

1 # Entrybot's Python code
2
3 import Entry
4
5 ismlar = ["Javlon", "Baxtiyor", "Entrybot", "Durdona"]
6 yoshlar = [12, 13, 200, 14]

```



ismlar

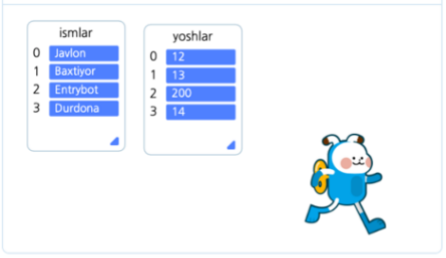
yoshlar

0	12
1	13
2	200
3	14

+ Obyekt qo'shish


▶ Boshlash


Ijro oynasida ko'rsatilgan ikkita ro'yxatni sichqoncha bilan siljitib tegishli joylarga joylashtiring. Ro'yxat faqat yaratilgan ro'yxatdagi ma'lumotlardan hali foydalanilmagan yoki print() funksiya orqali chiqarish qilinmaganligi sababli konsol oynasida hech qanday ma'lumot ko'rsatilmaydi.



|| To'xtatib turish

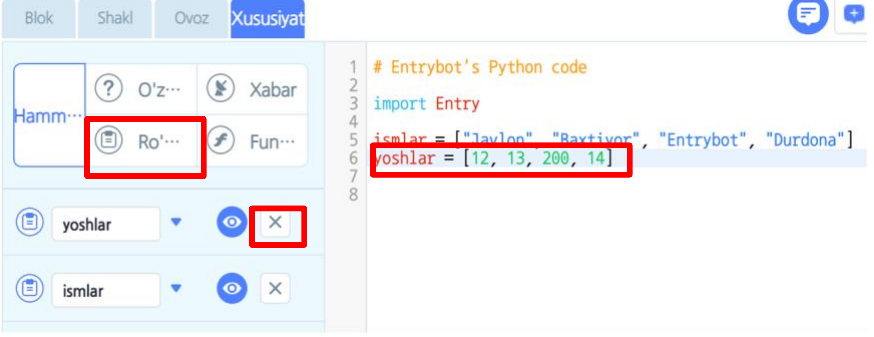
■ To'xtatish





Qo'shimcha ma'lumotlar

Agar noto'g'ri yaratgan yoki keraksiz bo'lgan ro'yxatni o'chirmoqchi bo'lsangiz, faqat kodlarni o'chirish orqali mukammal bo'lmaydi. [Xususiyat] tasmali paneldagi [Ro'yxat]ni bosib kod orqali avtomatik qo'shilgan ro'yxatni ham o'chirishingiz kerak.



Ro'yxat tuzilgach, ro'yxatga qo'shimcha ma'lumotlarni qo'shishga harakat qilaylik. Ro'yxatga ma'lumot qo'shish uchun oldin tayyor bo'lgan funksiya mavjud. Uning nomi append() dir. Lekin uni chaqirish uchun oldingi darsda o'rganganimizdek kutubxona yoki obyekt ichidagi funksiyalarni chaqirishda ishlatadigan nuqta(.) operatoridan foydalanish kerak. Shuning uchun aslida "Ro'yxat(List)" o'zi ham aslida obyekt turlardan bir bo'lgan sababli append() funksiyasini chaqirishda "ismlar" ro'yxatining nomidan keyin nuqtani qo'shib "ismlar." orqali chaqirishingiz kerak.

Albatta, o'tgan darsda o'rganganimizdek Entry-Pythonda buyruqni tashabbus qilish uchun birinchi navbatda **def_when_start()**: deb nomlangan funksiyani yaratishi kerak, to'g'rimi? Quyidagi buyruqni kiriting va "ismlar" deb nomlab oldin e'lon qilingan ro'yxatiga qanday qo'shimcha ma'lumotni qo'shib saqlanganligini ko'ring.

```

1 # Entrybot's Python code
2
3 import Entry
4
5 ismlar = ["Javlon", "Baxtiyor", "Entrybot", "Durdona"]
6 yoshlar = [12, 13, 200, 14]
7
8 def when_start():
9     ismlar.append("Atif")
10    yoshlar.append(16)

```

Keling, endi ma'lumotlarni boshqarish turli xil usullarni o'rganayilik.

4

Ro'yxatdagi ma'lumotlardan turli yo'llar bilan foydalanamiz.

Biz ro'yxatga ma'lumot qo'shib ko'rdik, keling, endi ma'lumotlarni o'chirib va o'zgartirib va uni chiqarish qilib ko'ramiz. Ro'yxatdagi eng muhim narsa shundaki, u turli xil ma'lumotlarni saqlash mumkin va ularni ma'lumotlarning joylashuvi orqali boshqarish mumkin. Agar siz ro'yxatga diqqat bilan qarasangiz, u quyidagicha tuzilganligini ko'rasiz:

Ro'yxatning joylashuvi

Ro'yxatning nomi

Ro'yxat(List)dagi ma'lumotlarning joylashuvi raqam bilan ifodalanadi. Va diqqat bilan qarasangiz raqam 0 dan boshlanadi.

Avvalo, **pop()** funksiyasi yordamida ma'lumotni oldirib o'chirishga harakat qilamiz. Oldinda yozilgan buyruqdan keyin **wait_for_sec()** funksiyani qo'shamiz, funksiyasini qiymat(parametr 3 bilan chaqirgan sababli keyingi buyruq bajarilishidan oldin 3 soniya dam olishi(kutishi)ga imkon beradi. Keyin 2-rasmdek "pop(2)" funksiyasi orqali ro'yxatning 2-joylashuvda turgan "Entrybot" qiymatini olib o'chirishga harakat qiladi. **Pop()** funksiyasini ishlatishda parametrda ro'yxatdagi ma'lumot joylashuv qiymati bilan foydalanishingiz kerak.

```

1 # Entrybot's Python code
2
3 import Entry
4
5 ismlar = ["Javlon", "Baxtiyor", "Entrybot", "Durdona"]
6 yoshlar = [12, 13, 200, 14]
7
8 def when_start():
9     ismlar.append("Atif")
10    yoshlar.append(16)
11    Entry.wait_for_sec(3)

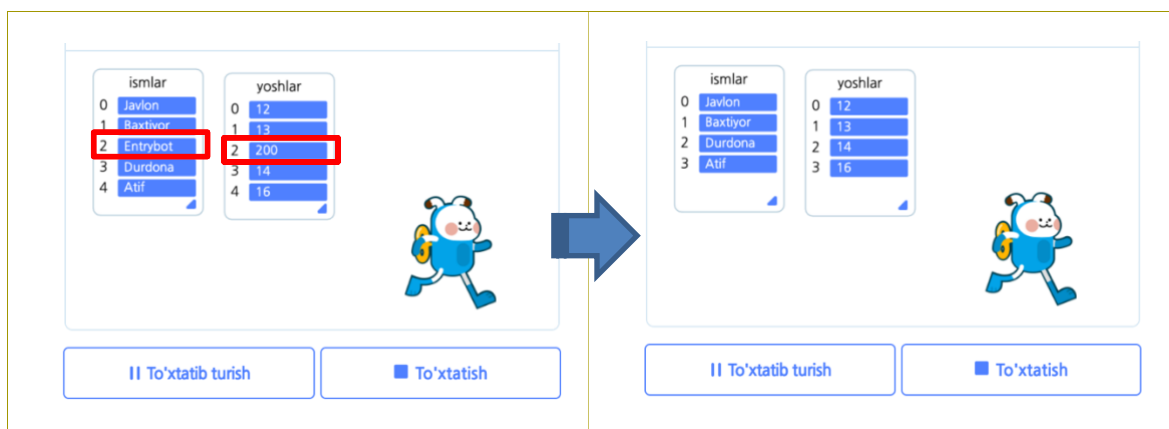
```

```

1 # Entrybot's Python code
2
3 import Entry
4
5 ismlar = ["Javlon", "Baxtiyor", "Entrybot", "Durdona"]
6 yoshlar = [12, 13, 200, 14]
7
8 def when_start():
9     ismlar.append("Atif")
10    yoshlar.append(16)
11    Entry.wait_for_sec(3)
12    ismlar.pop(2)
13    yoshlar.pop(2)

```

Kodlarni ishga tushiring va nima o'zgarishini ko'ring. Birinchidan, ro'yxatlarda ma'lumotlari saqlanganidan so'ng, "ismlar" ro'yxatda 2-joylashuvdagi "Entrybot" olib o'chiriladi va qolgan ma'lumotlar bir vaqtning o'zida bir pog'ona yuqoriga ko'tarilishini ko'rishingiz mumkin.



append() funksiyasi ro'yxatning oxiriga ma'lumot qo'shsa, ro'yxatning o'rtasiga qo'shishning biron bir usuli bormikan? Agar siz insert() funksiyasidan ikkita parametrlar(ma'lumot joylashuvi va ma'lumot) bilan foydalansangiz, ma'lumotni to'g'ridan-to'g'ri o'sha joyga kiritishingiz mumkin. Kiritishda o'tgan darsda o'rgangan foydalanuvchilardan kiritish/chiqarish(input/output)dan foydalanib ko'ramiz. Quyida ko'rsatilgandek qo'shimcha kodlarni kiriting va natijalarni tekshiring.


```

1 # Entrybot's Python code
2
3 import Entry
4
5 ismlar = ["Javlon", "Baxtiyor", "Entrybot", "Durdona"]
6 yoshlar = [12, 13, 200, 14]
7
8 def when_start():
9     Entry.input("Ismingiz nima?")
10    ismlar.insert(1, Entry.answer())
11    Entry.input("Yoshingiz nechida?")
12    yoshlar.insert(1, Entry.answer())

```



Bundan tashqari, ro'yxatlar bilan bog'liq turli xil funksiyalar mavjud. Keling, quyidagi jadvalni ko'rib chiqamiz va uni tartibga solib ulardan foydalangan kodlarni yozib ko'ramiz.

Funksiya yoki Buyruq	Funksiya yoki Buyruqning tavsifi
Ro'yxat nomi.append(ma`lumot)	Ro'yxatning oxiriga ma`lumotni qo'shadi.
Ro'yxat nomi.pop(joylashuv)	Ro'yxatning muayyan joyidagi ma`lumotni o'chiradi.
Ro'yxat nomi.insert(joylashuv, ma`lumot)	Ro'yxatning muayyan joyiga ma`lumotni kiritadi.
Ro'yxat nomi[joylashuv] = ma`lumot	Ro'yxatning muayyan joyidagi ma`lumotni o'zgartiradi.
Ro'yxat nomi[joylashuv]	Ro'yxatdagi muayyan joylashuvdagi ma`lumot qiymatiga ega bo'ladi.
len(Ro'yxat nomi) <i>Misol)</i> len(ismlar)	Ro'yxatdagi saqlangan ma`lumotlar son qiymatiga ega bo'ladi
Ma`lumot in Ro'yxat nomi <i>Misol)</i> "Entrybo" in ismlar	Ma`lumot ro'yxatda mavjudligini tekshiradi.

5 Chiqarish topshirig`ini bajaramiz.

(1) Dasturda ro'yxatlardagi 3-ism va 3-yoshni almashtirish uchun **pop()** va **insert()** funksiyalar orqali emas, ro'yxatning joylashuvidan to'g'ridan-to'g'ri ma'lumotlarni almashtirib keyin to'g'ri almashtirilganligini tekshirish uchun har ma'lumotni tartib bilan 2 soniya davomida chiqarib ko'ring.

1-Vazifa	
<pre> 1 # Entrybot's Python code 2 3 import Entry 4 5 ismlar = ["Javlon", "Baxtiyor", "Entrybot", "Durdona"] 6 yoshlar = [12, 13, 200, 14] 7 8 def when_start(): 9 ismlar[2] = "Alisher" 10 yoshlar[2] = 500 </pre>	
Muammom nima?	
Qanday hal qilish mumkin?	<p>* Ro'yxatdagi ma'lumotlarni to'g'ridan to'g'ri boshqarish uchun Ro'yxat nomi[joylashuv] = ma'lumot buyruqdan foydalansa osonroq.</p>

(2) Foydalanuvchilar bilan interaktiv dasturni yasab ko'ramiz. "Nechinch ismni ko'rish xohlaysizmi?" deb foydalanuvchidan so'rab u kiritgan songa qarab ro'yxatdagi ma'lumotni chiqaradigan qidirish dasturini yasab ko'ramiz. Keyin yosh ro'yxatidan yoshni topishga ham xuddi shunday o'zingizning dasturingizni yasab ko'ring.

2-Vazifa	
<pre> 1 # Entrybot's Python code 2 3 import Entry 4 5 ismlar = ["Javlon", "Baxtiyor", "Entrybot", "Durdona"] 6 yoshlar = [12, 13, 200, 14] 7 8 def when_start(): 9 Entry.input("Nechinch ismni ko'rish xohlaysiz?") 10 Entry.print_for_sec(ismlar[Entry.answer() - 1], 2) </pre>	

ismlar

0

Javlon

1

Baxtiyor

2

Entrybot

3

Durdona

yoshlar

0

12

1

13

2

200

3

14

Javob

0

Nechinch ismni ko'rish xohlaysiz?

3

II To'xtatib turish

To'xtatish

O'zizning dasturiy loyihasi

* Dasturga diqqat bilan qarasangiz, uni topishingiz kerak bo'lgan tartibdan 1 ni ayirish orqali qidiradi.

 manba kodlar

5-Dars. Dasturlash orqali tez va oson hisoblash

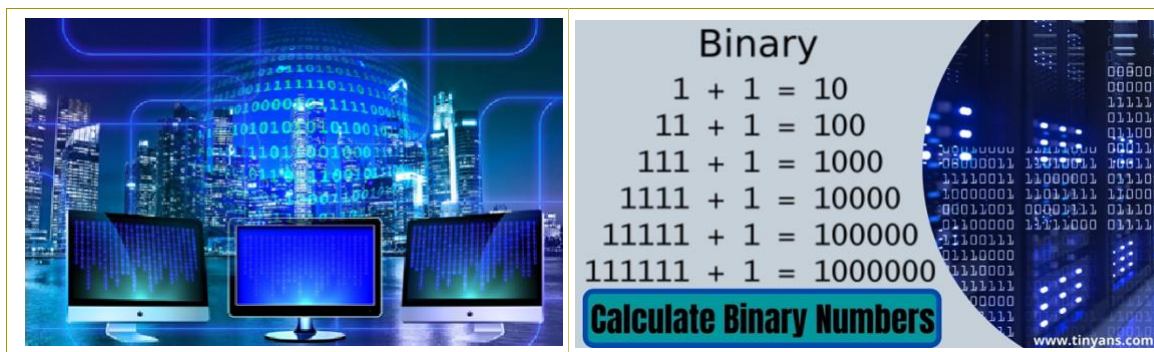
1

Kompyuter qanday hisoblarni qila oladi?

Kompyuter etimologiyasi Computer=compute+er bo'lib, hisob-kitoblarni amalga oshiradigan hisoblovchi mashina degan ma'noni anglatadi. Ammo biz duch keladigan muammolarning ko'pchiligini hisoblash yo'li bilan hal qilish mumkinligini bilasizmi? Unda kompyuterga muammolarni topshirib hisob-kitoblar qilinsa hal qilish yo'lini tez topish mumkin deb taxmin qilish mumkin. To'g'ri, lekin kompyuter tushunadigan shakl bo'lishi uchun oldin bizning kundalik hayotimizdagi muammolarni raqamli shaklga o'zgartirib kiritish kerak.

Hisob-kitoblarni kompyuterda bajarishning afzalliklari nimada? Kompyuterlar nafaqat hisoblash xatolaridan xoli, balki ular hisob-kitoblarni juda tez bajarishlari mumkin. Axir, murakkab hisob-kitoblar odamlarga qaraganda kompyuterlar bajarishi aniqroq va samaraliroq, to'g'rimi?

Kompyuter hisoblay oladi deganda hisob-kitoblarni amalga oshirishi mumkin? Kompyuterlar hisoblay oladigan qanday arifmetik amallar turlari mavjud? Ushbu darsda har xil turdagi arifmetik amallarni birgalikda ko'rib chiqamiz va ularni dasturlashga harakat qilamiz.



2

Entry-Python dagi arifmetik amallarni bilib olamiz.

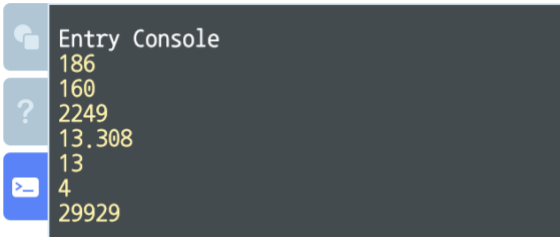
Arifmetik amallarning har xil turlari ham mavjud. Xususan, hayotda biz tez-tez uchrab turadigan to'rtta arifmetik amallari qo'shish, ayirish, ko'paytirish va bo'lishdir. Bu arifmetik amallarni oddiylarni to'g'ridan-to'g'ri yozish yoki aqliy arifmetika yordamida ham hal qilish mumkin, lekin agar u murakkab bo'lsa, kompyuterdan foydalanish qulay bo'ladi.

Entry-Pythonda foydalanish mumkin bo'lgan har xil turdagi arifmetik amallar mavjud. Keling, ushbu amallarning har birini sinab ko'raylik va natijalar qanday bo'lishini ko'rib chiqamiz.

Qo'shish, ayirish, ko'paytirish va bo'lish kabi arifmetik amallar uchun turli belgilardan misol uchun $+$, $-$, \times va \div dan foydalaniladi. Pythonda arifmetik amallarni bajarish buyruqlari ham arifmetik amalga qarab har xil ko'rinishda namoyon bo'ladi. Qanday hisoblash belgilari(operatorlar) mavjudligini bilish uchun quyidagi jadvalga qarang.

Arifmetik amallar(operatorlar)	Tavsif
$a + b$	Qo'shish
$a - b$	Ayirish
$a * b$	Ko'paytirish
a / b	Bo'lingan qism (3 kasrgacha yaxlitlangan)
$a // b$	Bo'lingan qism (Bo'linmaning butun qiymati)
$a \% b$	Qoldiq
$a ** 2$	Kvadrat daraja ($a**b$ amalida a soni b marta ko'paytiriladi, lekin Entry-Pythonda a faqat ikki marta ko'paytirilishgacha mumkin)

Chiqarish(output)dan foydalanib 7 ta arifmetik amallarni dasturlash qilib bajaring va natijaga qarang.

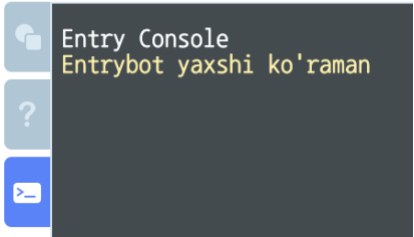
<pre> 3 import Entry 4 5 def when_start(): 6 Entry.print(173 + 13) 7 Entry.print(173 - 13) 8 Entry.print(173 * 13) 9 Entry.print(173 / 13) 10 Entry.print(173 // 13) 11 Entry.print(173 % 13) 12 Entry.print(173 ** 2) </pre>	
--	--

Agar raqamli qiymatlar o'rniga satrlidan foydalansak nima bo'ladi? **+ operator**da matnlarni birlashtiruvchi ma'noga ham ega. Agar siz ikkita matnni quyida ko'rsatilgandek $+$ belgisi bilan bog'lasangiz, belgilar birlashtirilib chiqarilganligini ko'rishingiz mumkin.

```

3 import Entry
4
5 def when_start():():
6     Entry.print(("Entrybot " + "yaxshi ko'raman"))

```



3 Taqqoslash va mantiqiy amallar natijalarini ko'ramiz.

Taqqoslash va mantiqiy amallar haqida eshitganmisiz? Bu notanish ibora bo'lishi mumkin, ammo bu biz allaqachon biladigan yoki oson va intuitiv ravishda tushuna oladigan hisoblash usuli. Avvalo, ushbu ikki amalda, natijada raqamli qiymatlar(0 yoki 1) chiqmaydi, o'rniga rost yoki qiymatlardan bir chiqadi.

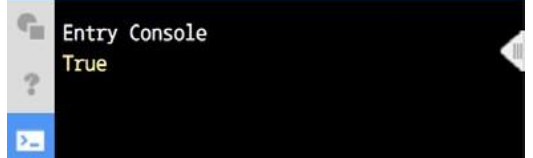
Avvalo, taqqoslash amalini ko'rib chiqaylik. Taqqoslash amalini ikki qiymatni bir-biri bilan taqqoslash natijasi to'g'ri yoki noto'g'ri ekanligini aniqlash uchun taqqoslaydigan amal deb tushunishingiz mumkin. Agar **taqqoslagan natija qiymati to'g'ri bo'lsa** rost(True), noto'g'ri bo'lsa, yolg'on(False) deb chiqaradi.

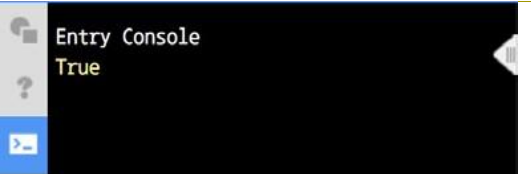
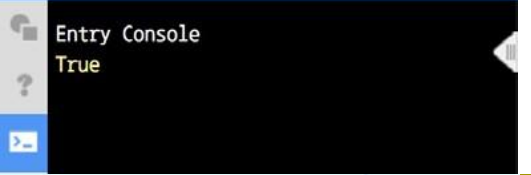
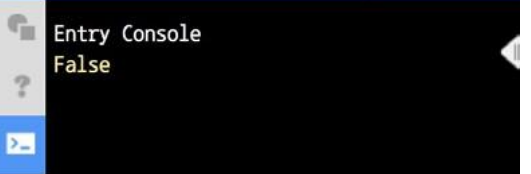
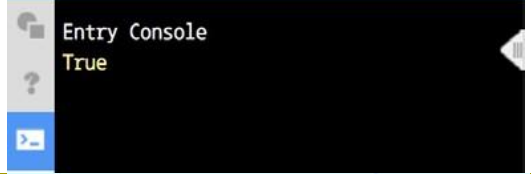
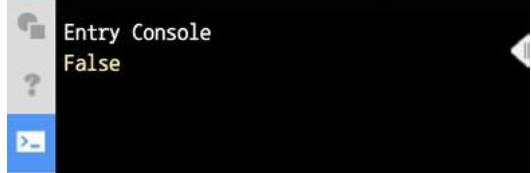
Taqqoslash amallari(operatorlar)	Tavsif
<code>a == b</code>	Ikkala qiymat teng bo'lsa, True, bo'lmasa, False
<code>a != b</code>	Ikkala qiymat teng bo'lmasa, True, bo'lsa, False
<code>a > b</code>	Chapdagi qiymat katta bo'lsa, True, kichik bo'lsa, False
<code>a < b</code>	O'ngdagi qiymat katta bo'lsa, True, kichik bo'lsa, False
<code>a >= b</code>	Chapdagi qiymat katta yoki teng bo'lsa, True yoki undan kichik bo'lsa, False
<code>a <= b</code>	O'ngdagi qiymat katta yoki teng bo'lsa, True yoki undan kichik bo'lsa, False

```

3 import Entry
4
5 def when_start():():
6     Entry.print(10 == 10)

```

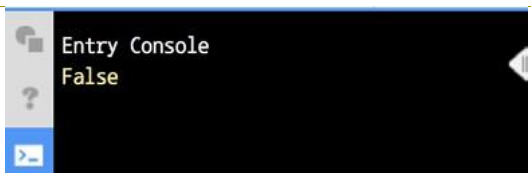


<pre> 3 import Entry 4 5 def when_start(): 6 Entry.print(15 != 10) </pre>	
<pre> 3 import Entry 4 5 def when_start(): 6 Entry.print(15 > 10) </pre>	
<pre> 3 import Entry 4 5 def when_start(): 6 Entry.print(15 < 10) </pre>	
<pre> 3 import Entry 4 5 def when_start(): 6 Entry.print(10 >= 10) </pre>	
<pre> 3 import Entry 4 5 def when_start(): 6 Entry.print(10 <= 9) </pre>	

Mantiqiy amallar - bu rost yoki yolg'on deb ma'lum bo'lgan qiymatlarni olish uchun mantiqiy operatorlar orqali hisoblaydigan operatsiya. Quydagi jadvalni ko'rib chiqib nima operatorlar mavjudligi va qanday ishlatishni bilib olishingiz mumkin. Tartibga solib ulardan foydalangan misol kodlarni yozib aniqroq tushunamiz.

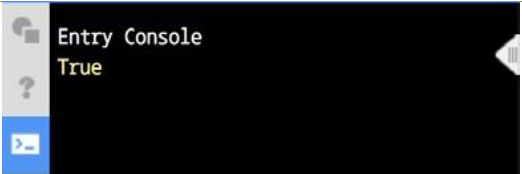
Mantiqiy amallari (operatorlar)	Tavsif
a and b	<u>Ikkala shartlar rost ekanligini tekshiradi.</u> Faqat Ikkala shartlar rost bo'lganda True
a or b	<u>Ikkala shartlardan biri rost ekanligini tekshiradi.</u> Ikkala shartlardan bir rost bo'lsa True
not a	<u>Qiymatning har doim teskarisini qaytaradi.</u> True bo'lsa False, False bo'lsa True

"and" operatorida ikkala qiymatlar rost bo'lgandagina rost qiymatiga ega, ya'ni rost(True) natija bo'ladi.

<pre> 3 import Entry 4 5 def when_start(): 6 Entry.print(((15 > 10) and (15 < 10))) </pre>	
--	--

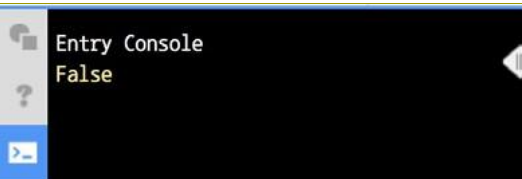
“or” esa operatorida, agar ikkita qiymatdan birortasi rost(True) bo'lsa, u 1 qiymatiga ega, ya'ni rost(True) natija bo'ladi.

```
3 import Entry
4
5 def when_start():
6     Entry.print(((15 > 10) or (15 < 10)))
```



“not” operatori har doim joriy qiymatning teskari qiymatini chiqaradi. Agar rost(True) bo'lsa, u yolg'on(False), agar yolg'on(False) bo'lsa, rost(True) chiqaradi.

```
3 import Entry
4
5 def when_start():
6     Entry.print(not (10 == 10))
```



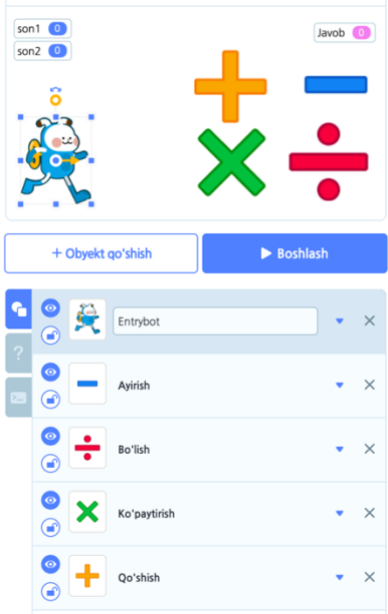
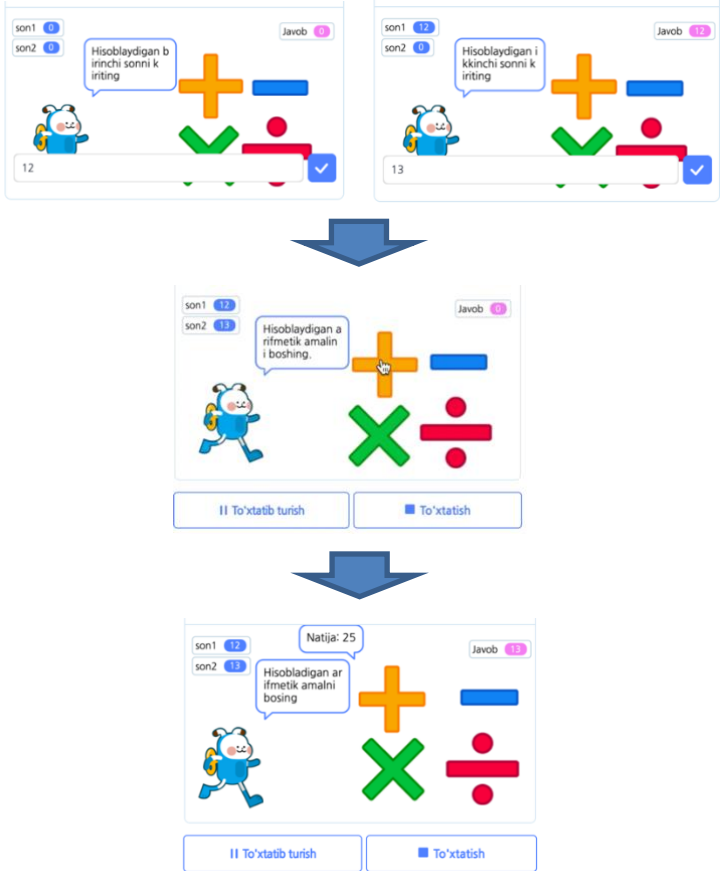



4



Kalkulyator topshirig'ini bajarib ko'ramiz.

- (1) Hozirgacha o'rganib olganlardan foydalanib 4ta turli arifmetik amallarni bajaradigan kalkulyatorni yasab ko'ramiz. Haqiqiy kalkulyatordek foydalanuvchi o'zining xohishdan hisoblaydigan 2ta sonlar va arifmetik amalini tanlab unga asosan hisoblab ko'ramiz.

Faqat e'tibor berish kerak qismi o'tgan darsda o'rgangan `when_start()` callback funksiyasidek har bir obyektни sichiqoncha orqali bosgan paytgina Entry dastur o'zidan avtomatik chaqiriladigan funksiyani `when_click_object_on()` dan foydalanib kod yozishingiz kerak.

1-Vazifa	
Oldin tayyor bo'lish kerakli obyektlar va o'zgaruvchilar	<ul style="list-style-type: none"> - Entrybotdan tashqari 4 ta obyektlar (qo'shish, ayirish, ko'paytirish, bo'lish) - 2ta o'zgaruvchilar: son1, son2

[Ishga tushirishdan oldin]	[Ishga tushirishgandan keyin]
	
Obyektlar	Obyektning kodlari
	<pre> 1 # Entrybot's Python code 2 3 import Entry 4 5 son1 = 0 6 son2 = 0 7 8 def when_start(): 9 Entry.input("Hisoblaydigan birinchi sonni kiriting") 10 son1 = Entry.answer() 11 Entry.input("Hisoblaydigan ikkinchi sonni kiriting") 12 son2 = Entry.answer() 13 Entry.print("Hisoblaydigan arifmetik amalni bosing") </pre>
	<pre> 1 # Qo'shish's Python code 2 3 import Entry 4 5 son1 = 0 6 son2 = 0 7 8 def when_click_object_on(): 9 Entry.print(("Natija: " + (son1 + son2))) </pre>
	<p>Obyekt uchun kerakli kodlarni to'ldiring.</p>

	Obyekt uchun kerakli kodlarni to'ldiring.
	Obyekt uchun kerakli kodlarni to'ldiring.

 manba kodlar

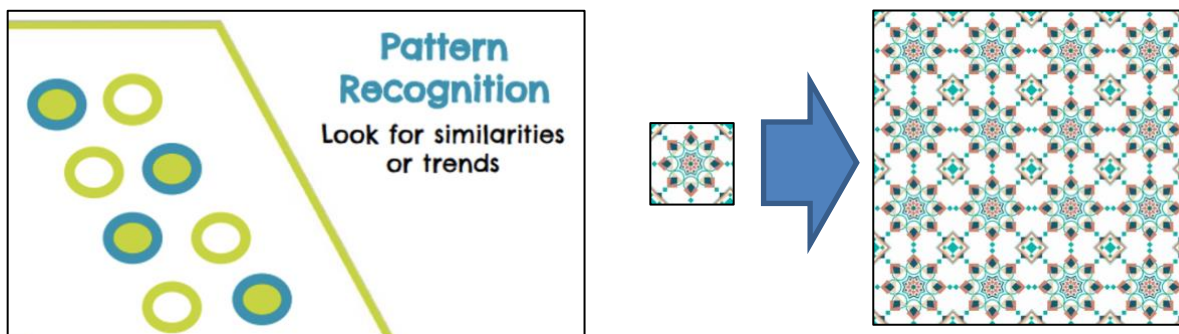
6-Dars. Kompyuterjon, qachongacha takrorlamoqchisiz?

1

Kompyuterning eng katta afzalligi, takrorlash?

Kundalik hayotimizga muammolarni diqqat bilan qarasangiz, umumiy xususiyatlarni topishingiz mumkin, agar siz u umumiy bo'lgan narsalarni yig'ib guruhlab ko'rsangiz takrorlanuvchi elementlarni topishingiz mumkin. Shu tarzda topilgan takrorlanuvchi elementlarni **naqsh(pattern)** deb ataldi. Muammolarni mana bu topgan naqshdan foydalansa yanada sodda va qulayroq ifodalashga imkon beradi.

Dasturlash orqali muammolarni hal qilish ham xuddi shunaqadir. Xususan, muammo qanchalik murakkab bo'lsa, shunchalik sodda, takrorlanadigan naqshlarni topishingiz kerak. Kompyuter muammolarni dasturlashtirilgan tarzda hal qiladi va minglab yoki o'n minglab takrorlanuvchi ishlarni yuqori tezlikda xatosiz bajarishi mumkin. Agar biz ushbu takrorlash qobiliyatini avtomatlashtirish orqali muammolarni hal qilsak, biz muammolarni samarali va aniq hal qila olamiz.



2

Kompyuterga takrorlash buyrug'ini berib ko'ramiz.

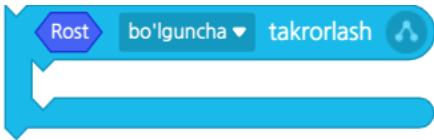
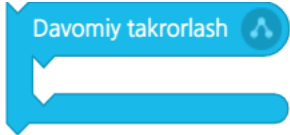
Takrorlash tom ma'noda kerakli buyruqni qayta-qayta bajarish imkoniyatini anglatadi. Ushbu takrorlash uchun bir nechta buyruqlar mavjud va ulardan foydalanish har xil. Muayyan buyruqni qayta-qayta bajarish uchun biz takrorlash sonini belgilashimiz mumkin yoki ma'lum bir shart bajarilsa, mantiqiy operatsiya qiymati qondirilmaguncha uni takrorlashimiz mumkin.

Kompyuterga buyruqlar orqali takrorlashni buyurganingizda, takrorlashlar sonini yoki vaziyatni to'g'ri yo'naltirsangiz muammoni eng samarali hal qilishingiz mumkin. Aks holda, kompyuter inson tarafidan qanday yo'naltiriganligiga qarab bajarishi sababli kompyuter juda ko'p resurslarni iste'mol qiladigan samarasiz dastur bo'lishi ham mumkin. Shuning uchun dasturlashda aniq ifoda etish qobiliyatingizni doimo rivojlantirib boring.

3 Davomiy yoki qo'yilgan shartlarga mos qondirgangina takrorlash usulini o'rganamiz.

O'tgan darsda arifmetik yoki mantiqiy amallar orqali ma'lumotlarni qanday boshqarishni o'rgangan edik. Bu darsda ham takrorlashni amalga oshirish uchun takrorlanish soni va shartni ifoda qilishda o'rgangan amallardan foydalanish zarurligi sababli ularni qo'llashni davom ettiramiz.

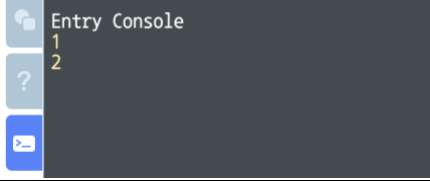
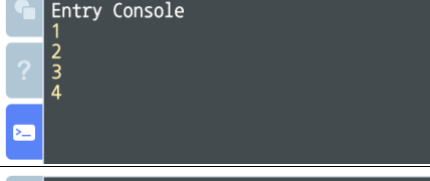
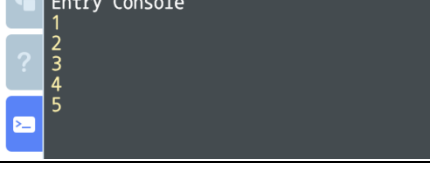
Avvalo, **"while"** sikl operatorning takrorlanish tuzilishini ko'rib chiqaylik. Aslida biz sikl operator o'zi nimaligini bilmasdan buni Entryda ko'p ishlatib ko'rganmiz.

Entrydagi takrorlash bloklar	Tavsifi
	Takrorlanishlar soni noma'lum bo'lganda, kod hatto bir marta ham ishga tushmasligi mumkin. Kodni ishga tushirishdan avval shart tekshiriladi. Agar shart yolg'on bo'lsa, unda sikldagi kod ishga tushmaydi.
	Takrorlashni boshlashdan avval tekshirgan shartda doim rost bo'lganda, cheksiz davom etib takrorlaydi.

Sintaksisi	Misol kodlari
while shart ifodalar : [4ta probel] bajaradigan buyruqlarning to'plami	n = 1 while n < 6 : Entry.print(n) n = n + 1

Misol kodni yanada batafsil ko'rib chiqamiz. **"while" operator yordamida takrorlashda, qachongacha takrorlashni aniqlash uchun shartni qo'yish muhimdir.** Avvalo, biz shartida taqqoslash amaldan foydalanish uchun **"n"** o'zgaruvchisini yaratib **"n"** o'zgaruvchining dastlabki qiymatni 1 ga sozlaymiz.

while sikl boshlash oldin boshda **"n"** qiymati 1 bo'lganda **"n < 5"** shartni tekshirishda natijasi rost bo'lganligi sababli **"while"** sikl ichidagi buyruqlarning to'plamidagi 2 ta qatorli 2 ta buyruqlarni bajarishga ichkariga kiradi. 1-qatorda **n** qiymatni konsolga chiqardi. Demak konsolda 1 chiqadi. Keyin 2-qatorda **"n = n + 1"** orqali **"n"**ga 1ni qo'shib yana **"n"**ga saqlaganda **"n"** qiymati 2 bo'ladi. Keyin 2 ta qatorli buyruqlarni to'liq oxirigacha bajarganda endi yana qaytadan **"while"** sikldagi shartni tekshirish vaqt keladi. Hozirgi **"n"** qiymati 2 bo'lgan bo'lsa ham **"n < 5"** taqqoslash amalning natijasi rost bo'ladi. Shuning uchun oldindek 2ta qatorli buyruqlarning to'plamini bajarish uchun sikl ichkariga kiradi. Demak shunday har bir marta tekshirgan shartning natijasi rost bo'lguncha takrorlaydi.

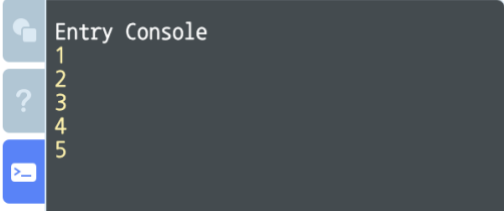
Sikl tartibi	"n" qiymati	Konsol natijasi	"n = n + 1" amla orqali "n" qiymati
1	1		2
2	2		3
3	3		4
4	4		5
5	5		6

Bu sikl qachon tugaydi? 5 marta takrorlab "n" qiymati 6 bo'lib endi shartni tekshirishda natijasi yolg'on bo'lganda paytda sikl ichkariga kirmasdan o'tib ketib sikl oxirigisining keyingi qatoriga ketadi. Bu kichik dasturning kodini Entry-Pythonda to'liq yozishda when_start() funksiya chaqirilganda ishga tushish uchun shunday bo'lish kerak. Faqat kodi yozishimizda "Tab" tugmachasini bosib bo'sh joylarni qoldirib va u orqali har bir funksiyaning hamda siklning chegarasi qayerdan qayergachaligini to'g'ri belgilash muhimligini unutmasligimiz kerak. Demak bu kichik dasturining maqsadi 5 marta takrorlab 1dan 5 gacha sonlarni konsolga chiqarish bo'ladi.

```

1  # Entrybot's Python code
2
3  import Entry
4
5  n = 1
6
7  def when_start():
8      while n < 6:
9          Entry.print(n)
10         n = n + 1

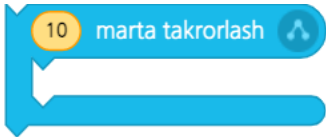
```



4

Nech marta takrorlashni aniqlab takrorlash usulini o'rganamiz.

Enid, "for" sikl operatorning takrorlanish tuzilishini ko'rib chiqaylik. Aslida biz mana bu sikl operator ham o'zi nimaligini bilmasdan buni Entryda ko'p ishlatib ko'rganmiz. "while" bilan fargi necha marta takrorlash kerakligini oldindan aniq ma'lum bo'lganda foydalaniladi.

Entrydagi takrorlash bloklar	Tavsifi
	Takrorlanishlar soni avvaldan takrorlaydi. Ma'lum bo'lganda qo'llaniladi.

Sintaksisi	Misol kodlari
if o'zgaruvchi nomi in range(takrorlash soni) : [4ta probel] bajaradigan buyruqlarning to'plami	<pre>n = 1 for i in range(5): Entry.print(n) n = n + 1</pre>

Oldin yasab ko'rgan 5 marta takrorlab 1dan 5gacha sonlarni konsolga chiqarish kichik dasturini "for" operatori orqali yaratsak tepadagi misol kodlaridek bo'ladi.



Qo'shimcha
ma'lumotlar

range() funksiyasi nima?

Aslida range() funksiyasi 3ta parametрни qabul qiladi. Lekin hozirgidek faqat 1ta parametri(takrorlanishlar soni)ni topshirib foydalansa, chiqarilgan ro'yxatidagi qiymati 0dan boshlaydi. Demak, agar misoldagidek range(5) deb chaqirsa [0,1,2,3,4] qiymatli ro'yxatini qaytarib beradi. Keyin har bir siklni takrorlash paytida ro'yxatning boshidan tartib bilan har bir qiymatini olib "for" amallaridagi o'zgaruvchi (misoldagi "i") ichida sozlab foydalanadi.

Bu kichik dasturning kodini Entry-Pythonda to'liq yozib ko'ramiz.

<pre>1 # Entrybot's Python code 2 3 import Entry 4 5 n = 1 6 7 def when_start(): 8 for i in range(5): 9 Entry.print(n) 10 n = n + 1</pre>	
--	--

5

Kalkulyator topshirig'ini bajarib ko'ramiz.

- (1) Entry [boshlang'ich darajali kitobi](#)da dasturlash asoslaridan "Takrorlash"ni o'rganishda amaliy mashg'ulot sifatida yasab ko'rgan "Gul bargi bilan gul yasaymiz" misolni Entry-Python orqali yasab ko'ramiz.

1-Vazifa	
Oldin tayyor bo'lish kerakli obyektlar va funksiyalar	
<p>[Ishga tushirishdan oldin]</p> <p>+ Obyekt qo'shish ▶ Boshlash</p>	<p>[Ishga tushirishgandan keyin]</p> <p> To'xtatib turish ■ To'xtatish</p>
Obyektlar	Obyektning kodlari

manba kodlar

7-Dars. Ikkitadan bittasini albatta tanlash kerak!

1

Tanlov, har doim bizlarga qiyin narsadir!

Biz hayotimizda tanlov qilishimiz kerak bo'lgan ko'plab vaziyatlarga duch kelamiz. Bu tanlovlar juda oddiylardan tortib bizning hayotimizga katta ta'sir ko'rsatadiganlargacha. Har doim biror narsani tanlash qiyin va qiyin deb aytiladi. Xo'sh, buni kompyuterda sinab ko'rishga nima deysiz? Keling, kompyuterni tanlovni amalga oshirishni aytish uchun nima qilishimiz mumkinligini o'ylab ko'raylik.

Inson bir vaqtning o'zida bir nechta vaziyatlarni solishtirishi va tanlashi mumkin. Ammo kompyuter faqat ikkita holatni solishtirishi mumkin. Ushbu cheklovlar tufayli kompyuterga muammoli vaziyatni taqdim etishda biz unga doimo ikkita vaziyatni berishimiz kerak.

O'tgan darslarda o'rgangan mantiqiy amallarni eslasak, kompyuterlar faqat ikkita narsani aytishi mumkin: rost va yolg'on, to'g'rimi? Buni tanlovning asosi, deyish mumkin. Kompyuterlar faqat 0 va 1 ni biladi. Bu ikkisi 1 rost, 0 yolg'on kabi ham ifodalanadi.

Shunday qilib faqat bu ikki narsani biladigan kompyuterga qanday qilib tanlov vaziyati haqida aytish mumkin? Keling, Entry-Python orqali birgalikda bilib olaylik.



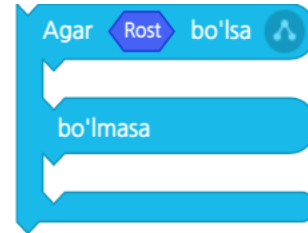
2

if operatori yordamida tanlash strukturasi yaratib ko'ramiz.

Birorta shartga ko'ra buyruqlar ketma-ketligining bajarilishi yoki bajarilmasligini belgilovchi algoritmi tarmoqlanuvchi algoritmlar deb ataladi. Shartlarni tekshirish uchun barcha dasturlash tillari kabi Python tilida ham shartli o'tish operatorlari mavjud.

Aslida biz u operator o'zi nimaligini bilmasdan Entry blokli dasturlashda ko'p ishlatib ko'rganmiz. Ketma-kitligi bilan bajarishda bu bloklarni uchrashganda bir shartlar tekshiriladi hamda rost yoki yolg'on qiymat qaytarishiga asoslanib, buyruqlar ketma-ketligi bajariladi

Entrydagi tanlash bloklar



Buni Python tilida **if** operatorida qanday sintaksisi orqali dasturlash qilishni ko'rib chiqamiz.

Sintaksisi	Tavsifi
if shart ifodalar : [4ta probel] bajaradigan buyruqlarning to'plami1 else : [4ta probel] bajaradigan buyruqlarning to'plami2	if operatori bilan birga else buyrug'ini ham qo'llash mumkin. Agar shart rost (True) qiymat qaytarsa, buyruqlar to'plami1 bajariladi, aks holda buyruqlar to'plami2 bajariladi.

Bu if operatoridan foydalanib kichik misol dasturini yasab ko'ramiz. Misol tariqasida, juft va toq sonlarni farqlash dasturini tuzamiz. Birinchidan, "n" deb nomlangan o'zgaruvchini yaratamiz va Entry.input() dan foydalanib foydalanuvchilardan raqamli qiymat(natural sonlar)ni olib saqlab turamiz. Keyin u saqlangan soni toq yoki juft sonlarligini ajratib olish uchun mezon tuzish kerak. Demak u mezon if operatoridagi shartlarning mantiqiy natijasi rost yoki yolg'on ekanligini aniqlay oladigan mezon bo'ladi.

Misol kodlari

```

n = 0

def when_start():
    Entry.input("Natural sonlardan birini.
    kiriting. Sonning juft yoki toq ekanligini bilib
    olaman.")
    n = Entry.answer()

    if n % 2 == 0 :
        Entry.print("Juft son")
    else :
        Entry.print("Toq son")
        
```

Keling, bu dasturda muhim bo'lgan mezonni batafsil ko'rib chiqaylik. "**n%2 == 0**" ni ikkiga bo'lib tushunishingiz kerak. Biz oldingi 5-darsda arifmetik amallardan bir bo'lgan "%" operatorini o'rgangandek "**n % 2**" buyruq bo'lib, n o'zgaruvchida saqlangan qiymatni 2 ga bo'lishdan qolgan qoldiqni bildiradi. Va qoldiq "**== 0**" bilan bog'langan. Demak "**==**"

taqqoslash operatori orqali bu qoldiq 0 ga teng yoki 0 ga teng emasligini ekanligini anglay oladi.




3

Takrorlash va tanlashni birlashtirib yasab ko'rish

(1) Misol dasturini rivojlantirib ko'ramiz. Hozirgi dasturi foydalanuvchi kiritgan sonini faqat bir martagina tekshirishga imkoniyat beradi. Endi foydalanuvchiga cheksiz tekshirishga imkoniyatni beramiz. O'tgan darsda o'rgangan takrorlash buyruqlardan foydalanib kiritilgan sonning natijasini 3 soniya davomida ko'rsatib keyin foydalanuvchiga yana sonni kiritish imkoniyatni beramiz.

1-Vazifa	
<pre> 1 # (1)Entrybot's Python code 2 3 import Entry 4 5 n = 0 6 7 def when_start(): 8 Entry.input("Natural sonlardan birini. kiriting. Sonning juft yoki toq ekanligini bilib olaman.") 9 n = Entry.answer() 10 11 if n % 2 == 0: 12 Entry.print("Juft son") 13 else: 14 Entry.print("Toq son") </pre>	
Mening dasturim	
Qanday hal qilish mumkin?	<p>* while operatori bilan print_for_sec() funksiyani ishlatish kerak.</p> <p>* Qayerdan qayergacha sikl qiladigan buyruqlarning to'plamini bo'lishni belgilash uchun bo'sh joylarni to'g'ri qo'yishda ehtiyot bo'ling.</p>

- (2) Takrorlash va tanlashni birlashtirib Entry [boshlang'ich darajali kitobi](#)da dasturlashning asoslaridan "Shart"ni o'rganishda amaliy mashg'ulot sifatida yasab ko'rgan "Changyutgich robot" misolni Entry-Python orqali yasab ko'ramiz.

2-Vazifa	
Oldin tayyor bo'lish kerakli obyektlar va funksiyalar	<div> <div>Boshlash tugmasini bosganda</div> <div> <div>while A :</div> <div>if A :</div> <div>Entry.is_touched(A)</div> </div> <div> <div>Entry.move_to_direction(A)</div> <div>Entry.add_rotation(A)</div> <div>Entry.start_drawing()</div> <div>Entry.set_brush_size(1)</div> <div>Entry.set_brush_color_to()</div> </div> </div>
<div> <div>[Ishga tushirishdan oldin]</div>  <div> <div>+ Obyekt qo'shish</div> <div>Boshlash</div> </div> </div>	<div> <div>[Ishga tushirishgandan keyin]</div>  </div>
Obyektlar	Obyektning kodlari
	

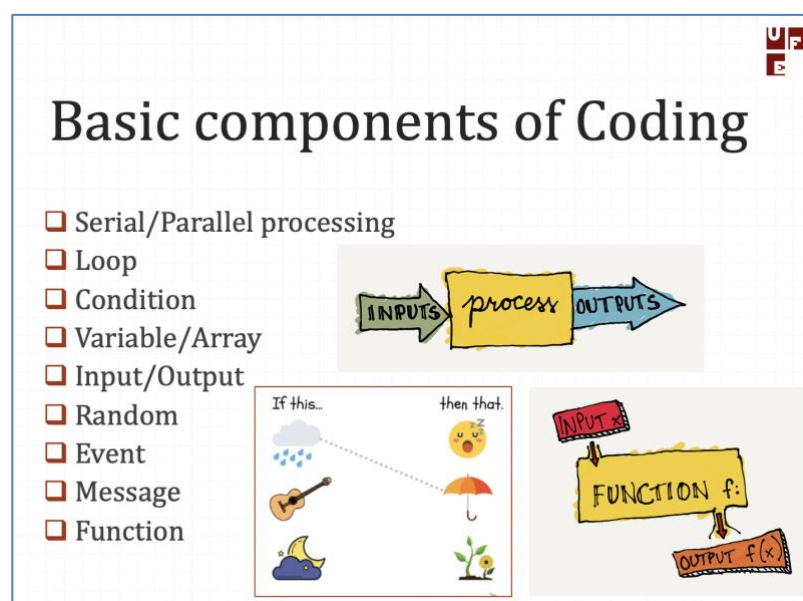
8-Dars. Up(Yuqoriga) yoki Down(Pastga) o'yinini yasab ko'ramiz

1 Matnli tillarda o'yin yaratishning iloji bormi?

Siz bilgan barcha o'yinlar va dasturlar matnli dasturlash tilida yozilgan. Har qanday dasturni matn tilida yozishingiz mumkinligiga ishona olasizmi? 1-darsda matnli tilining afzalligini bilib oldik, xohlagan narsalarni cheksiz yaratish uchun imkoniyatlar bor. Biz bilgan o'yinlarning ko'pchiligi murakkab kodlar asosida qiziqarli tuzilgan. Biz hozirgacha o'rgangan tushunchalar hali shunday yuqori darajali o'yinlarni yaratishga yetmaydi. Lekin siz bu kitob orqali Python dasturlashning asoslarini o'rganib keyin undan rivojlarni to'xtamasdan davom etib chuqurroq o'rganib olishga harakat qilsangiz albatta yuqori darajali murakkab o'yinlarni yasab olasiz.

2 Keling, Up(Yuqoriga) & Down(Pastga) o'yinini yasab ko'raylik

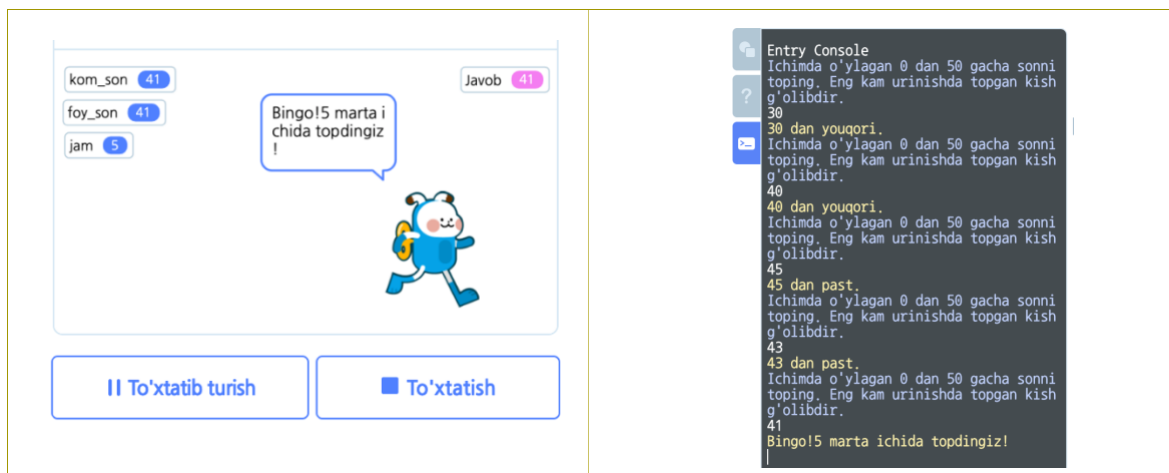
Bu kitob orqali hozirgacha o'rgangan dasturlashning asoslarni sanab ko'ramiz. **Ketma-ketlik**(Serial processing), **Funksiya**(Function), **Kiritish/Chiqarish**(Input/Output), **O'zgaruvchi/Ro'yxat**(Variable/List), **Takrorlash**(Loop), **Shart**(Condition)gacha 6ta komponentlarni o'rganib oldik. Bu darsda biz hozirgacha o'rganib olganlarimizni jamlab yana bir komponent, **Tasodifiy**(Random)ni qo'shib kichik o'yinini birga yasab ko'ramiz.



Entry-Pythonning ishga tushirish oynasi o'rniga konsol oynasida ishlaydigan kichik o'yinni yaratamiz. Siz oldin Entryda blokli dasturlash orqali dastur tuzgan paytingizdan farqli tuyg'ularga ega bo'lasiz!

Yangi narsalarni yaratishda, kimdir tomonidan oldindan batafsil tashkilashtirilgan narsani yaratish va uni noldan faqat talablar orqali chizmasiz yaratish o'rtasida katta farqlar mavjud. Hozir biz birinchi marta o'yin yaratayotganimiz sababli, biz qanday foydalanuvchi interfeysiga ega bo'lishligi haqida oldin tashkilashtirilganidan yaratishga harakat qilamiz.

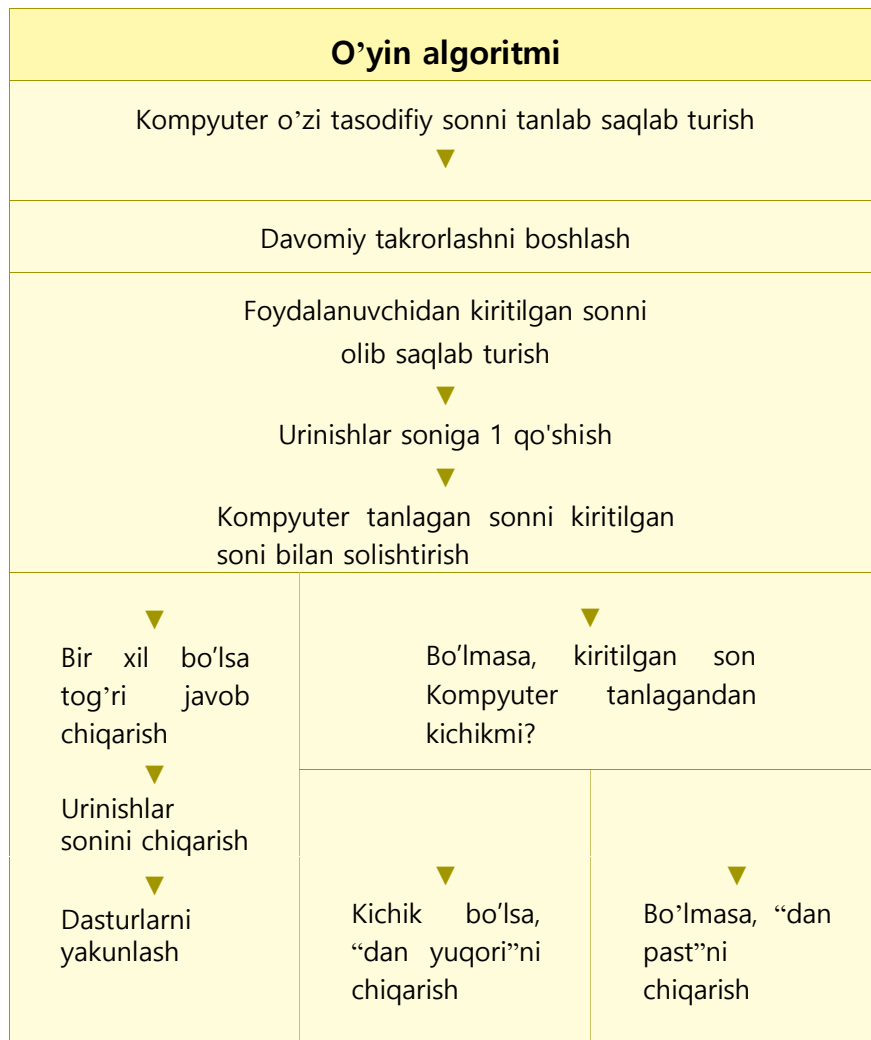
Ushbu o'yinning nomi **“yuqoriga yoki pastga”** bo'ladi. Foydalanuvchi kompyuter tomonidan o'ylangan 0 dan 50 gacha bo'lgan tasodifiy sonni taxmin qiladigan o'yindir. Biroq, qanchalik kamroq urinishlar qilsangiz, g'alaba qozonish shunchalik yuqori bo'ladi. Va ushbu o'yinning foydalanuvchi interfeysi quyidagicha bo'ladi.



Bu o'yini konsol o'yini bo'lganda faqat konsoldagi interfeysini qarasangiz bo'ladi. Keyingi jarayonda dasturlash asosiy komponentlarni birlashtirib ketma-ketlik bilan bajaradigan algoritmnı loyihalashtirish kerak. Keyin endgina algoritmgaga qarab kod yozishni boshlaysiz.

Interfeysi jihati / Dasturlash komponenti	Ma'lumotlar jihati / Dasturlash komponenti
O'yinning qo'llanmasini konsolga tanishtirish / Output	Kompyuter tanlagan tasodifiy sonni saqlab turish / Random, O'zgaruvchi
Foydalanuvchilar taxmin qilgan sonini klaviaturadan terish / Input	Foydalanuvchilar taxmin qilgan sonni saqlab turish / O'zgaruvchi
Foydalanuvchilar taxmin qilgan soni bilan Kompyuter tanlagan sonni solishtirib qanday farq borligini konsolga bildirish / Output	Ma'lumotlarni bir-biri bilan solishtirish va taqqoslash/ Shartlar
Foydalanuvchi kompyuterning sonini topmaganda yana taxmin qilishga imkon berish / Takrorlash	Foydalanuvchi necha marta urinib ko'rganini sanalgan sonni saqlab turish / O'zgaruvchi

Biz ikkita jihatlar tahlil qilib dasturlash komponentlardan nimalardan foydalanish kerakligini bilib oldik. Demak, o'zgaruvchilar, input, ouput, shartlar, takrorlashni birlashtirib algoritimni tuzib ko'ramiz.



Endigina algoritimga qarab kod yozishni boshlaymiz. Tahlil qilgan natijasida 3ta o'zgaruvchilar kerakligini bildik. Ularni e'lon qilishdan boshlaylik. Foydalanuvchi kiritgan soniga **"foy_son"**, Kompyuter tanlagan soniga **"kom_son"** va Foydalanuvchi urinishlar soniga **"jam"** deb nomlaymiz.

```

1  # (1) Entrybot's Python code
2
3  import Entry
4
5  kom_son= 0
6  foy_son = 0
7  jam = 0

```

Entryning "boshlash" tugmani bosgandan so'ng dasturimizni boshlash kerak bo'lganda `when_start()` funksiyasi ichida yozishni boshlash kerak. Tasodifiy sonni olish uchun oldin ishlatib ko'rmagan `random.randint()` funksiyaga qayerdan qayergacha sonlardan tanlash kerakligini 2ta parametr bilan bildirib foydalanish kerak. Foydalanuvchi to'g'ri javobni topmaganda davom etib urinishga imkoniyatni berish uchun **"while"** operatori ichida keyingi kodlarni yozishingiz kerak. Necha marta uringanini sanash uchun **"jam"** o'zgaruvchiga 1 qo'shish kerak. U maqsadda oldin biz **"jam = jam + 1"** deb kod yozganmiz. Buning qisqartirma kodi **"jam += 1"** bo'ladi.

```

8  def when_start():
9      kom_son = random.randint(0, 50)
10     while True :
11         Entry.input("Ichimda o'ylagan 0 dan 50 gacha sonni toping. " +
12                     "Eng kam urinishda topgan kishi g'olibdir.")
13         foy_son = Entry.answer()
14         jam += 1

```

Oddiy o'yinini yaratsa, foydalanuvchi kiritilgan soni bilan kompyuter tasodifiy tanlagan sonlar orasida tengligini tekshirish uchun faqat bitta tanlov tuzish yetadi. Biroq, yanada qiziqarli o'yin bo'lishi uchun farqni foydalanuvchilarga ma'lum qilish kerak. Birinchi bir-birga tengligi yoki teng bo'lmasligini tekshirish uchun uni **"If ~ else"** orqali ikkiga bo'lish kerak. Agar ikkita qiymat teng bo'lmasa, **"else"** ning ichkariga kirib yana katta yoki kichikligini bildirish uchun **"if ~ else"** ga bo'linadi.

Avvalo kiritilgan qiymat to'g'ri javob bo'lsa, so'zlarni **"Bingo! (To'g'ri javob)"** deb va **"Urinishlar soni"** deb chiqaramiz. Biz oldingi darslarda matnli qiymatlarni **"+"** operatori orqali birlashtirish mumkinligini o'rgandik. Undan foydalanib tartibda joylashtiring. Va chiqarib tugagach, o'yinni tugatish uchun **Entry.stop_code ("all")** funksiyasini chaqiring.

```

15     if foy_son == kom_son :
16         Entry.print("Bingo! " + jam + " urinish ichida topdingiz!")
17         Entry.stop_code("all")
18     else :
19         if foy_son < kom_son :
20             Entry.print(foy_son + " dan yuqori.")
21         else :
22             Entry.print(foy_son + " dan past.")
23         Entry.Wait_for_sec(2)

```

O'yinni ishga tushiring, konsol oynasini tekshiring va u to'g'ri ishlayotganligini tekshiring. Bundan tashqari, o'yinni bir necha marta ishga tushirishga harakat qiling va hech qanday xatolik yo'qligiga ishonch hosil qiling. Agar xatolik topilsa uning asosiy sababini topib uni to'g'rilab qaytadan ishga tushirib u hal qilinganligini tekshiradi. Bunday jarayon haqida **"Debugging"** deb ataladi. **"Debugging"** jarayoni tugab sinov jarayonigacha to'liq o'tsangiz endi maqsadli dasturimizni oxirigacha to'liq yasagandek bo'lasiz.



Qo'shimcha
ma'lumotlar

"elif" operatori bilasizmi?

Bilasizki Entry-Python aslida Python bilan 100% bir xil emas. Python tilida bu dasturni yaratishingizda "if-else" ni 2 martadan foydalanish emas, "if-elif-else" dan foydalanishingiz ham mumkin. Undan foydalanib kod yozsa pastdagidek bo'ladi.

```
if foy_son == kom_son :
    Entry.print("Bingo! " + jam + " urinish ichida topdingiz!")
    Entry.stop_code("all")
elif foy_son < kom_son :
    Entry.print(foy_son + " dan yuqori.")
else :
    Entry.print(foy_son + " dan past.")
Entry.Wait_for_sec(2)
```

3

Entry-Python kodlari va Python kodlarini solishtirib ko'rish

 Entry-Python
manba kodlar

```
1 # (1)Entrybot's Python code
2
3 import Entry
4
5 kom_son = 0
6 foy_son = 0
7 jam = 0
8
9 def when_start():
10     kom_son = random.randint(0, 50)
11
12     while True:
13         Entry.input("Ichimda o'ylagan 0 dan 50 gacha sonni toping. " +
14                     "Eng kam urinishda topgan kishi go'libdir.")
15         foy_son = Entry.answer()
16         jam += 1
17
18         if (foy_son == kom_son):
19             Entry.print(("Bingo! " + jam) + " urinish ichida topdingiz!")
20             Entry.stop_code("all")
21         else:
22             if (foy_son < kom_son):
23                 Entry.print(foy_son + " dan yuqori.")
24             else:
25                 Entry.print(foy_son + " dan past.")
26             Entry.wait_for_sec(2)
27
```

 Python
manba kodlar

```
1 import random
2
3 kom_son = 0
4 foy_son = 0
5 jam = 0
6
7 kom_son = random.randint(0, 50)
8 while True:
9     foy_son = input("Ichimda o'ylagan 0 dan 50 gacha sonni toping. " +
10                    "Eng kam urinishda topgan kishi g'olibdir.\n")
11     foy_son = int(foy_son)
12     jam += 1
13
14     if foy_son == kom_son:
15         print(("Bingo! " + str(jam)) + " urinish ichida topdingiz!\n")
16         quit()
17     elif foy_son < kom_son:
18         print(str(foy_son) + " dan yuqori.\n")
19     else:
20         print(str(foy_son) + " dan past.\n")
```


Entry-Python va Python kodlarini solishtiramiz. "**Entry**" deb nomlangan kutubxona faqat Entry-Python ichida kerak. Shuning uchun, Pythonda endi "Entry" kutubxonasini **import** qilishning hojati yo'q. Buning o'rniga, kompyuterdan tasodifiy sonlarni olish uchun ishlatiladigan **randint** funktsiyasi dastlab "**random**" deb nomlangan kutubxonada bo'lgan, shuning uchun "random" kutubxonasidagi randint funktsiyasidan foydalanish uchun u oldin **import** qilingan. Lekin Entry-Pythonda bu qism sizga qulay bo'lishi uchun o'tkazib yuborilgan.

Ikkinchidan, Pythonda `when_start` funktsiyasi kabi "**callback**" funktsiyalari mavjud bo'lmagani uchun funktsiya ishlatilmadi.

Vanihojat, Entry-Pythonni kiritish va chiqarish va Pythondagi kiritish va chiqarish funktsiyalaridan foydalanish boshqacha bo'lgani uchun Python da kiritish funktsiyasi chaqirgandan keyin foydalanuvchi kiritgan qiymatni `foy_son` o'zgaruvchisida saqlangan. Biroq, foydalanuvchi kiritgan bu qiymat satrli qiymati bo'lgani uchun uni darhol ishlatib bo'lmaydi va uni **int** funktsiyasi yordamida sonli qiymatga aylantirgandan so'ng foydalaniladi. Xuddi shunday, `print` funktsiyasida sonli qiymatlarni avvalgidek chiqarish mumkin emas, shuning uchun ular **str** funktsiyasi orqali satrlarga aylantiriladi va ishlatiladi.

4

O'zmizning Funktsiyamizidan o'yinni o'zgartirish

Hozircha biz faqat boshqalar tomonidan yaratilgan funktsiyalar yordamida kodladik. Biroq, funktsiyalarni o'zingiz yaratish va ishlatish uchun sizga ham bilim kerak bo'lganligi sababli, kodlarmizning bir qismini funktsiyaga almashtirib kodlashni o'rganamiz. Oldin [2-dars](#)da funktsiyalar yaratish grammatikasini ko'rib chiqqanimiz sababli, shu grammatika bo'yicha funktsiya yaratamiz.

Birinchidan, kodning qaysi qismini funktsiyaga almashtirishni qaror qilishimiz kerak. **Funktsiyaga almashtirish mezonlari kodning bir nechta joyida bir xil kod qayta-qayta ishlatiladigan qismlar yoki kodning o'qilishida yaxshiroq tushunchasi beradigan maqsadga to'g'ri keladigan qismi bo'lishi mumkin.**

Shuning uchun aslida, ushbu koddagi funktsiyaga almashtirish kerak bo'lgan qism ikkita qiymat(`kom_son` va `foy_son`)ning o'lchamlarini taqqoslaydigan qism bo'lishi kerak edi, lekin afsuski, Entry-Python cheklovlari tufayli funktsiya bajarilish natijasini chaqiruvchiga qaytara olmaydi, shuning uchun u taqqoslash natijasini qaytarmaydi va natijani foydalanuvchiga chiqaradigan qismi ham birga funktsiyaga almashtiriladi.

Shunday qilib, Entry-Python asosan Pythonning barcha funktsiyalarini sinab ko'rishda cheklovlar mavjudligi tan olinadi va Python grammatikasining asoslari o'rganib olinsa mamnun bo'ladi. Agar bu maqsadga erishilgan bo'lsa, keyingi qadam **Pygame zero** orqali Python kodlashni o'rganish bosqichiga o'tib haqiqiy Python tilidan foydalanib kodlashni boshlashingizni tavsiya qilamiz.

Yuqorida aytib o'tilgan kodda foydalanuvchidan ma'lumot oladigan qism 18~26 qatorlar bo'ladi. Endi funktsiyani yaratishda keyingi qadam funktsiya ichidagi kodni yaxshi tavsiflovchi funktsiya nomini tanlashdir. Funktsiyani o'zbek tilida nomlash mumkin, lekin dunyoning boshqa mamlakatlaridagi dasturchilar bilan muloqot qilish uchun iloji boricha ingliz tilidan foydalanishni odat qilish kerak. Biz "**compare_num**" funktsiyasi deb nomlansa "sonlarni taqqoslash" ma'nosiga to'g'ri keladi.

Funktsiyaga almashtirishda, funktsiyani chaqiruvchi tomondan o'tkazish kerak bo'lgan ma'lum bir qiymat bor yoki yo'qligini ko'rib chiqishingiz kerak (funktsiya tomoni buni **parametr** deb ataydi va funktsiyani chaqiruvchi uni **argument** deb ataydi). Biz yaratgan funktsiyaga ikkita sonli qiymat(kom_son va foy_son)ni o'tkazishimiz kerakligi sababli, funktsiyasi ikkita parametrغا ega deb aytishimiz mumkin.

Endi shu paytgacha ko'rib chiqqan mazkur narsalarni funktsiya grammatikasiga ko'ra kodga yozsak, u quyidagicha bo'ladi.

```
def compare_num(num1, num2):
    if (num1 == num2):
        Entry.print(("Bingo! " + jam) + " urinish ichida topdingiz!")
        Entry.stop_code("all")
    else:
        if (num1 < num2):
            Entry.print(num1 + " dan yuqori.")
        else:
            Entry.print(num1 + " dan past.")
        Entry.wait_for_sec(2)
```

Quyidagi kodlari funktsiyani qo'llash orqali kodni o'zgartirish natijasi keltirilgan.

Entry-Python manba kodlar

```
1 # (1)Entrybot's Python code
2
3 import Entry
4
5 kom_son = 0
6 foy_son = 0
7 jam = 0
8
9 def compare_num(num1, num2):
10     if (num1 == num2):
11         Entry.print(("Bingo! " + jam) + " urinish ichida topdingiz!")
12         Entry.stop_code("all")
13     else:
14         if (num1 < num2):
15             Entry.print(num1 + " dan yuqori.")
16         else:
17             Entry.print(num1 + " dan past.")
18         Entry.wait_for_sec(2)
19
20 def when_start():
21     kom_son = random.randint(0, 50)
22
23     while True:
24         Entry.input("Ichimda o'ylagan 0 dan 50 gacha sonni toping. " +
25                     "Eng kam urinishda topgan kishi go'libdir.")
26         foy_son = Entry.answer()
27         jam += 1
28
29         compare_num(foy_son, kom_son)
30
```

Python manba kodlar

```
1 import random
2
3 kom_son = 0
4 foy_son = 0
5 jam = 0
6
7 def compare_num(num1, num2):
8     if num1 == num2:
9         print(("Bingo! " + str(jam)) + " urinish ichida topdingiz!\n")
10        quit()
11    elif num1 < num2:
12        print(str(num1) + " dan yuqori.\n")
13    else:
14        print(str(num1) + " dan past.\n")
15
16 kom_son = random.randint(0, 50)
17 while True:
18     foy_son = input("Ichimda o'ylagan 0 dan 50 gacha sonni toping. " +
19                     "Eng kam urinishda topgan kishi g'olibdir.\n")
20     foy_son = int(foy_son)
21     jam += 1
22
23     compare_num(foy_son, kom_son)
```

