N.N. Zaripov

ZAMONAVIY DASTURLASH TILLARI

DARSLIK

N.N. Zaripov

ZAMONAVIY DASTURLASH TILLARI

Oliy o‘quv yurtlarining 60110600 - «Matematika va informatika» bakalavr ta’lim yo‘nalishi talabalari uchun
DARSLIK

“KAMOLOT” nashriyoti
Buxoro-2024

UO‘K: 519.682
KBK: 32.973.2
Z32

Zaripov Nozimbek Nayimovich, Zamonaviy dasturlash tillari / [Matn]: darslik / N.N. Zaripov - Buxoro : “BUXORO DETERMINANTI” MCHJning Kamolot nashriyoti, 2024. - 144 b.

Ushbu darslik oliy o‘quv yurtlarining 60110600 – “Matematika va informatika” hamda 60540200 – “Amaliy matematika” bakalavr ta’lim yo‘nalishi talabalari va professor-o‘qituvchilari uchun mo‘ljallangan. Darslikda zamonaviy dasturlash tillari shuningdek, Python dasturlash tilida dastur tuzish hamda funksiya va protseduralar bilan ishlashga doir dasturlar tuzish metodlari yoritilgan berilgan. Ushbu darslikdan oliy o‘quv yurtlari talabalaridan tashqari, akademik litsey, professional ta’lim muassasalari, umumiy o‘rta maktab o‘qituvchi hamda o‘quvchilari va mustaqil o‘rganuvchilar foydalanishlari mumkin.

Muallif:

N.N.Zaripov – Buxoro davlat pedagogika instituti «Aniq fanlar» kafedra dotsenti, p.f.f.d (PhD).

Taqrizchilar:

T.R.Ro‘ziyev — Buxoro davlat pedagogika instituti «Aniq fanlar» kafedra dotsenti, f-m.f.f.d (PhD).

O.I.Jalolov — Buxoro davlat universiteti «Amaliy matematika va dasturlash texnologiyalari» kafedra mudiri, f-m.f.n., dotsent

ISBN: 978-9910-729-41-6

Ushbu darslik Oliy ta’lim, fan va innovatsiyalar vazirligining 2024-yil 29-maydagi 194-sonli buyrug‘iga asosan nashrga ruxsat berildi. Ro‘yxatga olish raqami 194-147.

MUNDARIJA

KIRISH 4
1. Algoritm va algoritmlar samaradorligini baholash 5
1.1. Algoritm tushunchasi va ular haqida ma’lumotlar 5
1.2. Algoritm xossalari, turlari va ularning berilish usullari 6
1.3. Saralash algoritmlari 13
2. Zamonaviy dasturlash tillari 14
2.1. Dasturlash tillari va ularning klassifikatsiyasi 14
2.1. Yuqori darajali dasturlash tillari 17
2.2. Python dasturlash tili. IDLE bilan tanishish 18
2.3. Python dasturlash tilida xatoliklar 21
3. Python dasturlash tili va sintaksisi 26
3.1. Python tilida o‘zgaruvchilar va operatorlar 26
3.2 Python tilida ma’lumot turlari 28
3.3. Python tilida satrlar 35
4. Pythonda ma’lumot to‘plamlari va turlari 39
4.1. List ro‘yxatlar va ular bilan ishlashda qo‘llaniladigan funksiyalar va metodlar 39
4.2. Tuple kortejlar bilan ishlash 48
4.3. Set to‘plamlar bilan ishlash 52
4.4. Dictionary va ular bilan ishlash 56
5. Pythonda shart operatorlari 61
5.1. Pythonda if, else, elif operatorlari 61
5.2. Pythonda takrorlanuvchi operatorlar 62
5.3. Pythonda break va continue operatorlari 65
6. Pythonda funksiya va modullar 67
6.1. Funksiyalar. Lambda funksiyasi 67
6.2. Pythonda maxsus modullardan foydalanish 71
6.3. Pythonda fayllar bilan ishlash 74
7. Pythonda grafika 79
7.1. Grafika bilan ishlash operatorlari 79
7.2. Grafika bilan ishlovchi funksiyalar 80
XULOSA 135
ATAMALARNING IZOHLI LUG`ATI 136
TEST SAVOLLARI. 139
ADABIYOTLAR RO‘YXATI. 142

KIRISH

Bugungi globallashuv jarayoni jadal sur’atlar bilan hayotimizga kirib kelayotgan bir davrda, o‘z bilimi, kuchi, imkoniyatlariga tayanadigan, jamiyat, davlat manfaatlarini o‘z manfaatlari bilan uyg‘un holda ko‘radigan yetuk mutaxassis-kadrlarni tayyorlash zarurligi, mamlakatda demokratik o‘zgarishlarni yanada chuqurlashtirish va fuqarolik jamiyati asoslari konsepsiyasini amalga oshirish sharoitida mustaqil, ijodiy fikrlovchi, zamonaviy fan-texnika va uning ishlash texnologiyasini mukammal egallagan, tadbirkor mutaxassislarni tayyorlash muhim vazifalardan biriga aylandi.

Hozirgi zamonaviy axborot texnologiyalari keng taraqqiy etgan bir davrda oliy o‘quv yurtlari talabalariga dasturlash tillarini o‘qitishda alohida uslub bilan yondashish, soddalashtirilgan usullarni ishlab chiqish dolzarb masalalardan biri hisoblanadi. Bugungi kunda oliy o‘quv yurtlari talabalarida kompyuter savodxonligini oshirish, axborot texnologiyalardan foydalanish madaniyatini rivojlantirish, ularda ayniqsa, zamonaviy dasturlash tillari fanini o‘qitish jarayonida o‘z mutaxassisliklariga doir ilmiy-nazariy ma’lumot va amaliy ko‘nikma hamda malakalar bilan qurollantirish jarayonida, ularda o‘z kasbiga bo‘lgan qiziqish, mas’uliyat kabi tushunchalarni ham takomillashtirishni amalga oshirish muhim ijtimoiy-pedagogik hodisa sanaladi. Shu boisdan ushbu darslik talabalar uchun muhim ahamiyat kasb etadi.

Ushbu darslik orqali talabalar zamonaviy dasturlash tillari hamda Python dasturlash tiliga doir tushunchalar, jumlardan: funksiyalar hosil qilish, grafika bilan ishlash, operatorlar, maxsus modullar, dastur bajarilish jarayonida sodir bo‘ladigan xatoliklar haqida to‘liq ma’lumotlar keltirilgan. Mazkur darslik dasturlashga qiziquvchi yoshlarni bilim, ko‘nikma va malakalarini rivojlanishiga yordam beradi.

Muallif

1. Algoritm va algoritmlar samaradorligini baholash
1.1. Algoritm tushunchasi va ular haqida ma’lumotlar

Algoritm so‘zi va tushunchasi IX asrda yashab ijod etgan buyuk alloma Abu Abdulloh Muhammad ibn Muso al-Xorazmiy (783 – 850) nomi bilan bog’liq.

Algoritm – bu ijrochi uchun qo‘yilgan masalani yechishga qaratilgan aniq va tushunarli ko‘rsatmalarning chekli ketma-ketligidir.

Algoritmning ijrochisi — algoritmda belgilangan buyruq yoki ko‘rsatmalarni bajarishga qodir mavhum (abstrakt) yoki moddiy (texnik, biologik yoki biotexnik) tizimdir.

Ijrochi bajara olishi mumkin bo‘lgan ko‘rsatma yoki buyruqlar to‘plamiga ijrochining ko‘rsatmalar tizimi deyiladi.

Algoritmlarning asosiy xossalari

- Diskretlilik
- Aniqlilik
- Tushunarlilik
- Ommaviylik
- Natijaviylik

Algoritmning diskretlilik xossasi. Masalan: bir joydan boshqa joyga o'tish uchun oyoqlarimiz bir necha yuz marta takrorlanadigan “o'ng oyoq oldinga qo‘yilsin”, “chap oyoq oldinga qo‘yilsin” buyruqlari ketma-ket beriladi. Bu buyruqlar baravar berilsa nima hodisa ro‘y berishini izohlash shart emas.

Masalan: Kompyuterda matn yozishda ham klaviaturadan har bir belgi ketma-ket berilgan alohida buyruqlar asosida yoziladi. Birdaniga bir necha o‘n klavishni bosish mumkinmikan?

Algoritmning natijaviylik xossasi: Natijaviylik xossasi algoritm qadamining chekliligini anglatadi.

Masalan:

1) daryodan bir chelak suv olinsin;
2) chelakdagi suv daryoga solinsin;
3) 1-bandga o'tilsin.

Bu algoritm chekli qadamda tugamaydi, demak natijasi yo‘q.

Masalan: 1 dan 31.622.400.000 gacha sanalsin. Bu ko‘rsatmani bajara olmaysiz, chunki siz to‘xtamasdan sekundiga beshta sonni

aytsangiz 6.324.480.000 sekund= l05.408.000 minut = 1756800 soat=73200 sutka=200 yil kerak bo‘ladi.

Algoritmning ommaviylik xossasi. **Masalan**:

- Al-Xorazmiyning ustunli qo'shish algoritmi barcha o'nli kasr ko'rinishidagi sonlar uchun o'rinli;
- Evklid algoritmi barcha natural sonlar uchun o'rinli;
- kvadrat tenglama yechish algoritmi barcha haqiqiy sonlar uchun o'rinli;
- oyoq kiyimi kiyishda avval paypoq keyin tufli kiyish (teskarisi emas) barcha aqli raso inson uchun o'rinli.
- Evklid algoritmiga misol: N=18 va M=39 bo'lsa, natija (3, 3) bo‘ladi.

SAVOL VA TOPSHIRIQLAR

1. Algoritm tushunchasi?
2. Algoritmlarning asosiy xossalari?
3. Algoritmning diskretlilik xossasiga misol keltiring?
4. Algoritmning aniqlilik xossasiga misol keltiring?
5. Algoritmning tushunarlilik xossasiga misol keltiring?
6. Algoritmning ommaviylik xossasiga misol keltiring?
7. Algoritmning natijaviylik xossasiga misol keltiring?
8. Algoritm ijrochisi haqida nimalar bilasiz?

1.2. Algoritm xossalari, turlari va ularning berilish usullari

1. Diskretlilik (Cheklilik). Bu xossaning mazmuni algoritmlarni doimo chekli qadamlardan iborat qilib bo‘laklash imkoniyati mavjudligida. Ya’ni, uni chekli sondagi oddiy ko‘rsatmalar ketma-ketligi shaklida ifodalash mumkin. Agar kuzatilayotgan jarayonni chekli qadamlardan iborat qilib qo‘llay olmasak, uni algoritm deb bo‘lmaydi.

2. Tushunarlilik. Biz kundalik hayotimizda berilgan algoritmlar bilan ishlayotgan elektron soatlar, mashinalar, dastgohlar, kompyuterlar, turli avtomatik va mexanik qurilmalarni kuzatamiz. Ijrochiga tavsiya etilayotgan ko‘rsatmalar, uning uchun tushunarli mazmunda bo‘lishi shart, aks holda ijrochi oddiygina amalni ham bajara olmaydi. Undan tashqari, ijrochi har qanday amalni bajara

olmasligi ham mumkin. Har bir ijrochining bajarishi mumkin bo‘lgan ko‘rsatmalar yoki buyruqlar majmuasi mavjud, u ijrochining ko‘rsatmalar tizimi deyiladi. Demak, ijrochi uchun berilayotgan har bir ko‘rsatma ijrochining ko‘rsatmalar tizimiga mansub bo‘lishi lozim. Ko‘rsatmalarni ijrochining ko‘rsatmalar tizimiga tegishli bo‘ladigan qilib ifodalay bilishimiz muhim ahamiyatga ega. Masalan, quyi sinfning a’lochi o‘quvchisi "son kvadratga oshirilsin" degan ko‘rsatmani tushunmasligi natijasida bajara olmaydi, lekin "son o‘zini o‘ziga ko‘paytirilsin" shaklidagi ko‘rsatmani bemalol bajaradi, chunki u ko‘rsatma mazmunidan ko‘paytirish amalini bajarish kerakligini anglaydi.

3. Aniqlik. Ijrochiga berilayotgan ko‘rsatmalar aniq mazmunda bo‘lishi zarur. Chunki ko‘rsatmadagi noaniqliklar mo‘ljaldagi maqsadga erishishga olib kelmaydi. Odam uchun tushunarli bo‘lgan "3-4 marta silkitilsin", "5-10 daqiqa qizdirilsin", "1-2 qoshiq solinsin", "tenglamalardan biri yechilsin" kabi noaniq ko‘rsatmalar robot yoki kompyuterni qiyin ahvolga solib qo‘yadi.

Bundan tashqari, ko‘rsatmalarning qaysi ketma-ketlikda bajarilishi ham muhim ahamiyatga ega. Demak, ko‘rsatmalar aniq berilishi va faqat algoritmda ko‘rsatilgan tartibda bajarilishi shart ekan.

4. Ommaviylik. Har bir algoritm mazmuniga ko‘ra bir turdagi masalalarning barchasi uchun ham o‘rinli bo‘lishi kerak. Ya’ni masaladagi boshlang‘ich ma’lumotlar qanday bo‘lishidan qat’iy nazar algoritm shu xildagi har qanday masalani yechishga yaroqli bo‘lishi kerak. Masalan ikki oddiy kasrning umumiy mahrajini topish algoritmi, kasrlarni turlicha o‘zgartirib bersangiz ham ularning umumiy mahrajlarini aniqlab beraveradi. Yoki uchburchakning yuzini topish algoritmi, uchburchakning qanday bo‘lishidan qat’iy nazar, uning yuzini hisoblab beraveradi.

5. Natijaviylik. Har bir algoritm chekli sondagi qadamlardan so‘ng albatta, natija berishi shart. Bajariladigan amallar ko‘p bo‘lsa ham baribir natijaga olib kelishi kerak. Chekli qadamdan so‘ng qo‘yilgan masala yechimga ega emasligini aniqlash ham natija hisoblanadi. Agar ko‘rilayotgan jarayon cheksiz davom etib natija bermasa, uni algoritm deb atay olmaymiz.

Algoritm quyidagi usullarda tasvirlash mumkin:

1. Algoritmning so‘zlar yordamida ifodalanishi;
2. Algoritmning formulalar yordamida ifodalanishi;

3. Algoritmning jadval yordamida ifodalanishi;
4. Algoritmning grafik shaklda ifodalanishi;
5. Algoritmning dastur shaklida ifodalanishi.

1. Ijrochining ko‘rsatmalar sistemasi faqat {5 ni qo‘sh; 3 ni ayir} ko‘rsatmalaridan iborat. Bu ijrochi 0 sonidan 11 sonini hosil qilishi uchun algoritm tuzing.

2. Berilgan ikki natural sonning eng kichik umumiy karralisini topish algoritmini tuzing.

1) boshlansin;
2) M va N sonlari qiymati aniqlansin;
3) M va N sonlar ko‘paytmasi EKUK deb olinsin;
4) agar N = M bo‘lsa, N soni EKUB deb olinsin va 7)-bandga
o‘tilsin;
5) N va M sonlarning kattasi aniqlansin va o‘zi bilan kichik sonning ayirmasiga teng deb olinsin;
6) 4)-bandga o ‘tilsin;
7) EKUK ni EKUB ga nisbati EKUK deb olinsin;
8) javob sifatida EKUK yozilsin;
9) tugallansin.

3. Uchta tanga berilgan. Ulardan biri soxta va faqat og‘irligi bilan farqlanadi (aniq og‘ir yoki yengilligi ham ma’lum emas). Tortish uchun ikki pallali tarozi o‘lchov toshlarisiz berilgan. Eng kam tortish yordamida soxta tangani aniqlash algoritmini tuzing. Avval tangalarni A, B, D kabi nomlaymiz.

1) Agar A va B tangalar taroziga qo‘yilganda og‘irligi teng bo‘lsa, u holda D tanga soxta.
2) Agar A va D tangalar taroziga qo‘yilganda og‘irligi teng bo‘lsa, u holda B tanga soxta.
3) Aks holda A tanga soxta. Demak, soxta tangani ikki marta tortishda aniqlash mumkin.

4. To‘g’ri burchakli uchburchakni yuzini hisoblash algoritmi.

1) boshlansin;
2) to‘g’ri burchakli uchburchakni tomonlari a va b ning qiymatlari aniqlansin;
3) S = (a*b)/2;

4) S qiymat chop etilsin;
5) tugatilsin

5. Algoritm natijasida
y=x-7;
x=3*(y+1);
y=x+y;
y ni chiqarish;
y o‘zgaruvchining qiymati 15 ga teng bo‘ldi. x ni boshlang’ich qiymatini toping.

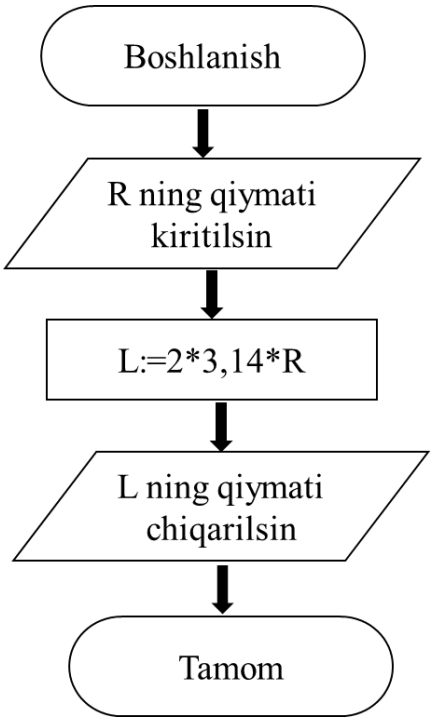
6. Algoritm natijasida
a) x=l;
b) agar x > 2 bo‘lsa, u holda x=x+l va d-bandga o‘tilsin, aks holda c-bandga o'tilsin;
c) x=x*x-4;
d) natija x yozilsin;
e) tugallansin.

XX asrning 70-yillarida golland olimi Edsger Deykstra (1930–2002) har qanday algoritm uning nima maqsadda tuzilganligi va murakkabligidan qat’i nazar, uchta: ketma-ketlik, tarmoqlanish va takrorlanish algoritmik konstruksiyadan foydalanilgan holda yozilishi mumkinligi haqidagi g‘oyani ilgari surdi va to‘liq asoslab berdi.

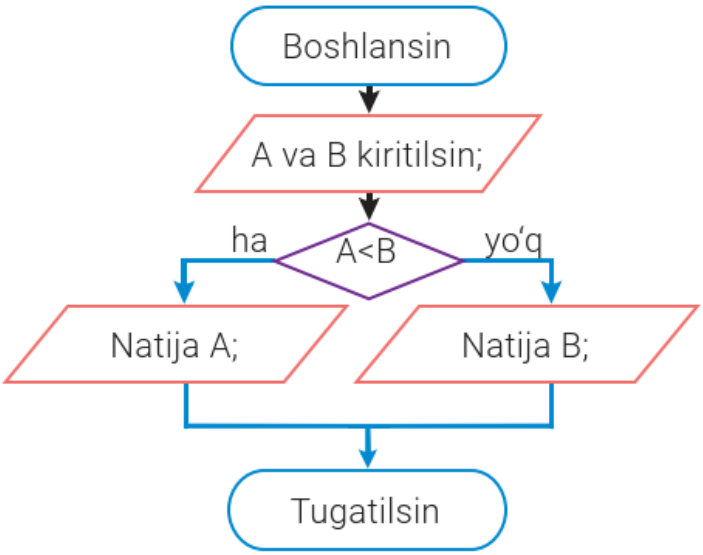
Har qanday algoritm mantiqiy tuzilishga, ya’ni bajarilish tartibiga qarab uch asosiy turga bo'linadi: chiziqli (ergashish), tarmoqlanuvchi va takrorlanuvchi.

Barcha ko‘rsatmalari hech qanday shartsiz, faqat ketma-ket bajariladigan jarayonlarga chiziqli algoritmlar deyiladi.

Misol: Aylana uzunligini topuvchi algoritm tuzing.

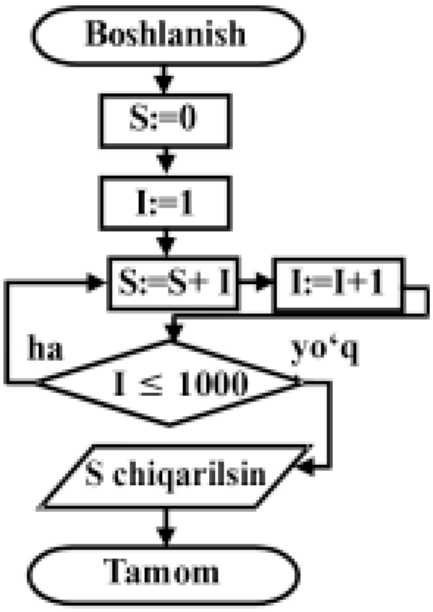


Tarmoqlanuvchi algoritmlar. Shartga muvofiq bajariladigan ko‘rsatmalar ishtirok etgan algoritmlar tarmoqlanuvchi algoritmlar deb ataladi. Algoritmlarning bu turi hayotimizda har kuni va har qadamda uchraydi. Eshikdan chiqishimiz eshik ochiq yoki yopiqligiga, ko'chaga kiyinib chiqishimiz ob-havoga, biror joyga borish uchun transport vositasini tanlashimiz to'lash imkonimiz bo'lgan pulga bog'liqdir.

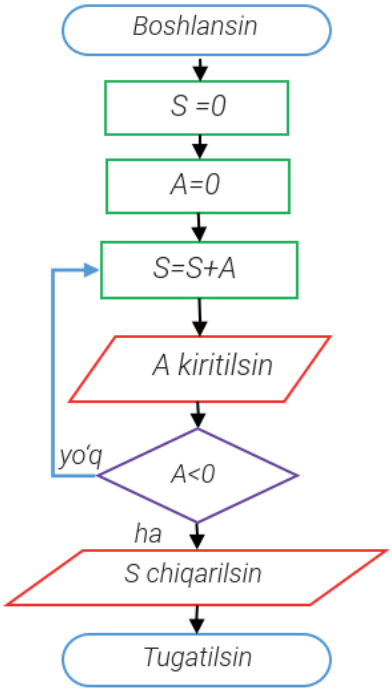


Takrorlanuvchi algoritmlar. Masalan, darslarning har hafta takrorlanishi, har kuni nonushta qilish yoki o‘qishga borish va hokazo. Ko‘rsatmalari takroriy bajariladigan algoritmlar takrorlanuvchi algoritmlar deb ataladi.

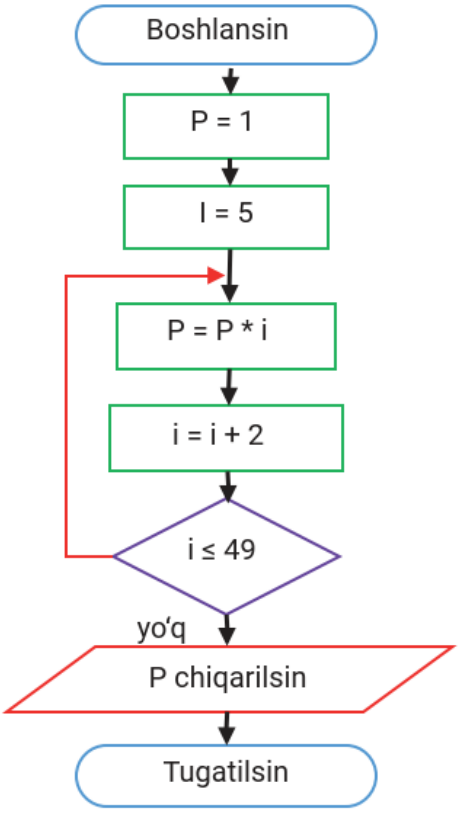
1-algoritm: 1 dan 1000 gacha bo‘lgan sonlar yig’indisini, ya’ni S=1+2+3+…+1000 ni hisoblavchi algoritm tuzamiz.



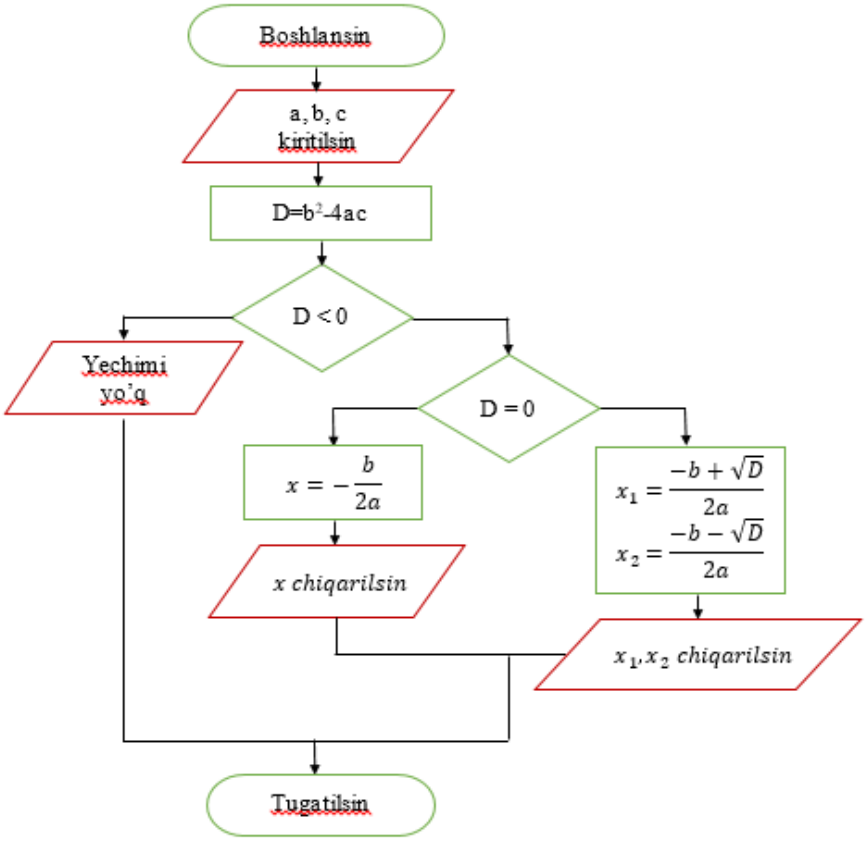
2-algoritm: Tasavvur qiling, klaviaturadan sonlar (1, 6, 8, 2, –6, 76, 1, –5) kiritilmoqda. Birinchi kiritilgan manfiy son (–6) gacha kiritilgan sonlar (1, 6, 8, 2) yig‘indisini hisoblash algoritmini tuzamiz.



3-algoritm: 5 dan 49 gacha bo‘lgan toq sonlar ko‘paytmasini, ya’ni P=5*7*9*…*49 ni hisoblash algoritmini tuzing.



4-algoritm: $ax^2 + bx + c = 0$ ko‘rinishidagi kvadrat tenglamaning yechimlarini topish algoritmini mos blok-sxema usulida tuzamiz.



SAVOL VA TOPSHIRIQLAR

1. Algoritm turlari va ularning vazifalari?
2. Algoritm xossalari?
3. Tarmoqlanuvchi algoritmlarga hayotdan misollar keltiring?
4. Takrorlanuvchi algoritmlarga hayotdan misollar keltiring?
5. So‘zlar yordamida algoritmlarni tasvirlashga misollar keltiring.

1.3. Saralash algoritmlari

Saralash - tartiblash (Sorting Algorithms) deb, berilgan obyektlar ketma-ketligini ma`lum mantiqiy tartibda qayta joylashtirish jarayoniga aytiladi. Saralash bir necha ko‘rsatkichlarga bog`liq bo‘lishi mumkin. Misol uchun maktab jurnalida o'quvchilar familiyasi alifbo tartibiga ko'ra saralangan bo'ladi. Masalan bizga sonlar qatori berilgan: 8, 23, 0, -50, 100 Bu qatorni kichigidan kattasiga qarab yoki kattasidan kichigiga qarab saralashimiz mumkin. Bu saralashni amalga oshirish jarayoni Saralash algoritmi deyiladi. Saralash jarayoni taqqoslashga asoslangan jarayon hisoblanadi. Yuqoridagi sonli qatorni kichigidan kattasiga qarab tartiblaganimizda -50, 0, 8, 23, 100 ko'rinishiga keladi. Biz buni qanday amalga oshirdik. Bunda har xil usuldan foydalanish mumkin va mana shu algoritm turlaridir Biz algoritmlardan bittasidan foydalanib yuqoridagi sonli qatorni tartiblaymiz. Avval, sonli qatordan eng kichigini topamiz va uni ro'yxatnin g boshiga qo'yamiz. Har bir sonni boshqasi bilan solishtirib chiqamiz. Agar son o'zidan keyingi sondan kichik bo'lsa, son shu joyida qoladi, agar katta bo'lsa sonlarning o'rnini almashtiramiz.

Saralash asosan ro'yxat, massiv elementlarida amalga oshiriladi. Masalan sizning sinfingizda 5 ta o'quvchi bor. Ularni familiyasini alifbo tartibida saralash mumkin.

Sonlar berilishi: 23, 54, 3, 22, 1, 45;

Eng kattasini boshiga o‘tkazamiz: 23, 3, 22, 1, 45, 54;(54 soni har bir son bilan solishtirilib eng katta ekani aniqlandi, 45 esa o‘z o‘rnida turipti) Shu tartibni davom ettiramiz: 3, 22, 1, 23, 45, 54; (23 undan keyinda turuvchi eng katta son) Yuqoridagi amalni yana davom ettiramiz: 3, 1, 22, 23, 45, 54;(22 esa davomchi) Oxirgi marta almashtirishimiz quyidagi natijani beradi: 1, 3, 22, 23, 45, 54; (1 eng kichigi) Saralangan tartib quyidagi holatga keldi: 1, 3, 22, 23, 45, 54;

Bizning miyamiz o‘zi optimal deb bilgan yo‘nalishdan ketadi va biz uchun faqat bitta saralash algoritmi mavjud. Ammo dasturlashda bunday deb bo‘lmaydi. Dasturlashga talab ortib, bu soha rivojlanib borgani sari unda bir qator sohalardagi kabi tezlikni oshirish muammosi paydo bo‘ladi. Chunki ilk kompyuter tizimlarida kompyuter tizimining 30% tezligi, operativ xotirasi saralashga sarflanar edi. Shu o‘rinda savol tug`iladi, operatsion tizimlarda ham saralashdan foydalaniladimi? Albatta ha! Fikrimiz isbotini hozirda keng foydalaniladigan Total Commander dasturi isbotlaydi. Unda bir necha xil saralash mavjud: fayl turi, nomi, o‘zgartirilgan sanasi va o‘lchami. Har birini o‘sish yoki kamayish tartibida saralash mumkin. Ha aytgancha, hozirgi tizimlar 30% emas anchagina kamroq tezlik va xotira sarflashadi. Chunki tezlik masalasi tobora yuqori cho‘qqiga chiqayotgan va ishlanayotgan ma`lumotlar o‘lchami oshib borayotgan bir paytda sekin ishlovchi algoritmlardan foydalanish kulguli. Ma`lumotlar o‘lchamlari esa juda katta, shu sababli ularni aniq va tez saralashga ehtiyoj mavjud. Buni amalga oshirish uchun esa yangi algoritmlarga ehtiyoj tug`ila boshladi. Buni yechimi sifatida bir necha turdagi algoritmlardan foydalaniladi.

SAVOL VA TOPSHIRIQLAR

1. Saralash algoritmlariga hayotdan misollar keltiring.
2. Algoritm turlarini sanab bering.
3. 8 ta turli son berilgan. Ular orasidagi musbat sonlar ko‘paytmasini hisoblash algoritmini tuzing.
4. Sinfda 25 nafar o‘quvchi bor. Qancha o‘quvchi informatikadan “a’lo” baho olganligini aniqlash algoritmi tuzilsin.
5. 3 xil takrorlanish tuzilmalaridan foydalanib, 1 dan 100 gacha bo‘lgan sonlar yig‘indisini hisoblash algoritmini tuzing.

2. Zamonaviy dasturlash tillari

2.1. Dasturlash tillari va ularning klassifikatsiyasi

Kompyuter dasturi – biror masalani yechish uchun kompyuter tomonidan ijro etilishi lozim bo‘lgan buyruqlarning izchil to‘plami.

Dasturlash – kompyuter uchun dastur tuzish jarayoni.

Dasturchi – dastur tuzuvchi shaxs.

Dasturlash tili – inson va kompyuter o‘rtasidagi rasmiy aloqa tili. Algoritmlash tillari yaratilishi bo‘yicha uchta turga ajratiladi:

-quyi darajadagi;
-o‘rta darajadagi;
-yuqori darajadagi.

Ma’lumki, ma’lum bir masalani yechish uchun buyruqlar ketma-ketligi ya’ni, algoritm algoritmlash tilida yozilayotganda kamroq buyruqlardan foydalanilsa, bunday tillar darajasi yuqoriroq hisoblanadi. Quyi darajadagi algoritmlash tillari bevosita kompyuter qurilmalari bilan bog‘liq bo‘lib, buyruqlar ularning kodlari bilan yoziladi. Bu kabi buyruqlardan tashkil topgan algoritmlar katta hajmli bo‘lib, ularni taxrirlash katta mehnat talab qiladi. Dastlabki kompyuterlar(ENIAK, MESM va boshqalar) ana shunday tillarda ishlagan. O‘rta darajadagi algoritmlash tillari buyruqlarida faqat raqamlar emas, balki insonlar tushunadigan ba’zi so‘zlar ishlatila boshlandi(Assembler).

Yuqori darajadagi algoritmlash tillari quyidagicha bosqichlarga bo‘linadi:

- Algoritmik(Basic, Pascal, C va b.)
- Mantiqiy(Prolog, Lisp va b.)
- Obyektga mo‘ljallangan(Object Pascal, PYTHON, Java va b.)

Algoritmlash tillarida yaratilgan algoritmlar mashina tiliga translyatorlar yordamida o‘tkaziladi. Translyator(translator-tarjimon) biror bir algoritmlash tilida yozilgan algoritmni mashina tiliga tarjima qiladi.

Translyatorlar ikki turda bo‘ladi:

- Kompilyatorlar (compiler-yig‘uvchi) biror bir algoritmlash tilida yozilgan algoritmni mashina tiliga to‘liq o‘qib olib tarjima qiladi;

- Interpretatorlar (interpreter - izohlovchi, og‘zaki tarjimon) biror bir algoritmlash tilida yozilgan algoritmni mashina tiliga satrma - satr tarjima qiladi.

Kompyuterning elektron qurilmalari ishlashning fizik tamoyillari shundan iboratki, kompyuter faqat bir va noldan iborat buyruqlarni — kuchlanishning pasayishi ketma-ketligini, ya’ni mashina kodini idrok eta oladi. Kompyuterlar rivojlanishining dastlabki bosqichida odam kompyuterga tushunarli tilda, mashina kodlarida dasturlar tuzishi kerak edi. Har bir ko‘rsatma birliklar va nollarning turli kombinatsiyasi sifatida ifodalangan opkod hamda operand manzillaridan iborat edi.

Shunday qilib, protsessor uchun har qanday dastur oʻsha paytda birlar va nollar ketma-ketligi sifatida qaragan.

Keyinchalik kompyuter bilan muloqot qilish amaliyoti shuni koʻrsatdiki, bunday tilni oʻzlashtirish qiyin va noqulaydir. Uni ishlatganda 1 yoki 0 ni notoʻgʻri ketma-ketlikda yozib koʻplab xatolarga yoʻl qoʻyish ehtimoli juda yuqori edi. Dasturni boshqarish juda qiyin boʻlgan. Bundan tashqari, mashina kodlarida dasturlashda kompyuterning ichki tuzilishini, har bir blokning ishlash prinsipini yaxshi bilish kerak edi va bunday tildagi eng yomon narsa shundaki, bu tildagi dasturlar — birlar va nollarning juda uzun ketma-ketligi mashinaga bogʻliq, yaʼni har bir kompyuter uchun oʻz dasturini tuzish kerak edi va mashina kodlarida dasturlash juda koʻp narsa: vaqt, ish va dasturchining eʼtiborini oshirishni talab etardi.

Tez orada mashina kodini yaratish jarayonini avtomatlashtirish mumkinligi maʼlum boʻldi. 1950-yildan boshlab dasturlarni yozish uchun mnemonik til — Assembler tilidan foydalanila boshlandi. Assembler tili mashina kodini inson uchun qulayroq shaklda koʻrsatishga imkon berdi: buyruqlar va bu buyruqlar bajariladigan obyektlarni belgilash uchun buyruqning mohiyatini aks ettiruvchi ikkilik kodlar oʻrniga harflar yoki qisqartirilgan maxsus soʻzlar qoʻllanilgan. Masalan, assembler tilida ikkita raqamni qoʻshish boʻyicha koʻrsatma add soʻzi bilan ifodalanadi, uning mashina kodi 000010 tarzida boʻladi.

Assembler quyi darajadagi dasturlash tilidir. Quyi darajadagi dasturlash tili, bu muayyan turdagi protsessorga yoʻnaltirilgan va uning xususiyatlarini hisobga oladigan dasturlash tili demakdir. Bunday holda, „quyi“ „yomon“ degani emas, balki bu shuni anglatadiki, til operatorlari mashina kodiga yaqin va maxsus protsessor koʻrsatmalariga qaratilgan boʻladi. Assemblar tilining paydo boʻlishi dasturchilarning hayotini sezilarli darajada osonlashtirdi, chunki endi ular koʻzda miltillovchi nollar va birlar oʻrniga oddiy tilga yaqin belgilardan iborat buyruqlar bilan dastur yozishlari mumkin edi. Oʻsha vaqt uchun bu til innovatsiya edi va mashhur edi, chunki u kichik dasturlarni yozishga imkon berardi, bu esa oʻsha davr mashinalari uchun muhim mezon sanalgan.

Ammo u orqali yirik dasturiy taʼminotlar ishlab chiqish murakkabligi boʻlganligi bois uchinchi avlod tillari hisoblanmish yuqori darajadagi tillarning paydo boʻlishiga olib keldi. Ammo

assemblerdan foydalanish shu bilan tugamadi, u bugungi kungacha tor doiralarda mashhur bo‘lib kelmoqda. Hozirgi vaqtda u dasturlarning alohida qismlarini yozishda yoki ba’zan dasturlarning o‘zini yozishda qo‘llaniladi. Misol tariqasida, drayverlar, o‘yinlar va operatsion tizimlar yuklagichi (bootloader)ni yozishda assemblerga murojaat etiladi. Shuni unutmaslik kerakki, bu til hakerlar orasida ham mashhurdir, chunki bu tilda yozilgan dasturning tezligi yuqori darajadagi dasturlash tilida yozilgan dastur tezligidan ancha yuqori bo‘lishidadir. Buning sababi shundaki, assemblerda yozilgan dastur hajmi juda kichik bo‘ladi. Antivirus ishlab chiquvchilari o‘z dasturlarining ba’zi modullarida assemblerdan ham foydalanadilar.

SAVOL VA TOPSHIRIQLAR

1. Translyatorlar turlarini tushuntiring.
2. Dastur nima?
3. Yuqori darajadagi algoritmlash tillariga qaysilar kiradi.
4. Algoritmlash tillari yaratilishi bo‘yicha necha turga ajratiladi.
5. Dasturlash nima?
6. Dasturlash tili nima?
7. Dasturlash muhit nima?

2.1. Yuqori darajali dasturlash tillari

1950-yillarning o‘rtalari dasturlashning jadal rivojlanishi bilan tavsiflanadi. Mashina kodlarida dasturlashning roli pasaya boshladi, mashinalar va dasturchilar o‘rtasida vositachi bo‘lgan yangi turdagi dasturlash tillari paydo bo‘la boshladi. Dasturlash tillarining ikkinchi va uchinchi avlodlari davri boshlandi. XX asrning 50-yillari o‘rtalariga kelib, birinchi yuqori darajadagi dasturlash tillari (high-level programming languages)ni yaratishga kirishildi. Ushbu tillar kompyuterning ma’lum bir turiga (mashinadan mustaqil) bog‘lanmagan edi. Ularning har biri uchun o‘z kompilyatorlari ishlab chiqilgandi. Kompilyatsiya — yuqori darajadagi manba tilda yozilgan dasturni mashina kodiga yaqin quyi darajali tildagi ekvivalent dastur (absolyut kod, obyekt moduli, ba’zan assembler tili)ga o‘girishni anglatadi.

Birinchi yuqori darajadagi dasturlash tili 1942-1945-yillarda Konrad Suze tomonidan yaratilgan en:Plankalkül dasturlash tili edi.

Buyruqlarni kompyuterga yetkazish uchun mo‘ljallangan birinchi ishlaydigan dasturlash tillari 1950-yillarning boshlarida yozilgan. 1949-yilda taklif qilingan Jon Mauchlyning en:Short code elektron kompyuter uchun yaratilgan birinchi yuqori darajadagi tillardan biri edi. Mashina kodidan farqli o‘laroq, Short code matematik ifodalarni tushunarli shaklda ifodalaydi. Biroq, dastur har safar ishga tushirilganda mashina kodiga tarjima qilinishi kerak edi, bu jarayon ekvivalent mashina kodini ishlatishdan ancha sekinroq davom etardi.

Quyi darajadagi dasturlash tillari kompyuter qurilmalari bilan bevosita bog’liq bo‘lib, buyruqlar maxsus raqamlar yordamida yoziladi. Bu kabi buyruqlardan tashkil topgan dastur hajmi kata bo‘ladi va ularni tahrirlash ancha mehnat talab qiladi. Dastlabki elektron hisoblash mashinalarida (“ENIAC”, “МЭСМ”) masalalarni yechish uchun ana shunday buyruqlar yordamida dasturlar tuzilgan. Dastur tuzishni osonlashtirish maqsadida inson tiliga yaqin bo‘lgan buyruqlar tizimini qo‘llash masalasi qo‘yildi va hal etildi. Bu kabi dasturlash tillari o‘rta darajadagi dasturlash tillari (ba’zan assemblerlar) deb yuritila boshlandi. Bunday tillarga AVTOKOD-BEMSH, AVTOKOD-MADLEN va boshqalar kiradi.

Ular BESM-6, Minsk-22, IBM-360 elektron hisoblash mashinalarida qo‘llaniladi.

Keyingi yillarda juda ko‘p yuqori darajadagi dasturlash tillari ishlab chiqarilgan bo‘lib, ular qatoriga Object Pascal, Ada, KARAT, C++, Delphi, Visual Basic, C#, Java, JavaScript, PHP, Python va boshqalar.

Hozirgi kunda ishlab chiqilayotgan dasturlash tillari biror yo‘nalishdagi masalalarni hal qilishga mo‘ljallangan bo‘lib, ularni obyektga yo‘naltirilgan dasturlash tillari deb atashadi.

SAVOL VA TOPSHIRIQLAR

1. Yuqori darajadagi dasturlash tillariga qaysi dasturlar kiradi?
2. Quyi darajadagi dasturlash tillariga qaysi dasturlar kiradi?
3. O‘rta darajadagi dasturlash tillariga qaysi dasturlar kiradi?

2.2. Python dasturlash tili. IDLE bilan tanishish

Dasturlash tillari juda ko‘p bo‘lib, ularning har biridan o‘ziga xos masalalarni yechishda foydalanish mumkin. Aksariyat dasturlash tillari,

xususan, C++, Pascal, Java, Python va boshqalar integrallashgan dasturlash muhiti (IDE)ga ega.

IDE (Integrated Development Environment – integrallashgan dasturlash muhiti) – dasturiy ta’minot yaratish uchun dasturiy vositalar majmui. **IDE** (Integrated Development Environment – integrallashgan dasturlash muhiti) – dasturchilar uchun qo‘shimcha funksiyalarga ega bo‘lgan maxsus matn muharriri.

IDLE (Integrated Development and Learning Environment – integrallashgan dasturlash va o‘rganish muhiti) – Python tilini o‘rganish uchun taqdim etilgan IDE.

IDLE oynasining ikki xil muhiti mavjud bo‘lib, ular interfaol muhit va dasturlash muhiti deb nomlanadi.

IDLEning **interfaol muhiti** (konsol deb ham yuritiladi) – dastur kodini kiritib, natijasini darhol ko‘rish mumkin bo‘lgan oyna.

IDLEning **dasturlash muhiti** – dastur kodini kiritish, tahrirlash va ishga tushirish mumkin bo‘lgan oyna. Dastur natijasi interfaol muhitda chiqariladi. **Afzalligi** – kodlar qayta-qayta yozilmaydi.

Kamchiligi – dastur dastlab saqlanadi, undan keyingina ishga tushiriladi.

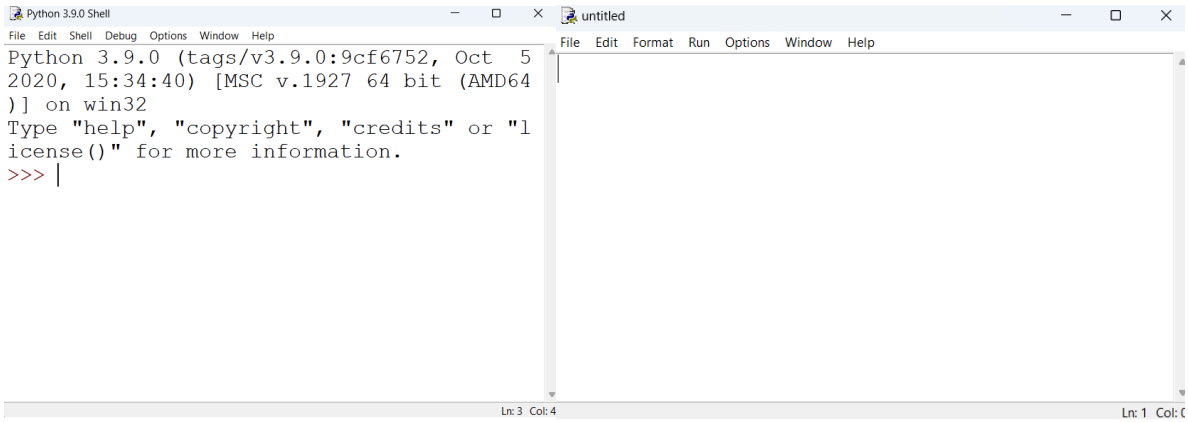
PEP8 (Python Enhanced Proposal - Python tilini takomillashtirish bo'yicha takliflar) ham ishlab chiqilgan. **http://dr.rtm.uz/pep8** sahifa orqali PEP8 ning to'liq tavsifi bilan tanishishingiz mumkin.

Protsessor dasturlash tilida yozilgan dasturni to‘g‘ridan-to‘g‘ri tushunmaydi. Buning uchun dasturni protsessor tiliga tarjima qiladigan (**raqamli ko‘rinishga o‘tkazib beruvchi**) tarjimon tili – **translyatordan** foydalaniladi.

Tarjimon tilining ikkita turi mavjud: **Kompilyator** va **Interpretator**.

Kompilyator dasturlash tilida yozilgan dastur kodlarini to‘laligicha o‘qib, mashina kodiga tarjima qiladi va tarjima natijalarini bajariladigan yaxlit bitta faylga yig‘adi.

Interpretator dasturlash tilida yozilgan kodni bosqichma-bosqich mashina kodiga aylantirib, tahlil qiladi va berilgan buyruqlarni ketma-ketlikda bajaradi. Agar xatolik sodir bo‘lsa, o‘sha paytning o‘zida xabar beradi



Van Rossum tomonidan Python 1.2 versiyasi 1995 yili matematika va informatika laboratoriya markazida ishlayotgan paytda ishlab chiqarilgan. Python dasturlash tili mukammal darajada ishlab chiqilgan dasturlash tili bo‘lib u insoniyat oldidagi muammolarni hal qilish uchun juda mos til hisoblanadi. Python dasturlash tili, dasturlash tillarining eng keng imkoniyat doirasiga ega hisoblanadi, bu dasturlash tili boshqa dasturiy vositalarni boshqarish va ularning tarkibiy qismlarini mustaqil boshqarishni amalga oshirdi. Aslida, Python ko‘p maqsadli dasturlash tili sifatida o‘rganilishi mumkin, bu dasturlash tili yordamida bir qancha jarayonlarni dasturlash imkoni yaratiladi. Python amaliy dasturiy maxsulotlar, web ilovalar va ilmiy dasturiy maxsulotlar yaratish imkonini beradi. Python tarkibida xotiradan foydalanish va ishlash talablari bo‘yicha cheklovlar mavjud emas, ya’ni imkoniyatlar shu qadar kattaki, boshqa dasturlash tillari kabi ma’lumotlarni e’lon qilish tabaqasi mavjud emas. Bu esa dastur yozish vaqti kamaytiradi va boshqarish qulayligini oshiradi.

Python dasturlash tilini bu qadar keng tarqalishining sababi juda katta miqdordagi yuqori sifatli tayyor bepul tarqatiladigan modullar mavjud va ularni siz dasturning istalgan joyidan foydalanishingiz mumkin. Tayyor modullardan foydalangan holda dasturni tuzish bir qancha optimal hisoblanadi. Dasturlash tili tarkibida fundamental algoritmlar, funksiyalar va modullar tayyor holatga keltirilgan, bunda faqatgina bu algoritm yoki funksiyalarga murojaat qilinsa yetarli siz faqat tegishli qismlarni tanlashingiz va ularni bir joyga to‘plashingiz kerak.

Modullar har bir misolning boshida mavjud bo‘lgan import buyrug‘i yordamida biriktiriladi. Ko‘p ishlatiladigan modullar ikkita asosiy qismga bo‘lingan:

- Python interpretatori bilan ta'minlangan standart kutubxonaning modullari (ushbu modullar doimo dastur bilan birga aktivlashadi);
- Tashqi vazifa bajaruvchi modullar, bu modullar alohida dastur tarkibiga o‘rnatish orqali amalga oshiriladi.

Python dasturlash tilining web dasturlash sohasiga ham to‘g‘ridan to‘g‘ri qo‘llanilishi mumkin. Python an'anaviy ravishda oddiy va murakkab strukturali saytlarni yaratish uchun foydalaniladi.

Python dasturlash tilining eng muhim afzalliklaridan biri shundaki, uning barcha amaliy kutubxonalari va qo‘shimcha maxsus modullarining rivojlanish muhiti bepul tarqatiladi. Bu esa Python dasturlash tilini rivojlantirish vositasi bo‘lishi mumkinligini anglatadi.

Python dasturlash tili dasturlashning quyidagi sohalarrida qo‘llaniladi:

- Tizimli dasturlash;
- Grafik interfeysli dasturlarni ishlab chiqish;
- Dinamik veb-saytlarni yaratish;
- Komponentlarning integratsiyasi;
- Ma'lumotlar bazalari bilan ishlash uchun dasturlarni ishlab chiqish;
- Ilmiy hisoblash uchun dasturlarni ishlab chiqish;
- O‘yinlarni rivojlantirish.

SAVOL VA TOPSHIRIQLAR

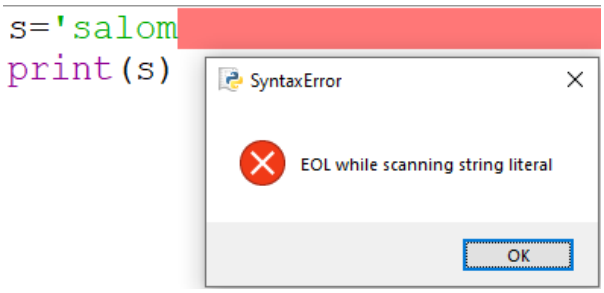
1. IDLEning interfaol muhiti afzalligi?
2. IDLEning dasturlash muhiti kamchiligi.
3. Tarjimon tilining necha turi mavjud.
4. PEP8 nima?

2.3. Python dasturlash tilida xatoliklar

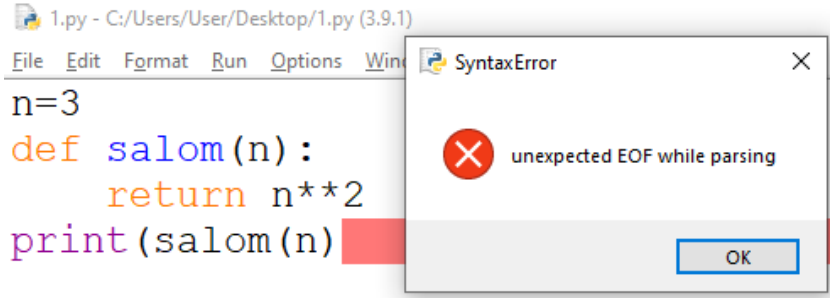
Sintaksis xatoliklari

SyntaxError – xatolik haqidagi xabar ushbu oynada aniqlanadi.

EOL (ingl. **End of line** – **qator yakuni**) xatoligi sintaksis xatolikning bir turi bo‘lib, odatda, qator oxirida qo‘shtirnoq (birtinoq)ni yopish tushirib qoldirilganda yuzaga keladi.

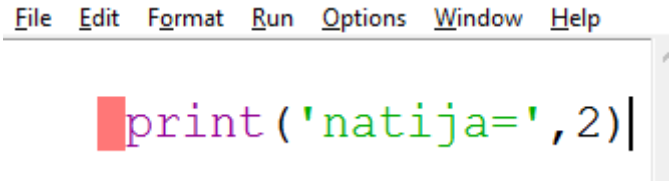


SyntaxError – xatolik haqidagi xabar ushbu oynada aniqlanadi.
EOF (ingl. **End of function** – **funksiya yakuni**) – xatoligi esa funksiya oxirida qavsni yopish tushirib qoldirilganda yuzaga keladi.



Sintaksis xatoliklari

Yana bir eng ko‘p yo‘l qoyiladigan sintaksis xatoliklaridan biri bu – “**Indentation Error**” xatoligidir. Pythonda, vaziyatga qarab, kod qator boshidan joy tashlab yoki joy tashlamasdan yoziladi. Agar asossiz joy tashlansa yoki aksincha, kerakli joy tashlanmasa, “IndentationError” xatoligi yuz beradi.
unexpected indent – dasturda noto‘g’ri joy ajratilganligini bildiradi.

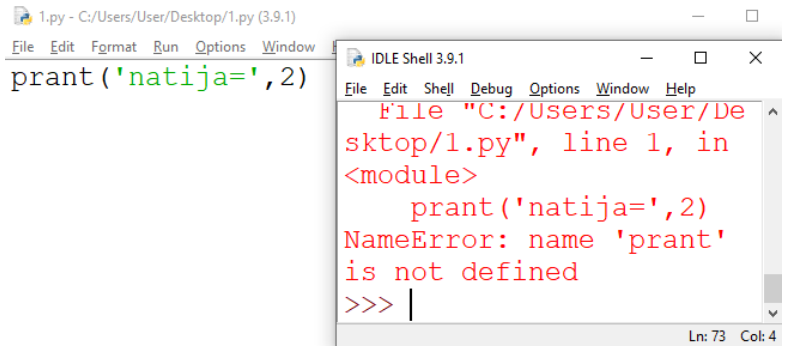


Istisnolar

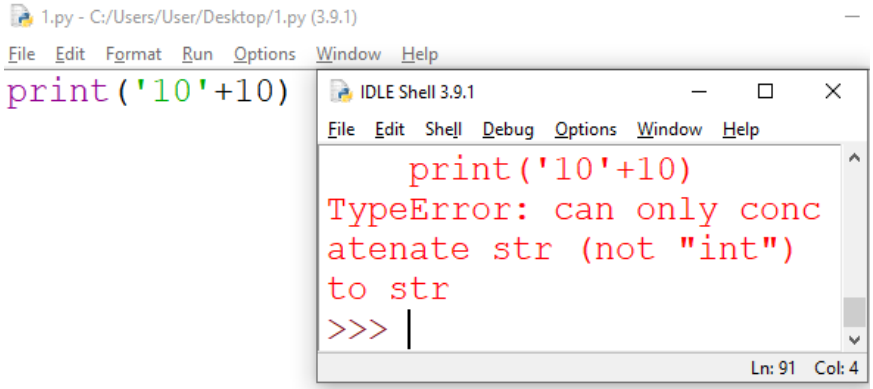
Python qoidalariga ko‘ra, sintaksis xatolari, odatda, **xatolik** deb ataladi. Ammo aksariyat hollarda dastur sintaksis xatosi bo‘lmasa ham, ishga tushirilganidan so‘ng ba’zi xatoliklarga duch keladi. Bunday xatoliklar **istisno** deb ataladi.

NameError – lokal yoki global o‘zgaruvchi, funksiya, obyekt nomi noto‘g‘ri yozilganda yoki mavjud bo‘lmagan o‘zgaruvchi, funksiya yoki obyekt chaqirilganda kelib chiquvchi istisno.

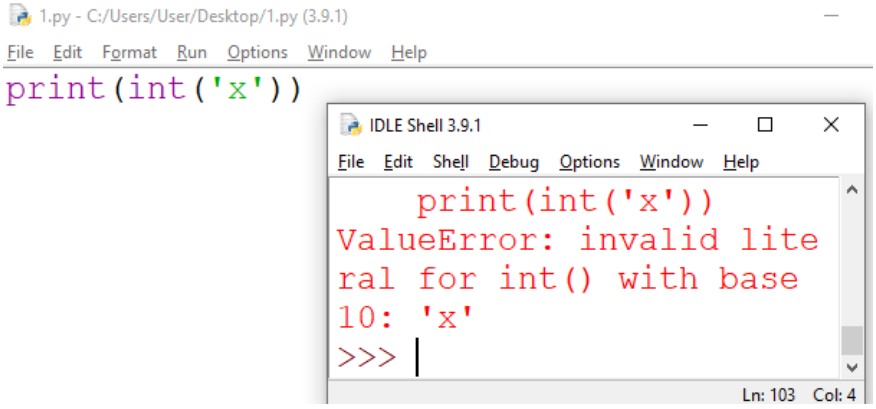
masalan: prant('natija=',2)



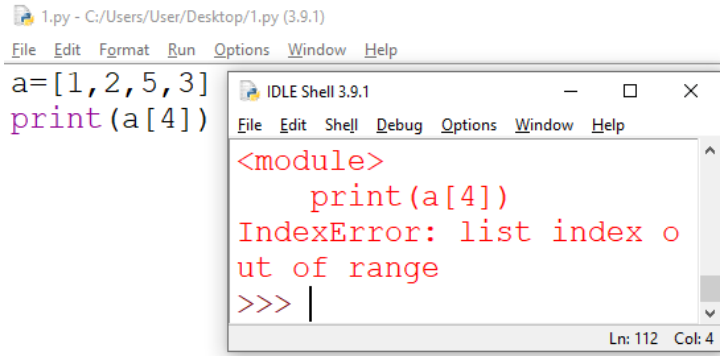
TypeError – funksiya yoki qandaydir amalga noto‘g‘ri ma’lumot yuborilganda yuz beradigan istisno.



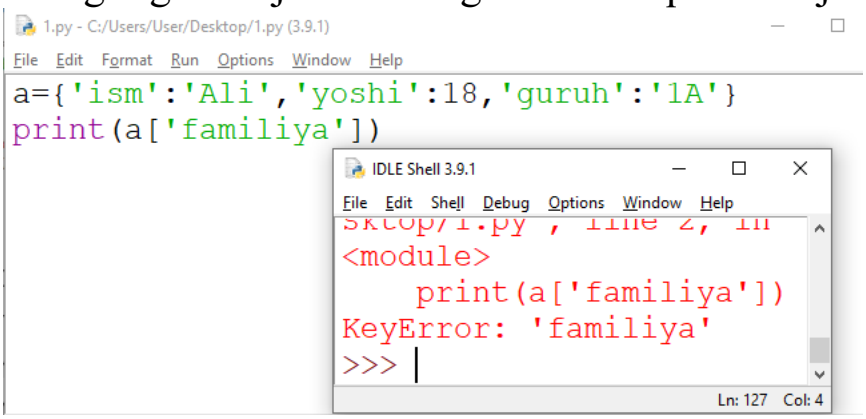
ValueError – funksiya to‘g‘ri turdagi qiymatni yuborishi natijasida yuzaga keladigan istisno. Python tilida “int” turi sonli qiymatni qabul qilishi belgilanganligi uchun, unga satr turidagi o‘zgaruvchini yuklash mumkin emasligi haqida xabar chop qilindi.



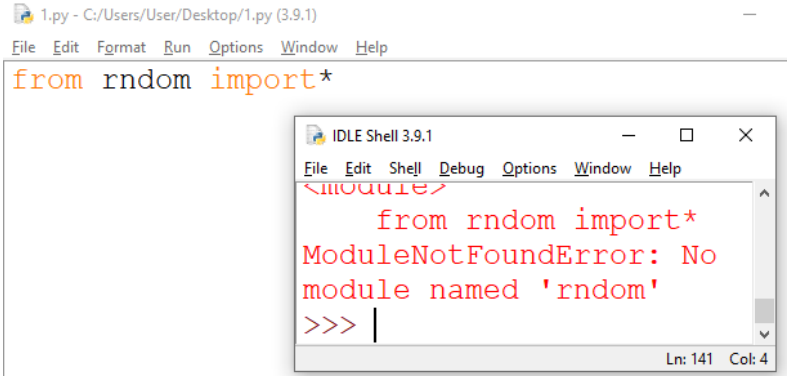
IndexError – ro‘yxat elementlariga murojaat qilishda, elementga ro‘yxat indekslarida mavjud bo‘lmagan indeks orqali murojaat qilishda yuzaga keladi.



KeyError – lug‘atga mavjud bo‘lmagan kalit orqali murojaat qilish.



ModuleNotFoundError – mavjud bo‘lmagan modulni yuklashda yuz beradigan istisno.



Istisno	Ta’rifi
AttributeError	Atribut o‘zlashtirish xatoligi yuz berganda chaqiriladi.
MemoryError	Amallarni bajarish uchun xotira yetmaganda chaqiriladi.
OSError	Sistemaviy amalda xatolik bo‘lganida chaqiriladi.
OverflowError	Arifmetik amalning natijasi chop etish uchun juda katta bo‘lganida chaqiriladi.
UnicodeError	Unicode bilan bog‘liq kodlash yoki dekodlash xatosi paydo bo‘lganda chaqiriladi.
UnboundLocalError	Funksiya yoki metodda hech qanday qiymat qabul qilmagan lokal o‘zgaruvchiga murojaat qilinganida paydo bo‘ladi.
FloatingPointError	Qo‘zg‘aluvchi nuqtali haqiqiy sonlar ustida amallar bajarishda xatolik yuz berganda chaqiriladi.
RuntimeError	Yuz bergan xato bironta xatolik toifasiga tegishli bo‘lmaganida chaqiriladi.

SAVOL VA TOPSHIRIQLAR

1. Python dasturida nuqtali vergul xato ishlatilsa qanday xatolik sodir bo‘ladi.
2. invalid syntax qanday xatolik.
3. UnboundLocalError qanday xatolik.
4. TypeError qanday xatolik.
5. MemoryError qanday xatolik.

3. Python dasturlash tili va sintaksisi
3.1. Python tilida o‘zgaruvchilar va operatorlar

O‘zgaruvchilar – o‘z qiymati va turiga ega kattalik, o‘zida qiymatlarni saqlaydigan kompyuter xotirasidagi yacheyka nomi.

Doimiy (o‘zgarmas) – faqat o‘qish uchun mo‘ljallangan qiymatlarni saqlovchi kompyuter xotirasidagi yacheyka nomi. Doimiy (konstanta)larni ifodalash uchun faqat bosh harflardan foydalaniladi. Masalan, PI = 3.1415

Identifikatorlar – o‘zgaruvchilar, doimiylar, funksiyalar, protseduralar, modullar, dasturlarning umumiy nomi.

Identifikatorlar harf va raqamlar kombinatsiyasidan tarkib topadi. Masalan, a25, b5c88, sonlar to‘plami va boshqalar. Pythonda katta va kichik harflar bir-biridan farq qiladi.

print() – operatori o‘zgaruvchi qiymatini ekranga chiqaradi.

Dastur:

```
a=15
print(A)
```

Odatda, dasturlar kiritilgan ma’lumotlarni qabul qilish, qayta ishlash, shuningdek, natijani ekranga chiqarish uchun mo‘ljallangan bo‘ladi. Dasturlarni yozishda o‘zida asosiy ma’lumotlarni saqlaydigan o‘zgaruvchi yoki doimiylardan foydalaniladi. O‘zgaruvchilar dastur jarayonida o‘zgarishi mumkin bo‘lgan ma’lumotlarni belgilaydi, doimiydan esa o‘zgarmas ma’lumotlar uchun foydalaniladi. O‘zgaruvchilar va doimiylarni belgilash uchun turli nomlar, ya’ni identifikator (identification)lardan foydalaniladi.

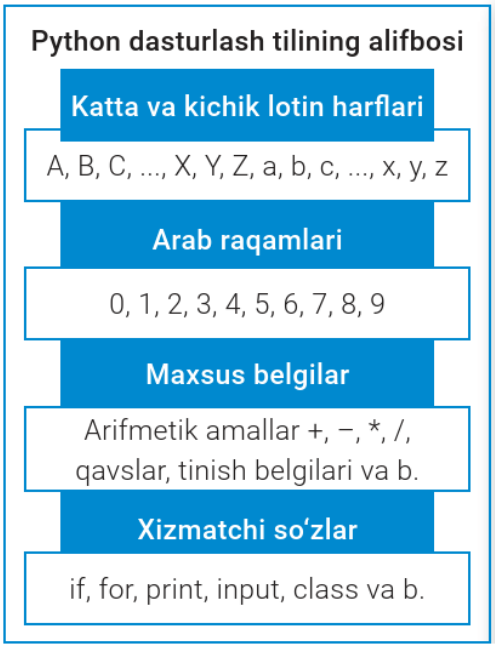
Identifikatorlar harf va raqamlar kombinatsiyasidan tarkib topadi. Masalan, a25, b5c88, sonlar to‘plami va boshqalar. Pythonda katta va kichik harflar bir-biridan farq qiladi.

O‘zgaruvchilarni e’lon qilish qoidalari:

o‘zgaruvchi nomida ixtiyoriy harf yoki raqamdan foydalanish mumkin; katta va kichik harflar bir-biridan farqlanadi. belgi5, Belgi5, BELGI5 nomli o‘zgaruvchilar har xil o‘zgaruvchi nomlaridir, shu sababli kichik harflardan foydalangan ma’qul;

- o‘zgaruvchi nomlarida probellar ishlatish mumkin emas, uning o‘rniga so‘zlar (‘_’) tagchiziq belgisi yordamida bir-biriga bog‘lanadi;

- o‘zgaruvchining nomi raqamdan boshlanishi mumkin emas;
- –, /, # yoki @ belgilarni ishlatish mumkin emas;
- maxsus buyruq nomlarini ishlatish mumkin emas;
- and, as, assert, break, class, continue, def, del, elif, else, except, finally, for, from, global, if, import, in, is, lambda, nonlocal, not, or, pass, print, raise, return, try, while, with, yield kabi xizmatchi so‘zlarni o‘zgaruvchi nomi sifatida ishlatish mumkin emas.



3.1-rasm. Python tili alifbosi.

O‘zgaruvchini e’lon qilish. O‘zgaruvchi “=” belgisi yordamida satr yoki sonni o‘zlashtirishi mumkin va bu o‘zgaruvchining qiymati deb ataladi. O‘zgaruvchiga sonli qiymat berish uchun “=” belgisidan so‘ng son yoziladi. Satrli qiymat berilganda esa “=” belgisidan so‘ng bittalik (' '), ikkitalik (“ ”) qo‘shtirnoq ichida satr yoziladi.

O‘zgaruvchi qiymatini o‘zgartirish. O‘zgaruvchi qiymatini o‘zgartirish uchun unga yangi qiymat berish kifoya.

O‘zgaruvchilarni o‘zaro ishlatish.

Ikkita o‘zgaruvchida birining qiymatini ikkinchisiga o‘zlashtirish uchun «=» belgisidan foydalaniladi.

```
>>> apple = 20
>>> on_ apple = apple
>>> print(apple, on_ apple)
```

Natija: 20, 20

Doimiylar. Doimiy (konstanta)larni ifodalash uchun faqat bosh harflardan foydalaniladi. Masalan, PI = 3.1415

SAVOL VA TOPSHIRIQLAR

1. Python dasturlash tilining alifbosi qanday tuzilmadan iborat?
2. Identifikator nima?
3. O‘zgaruvchan va doimiyning bir-biridan farqli jihati nimada?
4. O‘zgaruvchilarni nomlashda qanday belgilardan foydalanish mumkin emas?
5. Darvozaning eni 4 metr, bo‘yi esa 3 metr:
1) darvoza yuzasi (S)ni hisoblash dasturini tuzing;
2) darvoza qirrasining uzunligi (P)ni hisoblash dasturini tuzing.
6. Ikkita qayiq turg‘un suvda bir-biriga tomon 4 km/soat va 2 km/soat tezlik bilan
suzmoqda. Ular orasidagi masofa 24 km bo‘lsa:
1) ular qancha vaqtdan keyin uchrashadi?
2) qancha vaqtdan keyin ular orasidagi masofa 12 km ni tashkil etadi?
7. Doiraning radiusi 4 metrga teng (PI=3.14):
1) doira yuzini hisoblang;
2) aylana uzunligini hisoblang.

3.2 Python tilida ma’lumot turlari

Ma’lumki, axborotlar matnli, raqamli, ovozli, grafik va boshqa shakllarda uzatilishi mumkin. Bunday axborotlarni dasturlash tillarida qayta ishlash uchun, ularni turlarga ajratish lozim. Dasturlarda foydalaniladigan ma’lumotlar turlari dasturning maqsadiga bog‘liq bo‘ladi: oddiy kalkulyator sonlardan foydalanadi, elektron pochta manzillarini tekshirishga mo‘ljallangan dastur esa matnlar bilan ish ko‘radi. Sonlar natural, butun va haqiqiy sonlarga ajratiladi. Matnli axborotlar esa belgilar yoki satrli ma’lumotlardan iborat bo‘lishi mumkin.

Ma’lumotlar turi – kompyuter xotirasidagi yacheykada saqlanadigan ma’lumotlar shakli.

input (kiruvchi ma’lumot) – ma’lumotlarni kiritadi. input() operatori yordamida kiritilgan ma’lumotlar satrli ko‘rinishda bo‘ladi.

int() – butun sonlar tipi
float() – haqiqiy sonlar tipi
str() – satrli tip
bool() – mantiqiy tip

Ma’lumotlar turi – bu o‘zgaruvchi yoki doimiy qiymatlardagi ma’lumotlar shakli. Ma’lumotlar turi kompyuter xotirasida yetarlicha joyni zaxiraga olib qo‘yish uchun kerak bo‘ladi. Odatda, dasturlash tillarida ma’lumotlar turi o‘zgaruvchi yoki doimiy bilan birga e’lon qilinadi. Python dinamik turlarga ajratuvchi dasturlash tili hisoblanadi. Shu sababli, Pythonda o‘zgaruvchining turi u foydalanayotgan qiymat bo‘yicha belgilanadi, lekin ma’lumot turini boshqa turga o‘zgartirish uchun tur ko‘rsatilishi shart.

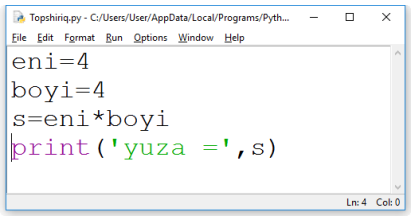
Ma’lumotlar turini o‘zgartirish. O‘zgaruvchi tarkibida ixtiyoriy turdagi ma’lumot saqlanishi mumkin. Ma’lumotlar turini o‘zgartirish uchun mos ma’lumotlar turi buyruqlaridan foydalaniladi. input() operatori yordamida kiritilgan ma’lumotlar satrli ko‘rinishda bo‘ladi.

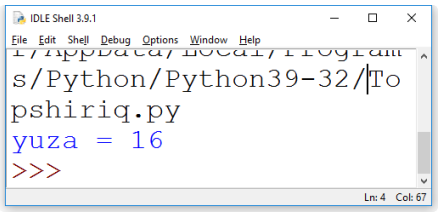
1-Topshiriq. Darvozaning eni 4 metr, bo‘yi esa 3 metr:

a) darvoza yuzasi (S)ni hisoblash dasturini tuzing;

Dastur kodi:

```
eni=4
boyi=4
s=eni*boyi
print('yuza =',s)
```

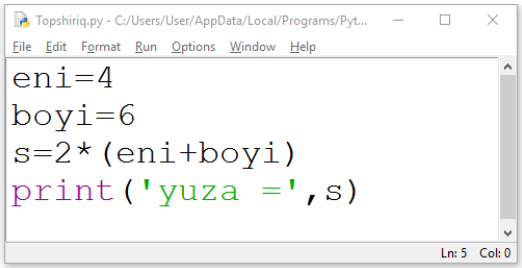


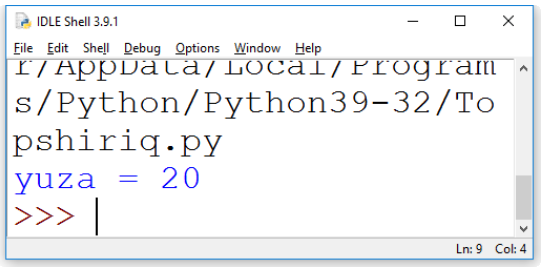


b) darvoza qirrasining uzunligi (P) ni hisoblash dasturini tuzing.

Dastur kodi:

```
eni=4
boyi=6
s=2*(eni+boyi)
print('yuza =',s)
```





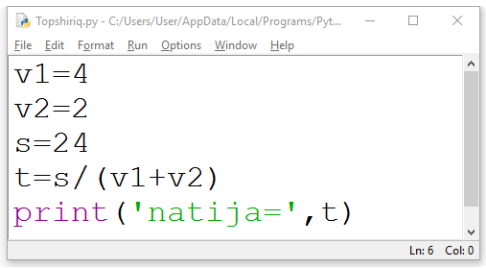
2-Topshiriq. Ikkita qayiq turg‘un suvda bir-biriga tomon 4 km/soat va 2 km/soat tezlik bilan suzmoqda. Ular orasidagi masofa 24 km bo‘lsa:

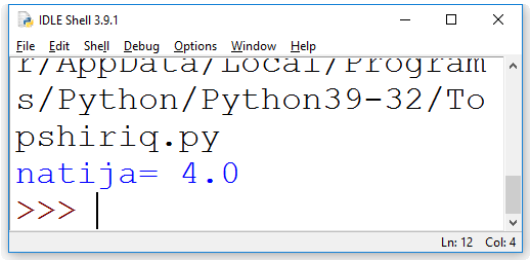
a) ular qancha vaqtdan keyin uchrashadi?

Dastur kodi:

```
v1=4
v2=2
```

```
s=24
t=s/(v1+v2)
print('natija=',t)
```

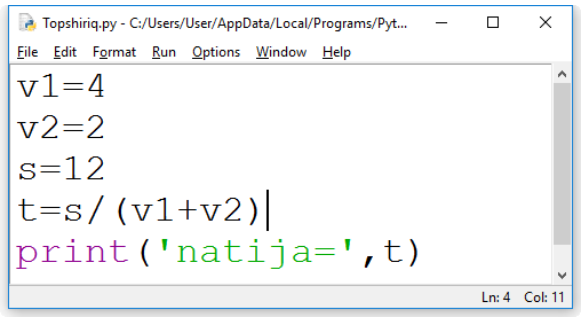


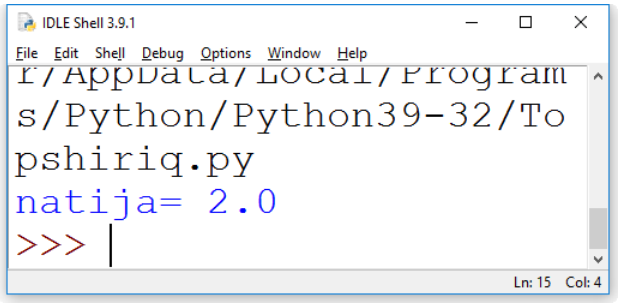


b) qancha vaqtdan keyin ular orasidagi masofa 12 km ni tashkil etadi?

Dastur kodi:

```
v1=4
v2=2
s=12
t=s/(v1+v2)
print('natija=',t)
```



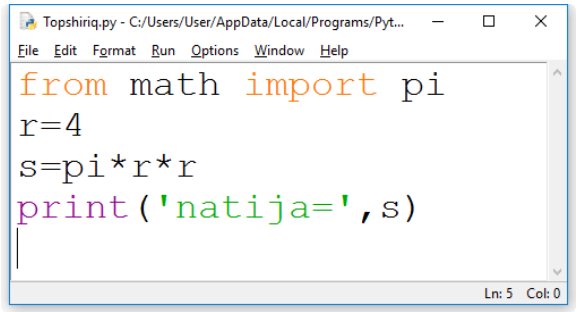


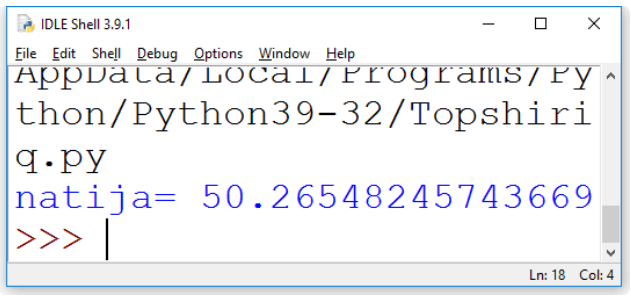
3-Topshiriq. Doiraning radiusi 4 metrga teng (PI=3.14):

a) doira yuzini hisoblang;

Dastur kodi:

```
from math import pi
r=4
s=pi*r*r
print('natija=',s)
```

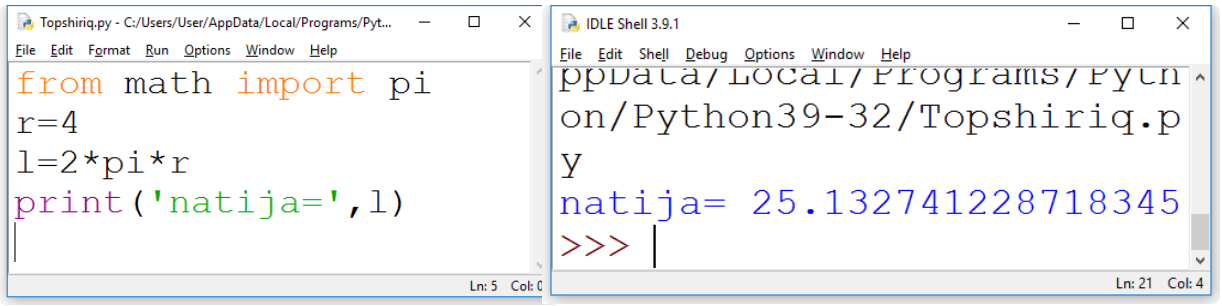




b) aylana uzunligini hisoblang.

Dastur kodi:

```
from math import pi
r=4
l=2*pi*r
print('natija=',l)
```

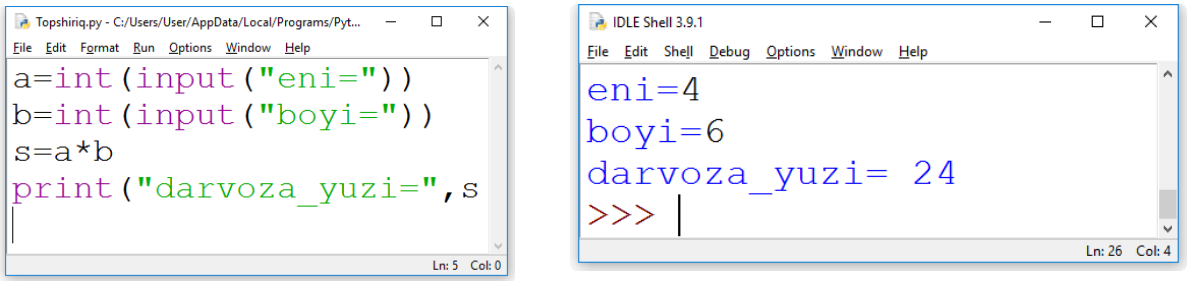


4-Topshiriq. Darvozaning eni va bo‘yi foydalanuvchi tomonidan kiritiladi:

a) darvoza yuzi (S) ni hisoblash dasturini tuzing;

Dastur kodi:

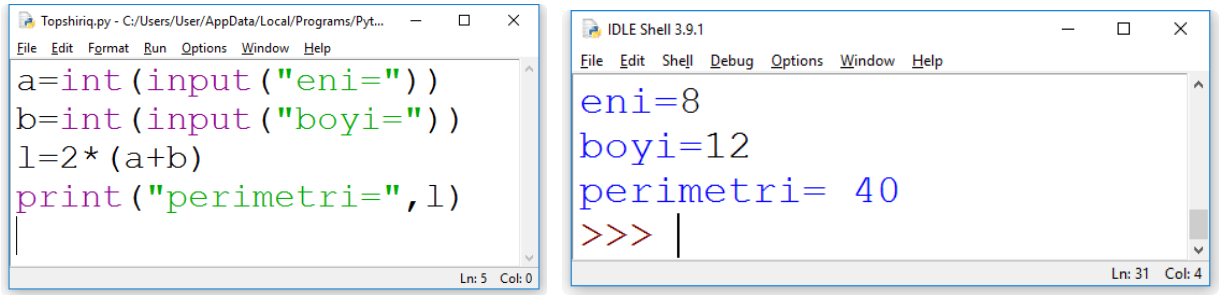
```
a=int(input("eni="))
b=int(input("boyi="))
s=a*b
print("darvoza_yuzi=",s)
```



b) darvoza qirrasining uzunligi (P) ni hisoblash dasturini tuzing.

Dastur kodi:

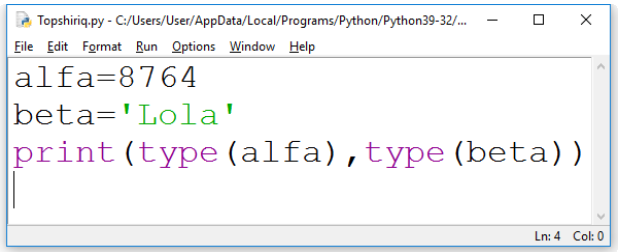
```
a=int(input("eni="))
b=int(input("boyi="))
l=2*(a+b)
print("perimetri=",l)
```

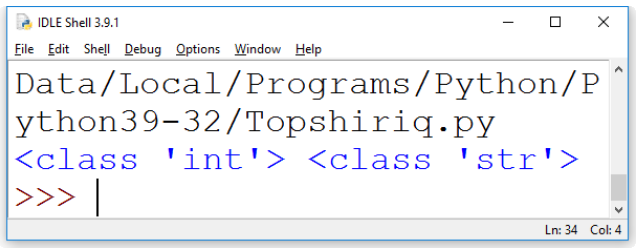


5-Topshiriq. Quyidagi o‘zgaruvchilar turini aniqlang. alfa=8764; beta=’Lola’

Dastur kodi:

```
alfa=8764
beta='Lola'
print(type(alfa),type(beta))
```

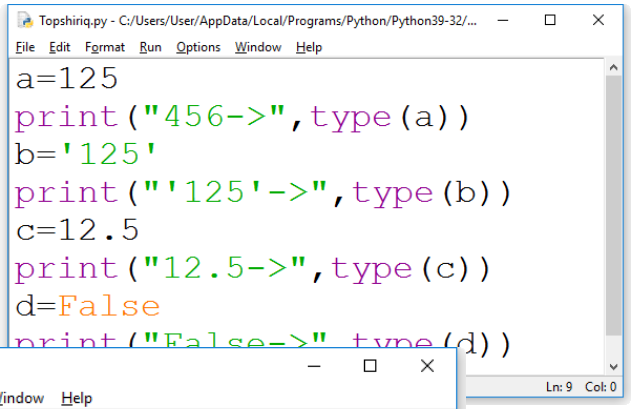


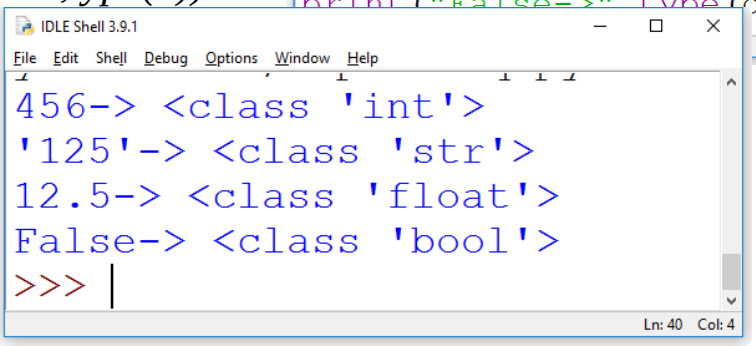


6-Topshiriq. O‘zgaruvchi uchun qiymat kiritilganda, uning turini aniqlovchi dastur tuzing.

Dastur kodi:

```
a=125
print("456->",type(a))
b=’125’
print("'125'->",type(b))
c=12.5
print("12.5->",type(c))
d=False
print("False->",type(d))
```



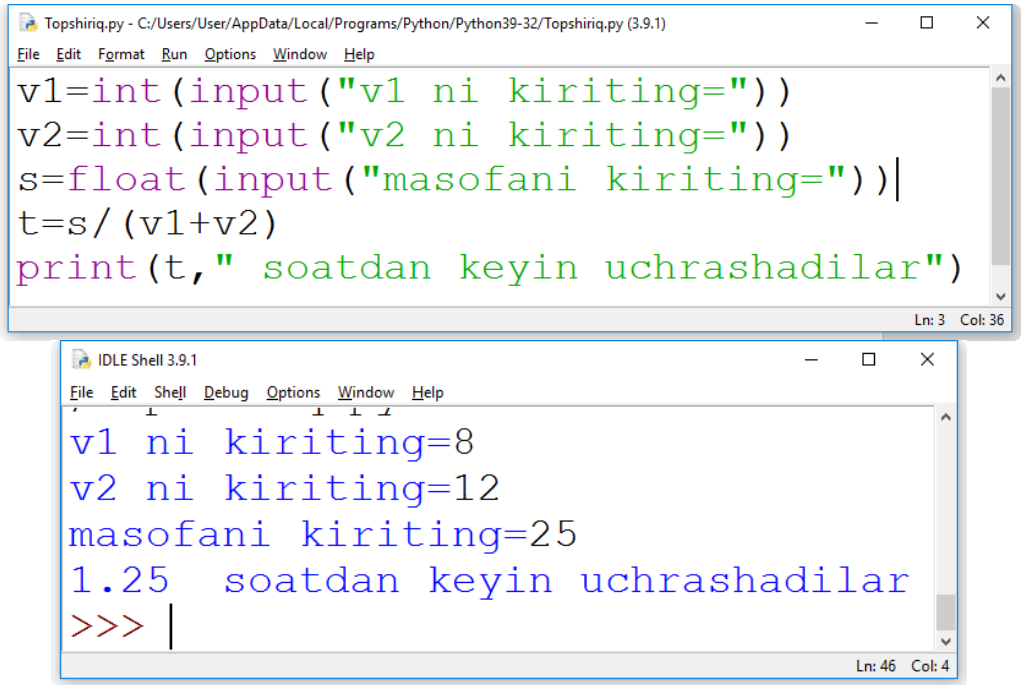


7-Topshiriq. Ikki qayiq turg’un suvda bir-biriga tomon a km/soat va b km/soat tezlik bilan suzmoqda. Ular orasidagi masofa S km bo‘lsa, ular qancha vaqtdan keyin uchrashadi?

Dastur kodi:

```
v1=int(input("v1 ni kiriting="))
v2=int(input("v2 ni kiriting="))
s=float(input("masofani kiriting="))
```

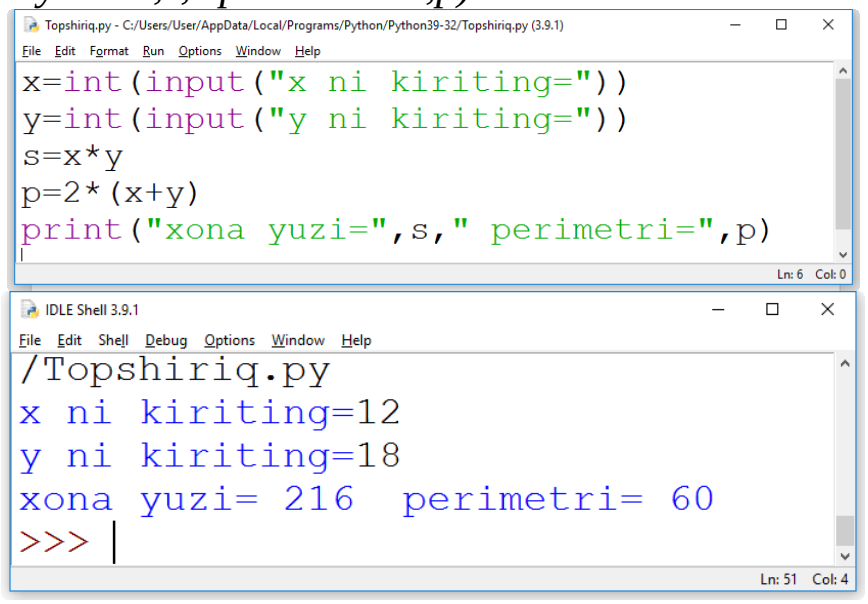
t=s/(v1+v2)
print(t," soatdan keyin uchrashadilar")



8-Topshiriq. Xonaning bo‘yi (x) va eni (y) metr foydalanuvchi tomonidan kiritiladi. Xonaning yuzasi (S) va perimetri (P)ni topish dasturini tuzing.

Dastur kodi:

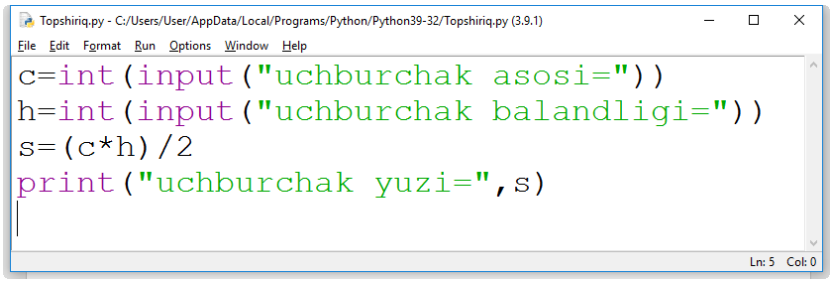
x=int(input("x ni kiriting="))
y=int(input("y ni kiriting="))
*s=x*y*
p=2(x+y)*
print("xona yuzi=",s," perimetri=",p)

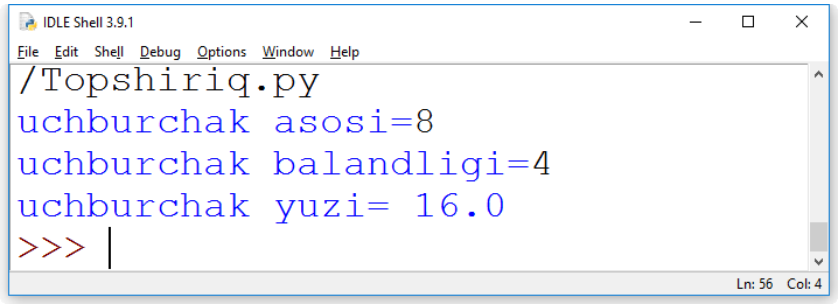


9-Topshiriq. Uchburchak asosining uzunligi (c) va balandligi (h) foydalanuvchi tomonidan kiritiladi. Uchburchak yuzasi (S) ni hisoblash dasturini tuzing.

Dastur kodi:

```
c=int(input("uchburchak asosi="))
h=int(input("uchburchak balandligi="))
s=(c*h)/2
print("uchburchak yuzi=",s)
```

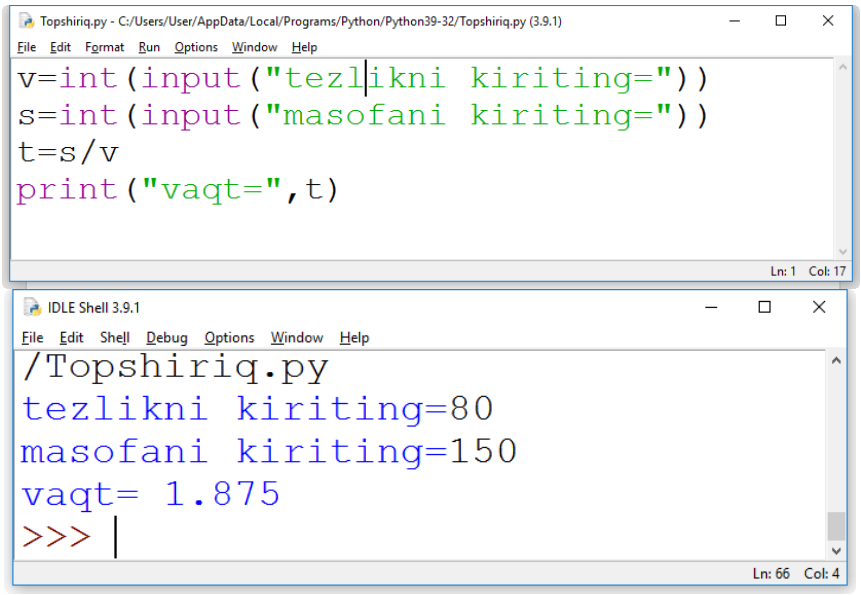




10-Topshiriq. Avtomobilning o‘rtacha tezligi v (km/soat) va bosib o‘tgan yo‘li s (km) foydalanuvchi tomonidan kiritiladi. Avtomobilning yurgan vaqti t (soat)ni hisoblash dasturini tuzing.

Dastur kodi:

```
v=int(input("tezlikni kiriting="))
s=int(input("masofani kiriting="))
t=s/v
print("vaqt=",t)
```



SAVOL VA TOPSHIRIQLAR

1. Ma’lumotlar turi nima?
2. Ma’lumotlarning qanday turlarini bilasiz?
3. Ma’lumotlar turini o‘zgartirish mumkinmi?
4. O‘zgaruvchi qiymati qaysi turdagi ma’lumotlarni qabul qiladi?
5. Identifikator nima?
6. O‘zgaruvchi va doimiyning bir-biridan farqli jihati nimada?
7. O‘zgaruvchilarni nomlashda qanday belgilardan foydalanish mumkin emas?

3.3. Python tilida satrlar

Python matn va uning qismlari bilan ishlash uchun eng qulay dastur hisoblanib, uning yordamida satrlarni bir-biriga bog‘lash yoki satr ichidan biror qismini qirqib olish mumkin.

Satrlar – qo‘shtirnoq ichiga olingan Unicode kodidagi belgilar ketma-ketligi.

Satr – harf, son va belgilar hamda probeldan tarkib topgan belgilar ketma-ketligi. Satrlarni o‘zgaruvchilar yordamida kiritish mumkin. Pythonda satrlar bittalik va ikkitalik qo‘shtirnoqlar orqali beriladi.

Satr uzunligini **len()** funksiyasi yordamida aniqlash mumkin. Pyhton barcha belgi va probellar sonini o‘zi hisoblab chiqaradi.

Satr – harf, son, va belgilar hamda probeldan tashkil topgan belgilar ketma-ketligidir.

a[index] – **a** satrdagi **indexda** turgan belgini qirqib oladi.

a[:end] – **a** satrdan **0** indeksdan boshlab **end** indeksgacha bo‘lgan belgilar ketma-ketligini qirqib oladi;

a[start:end] – **a** satrdan **start** indeksdan boshlab **end** indeksgacha bo‘lgan belgilar ketma-ketligini qirqib oladi;

a[start:] – **a** satrdan **start** indeksdan boshlab oxirigacha bo‘lgan belgilar ketma-ketligini qirqib oladi;

a[start:end:step] – **a** satrdan **step** qadam bilan **start** indeksdan boshlab **end** indeksgacha bo‘lgan belgilar ketma-ketligini qirqib oladi.

Pythonda satrdagi belgilarni raqamlash 0 dan boshlanadi va bu raqam indeks deb ataladi.

Quyidagi dastur natijalarini aniqlang.

1
```
satr = 'instagram'
print (satr[1:3])
```
2
```
x = 64/2**4*(7-5)%4//2
print(x)
```
3
```
print(type(6/3))
```
4
```
var = "Maktab "*2*4
print(var)
```
5
```
var= "Kutubxona"
print(var[3:2])
```
6
```
var= "Kutubxona"
print(var[-3:1])
```
7
```
var= "Kutubxona"
print(var[3:-2])
```
8
```
var= "Kutubxona"
print(var[-3:-1])
```
9
```
var= "odobli bola"
print(var[1::2])
```
10
```
a= "Informatika"
print (a[-4::2])
```
11
```
var= "O`zbekiston"
print(var[7::-3])
```
12
```
a='Qudratli'
print (a[-4::-2])
```
13
```
var= "Kutubxona"
print(var[5::])
```
14
```
var= "Axborot"
print(var[-3::])
```
15
```
a= "Matematika"
print (a[::-3])
```
16
```
a= "Matematika"
print (a[::3])
```
17
```
var= "Samsung"
print(var[6::-2])
```
18
```
var= "YouTube"
print(var[-4::-1])
```

split() usuli **input()** operatori orqali kiritilgan satrli ma’lumotlarni ajratuvchi belgi yordamida qismlarga ajratadi. Odatda, probel ajratuvchi belgi vazifasini bajaradi. Agar satr tarkibidagi qismlarni ajratish uchun boshqa belgi ishlatilgan bo‘lsa, u holda bu belgini split() qavslari ichida ko‘rsatish kerak bo‘ladi.

split — operator (yoki funksiya);

Ma’lumotlarni chiqarish usullari yordamida ma’lumotni ixtiyoriy ko‘rinishda aks ettirish mumkin. Buning uchun sep va end argumentidan foydalanamiz.

end va **sep print()** funksiyasining argumentlari bo‘lib, ma’lumotlarni chiqarish parametrlarini o‘zgartirish uchun qo‘llaniladi.

in amali satr ichida qism satr yoki belgi mavjudligini aniqlashda ishlatiladi.

```
s="matematika"
a=s.split("a")
print(a)
```

Natija: ['m', 'tem', 'tik', '']

```
s="Mustaqillik"
print(' '.join(s))
```

Natija: M u s t a q i l l i k

SAVOL VA TOPSHIRIQLAR

1. Dastur natijasini aniqlang.

```
x=45//4%(3+2)**4+25//6
print(x)
```

2. Dastur natijasini aniqlang.

```
x=25%12**2-(6+4)//2**2*len('test')
print(x)
```

3. Dastur natijasini aniqlang.

```
a='Informatika'
print(a[2:5])
```

4.

```
a='Informatika'
b='Matematika'
c='Haykaltarosh'
print(a[:5]+b[4:8]+c[3:8:4])
```

5. Dastur natijasini aniqlang.

```
x=15//8%(16-14)**5+17//7
print(x)
```

6. Dastur natijasini aniqlang.

```
x=34%len('dars')**2-(7*3)//2**2*7
print(x)
```

7. Dastur natijasini aniqlang.

```
a='O‘zbekiston'
print(a[3:6])
```

8.

```
a=‘Biologiya'
b=‘Kimyo'
c=‘Adabiyot'
print(a[:4]+b[1:3]+c[1:6:3])
```

9. Satrlarni birlashtirish qanday amalga oshiriladi?
10. Satr uzunligi qaysi funksiya yordamida aniqlanadi?
11. Satrdan qism satr qirqib olish qanday bajariladi?
12. “*” amali satrlar bilan ishlashda qanday vazifa bajaradi?

4. Pythonda ma’lumot to‘plamlari va turlari
4.1. List ro‘yxatlar va ular bilan ishlashda qo‘llaniladigan funksiyalar va metodlar

Massiv – bitta identifikator ostida bir xil obyektlar to‘plamini saqlovchi ma’lumotlar turi.

Ro‘yxat (list) – bitta identifikator ostida har xil obyektlar to‘plamini saqlovchi ma’lumotlar turi.

Indeks – elementga murojaat qiluvchi raqamlar ketma-ketligi

Ro‘yxatda bir vaqtda son, satr va boshqa turdagi ma’lumotlarni saqlash mumkin. Odatda, har bir o‘zgaruvchi biror nom bilan aniqlanib, tarkibida bitta ma’lumot (qiymat)ni saqlaydi. Ro‘yxatlar esa bir nechta ma’lumot (qiymat)larni o‘z
ichiga olishi mumkin. Masalan, 25 nafar o‘quvchi familiyasini saqlash uchun 25 ta o‘zgaruvchi yoki bitta ro‘yxatdan foydalanish mumkin. Ro‘yxat tartib bilan saqlangan elementlar tuzilmasi bo‘lib, har bir elementga bitta indeks to‘g‘ri keladi va u orqali elementga murojaat etish mumkin. Indeks raqamlari boshqa dasturlash tillaridagi kabi 0 dan boshlangan sonlar ketma-ketligidan iborat.

Ro‘yxat nomi	mevalar			
Ro‘yxat indeksi	0	1	2	3
	0	-3	-2	-1
Ro‘yxat elementi	olma	banan	shaftoli	nok

Ro‘yxatlarni e’lon qilish. Ro‘yxatlar ham o‘zgaruvchilar kabi e’lon qilinadi. Faqat ro‘yxatlarni e’lon qilishda, ular o‘lchami, ya’ni ro‘yxatda saqlanuvchi elementlar sonini bilish lozim. Ro‘yxatlar []qavs yoki list() konstruktori yordamida e’lon qilinadi. Ro‘yxatga elementlarni qo‘shish, ya’ni ro‘yxatlar hosil qilish uchun turli usullardan foydalanish mumkin.

List – tartiblangan va o‘zgaruvchan ro‘yxat.

List - Pythonda erkin turdagi obyektlarning o‘zgaruvchan qatorlashgan kolleksiyasi hisoblanadi (*massivga o‘xshash, lekin tiplar har xil bo‘lishi mumkin).* Ro‘yxatlardan foydalanish uchun ularni tuzish kerak. Ro‘yxatni kvadrat qavslar bilan yoki oddiy qavslar bilan hosil qilish mumkin:

Kvadrat qavslar bilan e’lon qilish:

```
a=["anor","olma","nok","gilos","shaftoli"]
print(a)
```

Oddiy qavslar bilan e’lon qilish:

```
a=list(("anor","olma","nok","gilos","uzum"))
print(a)
```

List elementlariga murojaat qilish uchun, murojaat qilinayotgan elementning indeksi ko‘rsatiladi. Sanoq har doimgidek 0 dan boshlanadi.

Manfiy indeks bu – sanoq oxiridan boshlanishini bildiradi. Masalan, -1 eng oxirgi, -2 oxiridan ikkinchi element.

Masalan:

```
fan = ['Fizika', 'Matematika','Kimyo','Adabiyot']
print(fan[-2])
```

Ro‘yxatning elementlaridan bir nechtasini tanlab olish uchun indekslar oralig’i kiritiladi. Bunda uning boshlanish va oxirgi nuqtalari kiritiladi. Element tanlashda oxirgi element indeksini inobatga olmaymiz. Ya’ni boshlang’ich nuqtadan boshlanib oxirgi nuqtadan bitta oldingi elementgacha olinadi.

```
fan = ['Fizika', 'Matematika','Kimyo','Adabiyot']
print(fan[1:3])
fan = ['Fizika', 'Matematika','Kimyo','Adabiyot']
print(fan[:3])
fan = ['Fizika', 'Matematika','Kimyo','Adabiyot']
print(fan[1:])
```

List ro‘yxatidagi istalgan element qiymatini o‘zgartirish mumkin. Bunda ro‘yxat indeksiga murojaat qilinadi va yangi qiymat beriladi.

```
 fan = ['Fizika', 'Matematika','Kimyo','Adabiyot']
fan[0]='Informatika'
print(fan)
fan = ['Fizika', 'Matematika','Kimyo','Adabiyot']
fan[0]='Informatika'; fan[2]= 'Biologiya'
print(fan)
```

Ro‘yxat elementlarini tanlab olishda **for** operatoridan foydalanish mumkin.

```
 meva=['olma','anor','nok','gilos']
for i in meva:
   print(i)
```

```
son=list((5,2,6,3,4,8))
for i in son:
    print(i)
```

1. Ro‘yxat elementlari yig’indisini aniqlovchi dastur tuzing.

```
son=[5,2,6,3,4,8]
s=0
for i in son:
    s+=i
print(s)
```

2. Berilgan ro‘yxatdagi eng katta sonni aniqlovchi dastur tuzing.

```
son=[445,600,9000,3,964,800]
a=son[0]
for i in son:
    if i>a:
        a=i
print(a)
```

3. Ro‘yxatning juft indeksidagi sonlarni chiqaruvchi dastur tuzing.

```
son=[44,7,19,3,9,80,10]
for i in range(0,7):
    if i%2==0:
        print(son[i])
```

4. Elementlari 0 dan 5 gacha bo‘lgan sonlar kvadratlariga teng ro‘yxat hosil bo‘ladi.

```
n=5
a=[i**2 for i in range(n)]
print(a)
```

map() funksiyasi

map() funksiyasi – takrorlanish operatoridan foydalanmasdan funksiya orqali natijalarni ro‘yxatga yozish, qayta ishlash va o'zgartirish imkonini beradi.

map(funksiya nomi, ro‘yxat nomi)

map() ning natijasi “takrorlanuvchi” xarita obyekti bo‘lib, u asosan ro‘yxat kabi ishlaydi, **lekin chop etilmaydi**.

```
b = [1, 2, 3, 4, 5]
a=[]
for i in b:
    a.append(i**2)
```

```
b = [1, 2, 3, 4, 5]
print(list(map(lambda n: n*2, b)))
```

```
print(a)

b = [1, 2, 3, 4, 5]
def daraja(a):
    return a**2
print(map(daraja,b))
```

```
b = [1, 2, 3, 4, 5]
def daraja(a):
    return a**2
print(list(map(daraja,b)))

a = ['Summer', 'is', 'coming']
print(list(map(lambda b: b.upper()+'!', a)))
```

Ro‘yxatlarni o‘zgartirish. Xohlagan vaqtda ro‘yxat elementini almashtirish, o‘chirish yoki ro‘yxatga yangi element qo‘shish mumkin. Buning uchun metodlardan foydalaniladi.

Metodlar	Tavsifi
list.insert(index, item)	Ro‘yxatga index indeksi bo‘yicha item elementini qo‘shish.
list.append(item)	Ro‘yxat oxiriga item elementini qo‘shish.
list.remove(item)	Ro‘yxatdan item elementini o‘chirish. Ushbu metod ro‘yxatdan birinchi uchragan item elementini o‘chiradi, agar bu element ro‘yxatda mavjud bo‘lmasa, ValueError holati yuzaga keladi.
list.pop([index])	Ro‘yxatdan index indeksi bo‘yicha elementni o‘chiradi. Indeks ko‘rsatilmasa, oxirgi elementni o‘chiradi.
list.clear()	Ro‘yxatdagi barcha elementlarni o‘chiradi.
list.index(item,[start [, end]])	Ro‘yxatdagi item elementining indeksini qaytaradi (bu metodda start va end dan ham foydalanish mumkin).
list.count(item)	Ro‘yxatdagi item elementlar sonini hisoblaydi.
list.sort()	Ro‘yxat elementlarini o‘sish tartibida tartiblaydi.
list.sort(reverse=True)	Ro‘yxat elementlarini kamayish tartibida tartiblaydi.
list.copy()	Ro‘yxatdan nusxa oladi.

Ma’lum bir elementning ro‘yxatda mavjudligini tekshirish uchun **in** operatori ishlatiladi.

```
ism=['Ali','Vali','Soli','Xoli']
if 'Vali' in ism:
    print('bu ism mavjud')
else:
    print('bu ism mavjud emas')
```

Ro‘yxatda elementlari sonini aniqlash uchun **len()** funksiyasi ishlatiladi.

```
ism=['Ali','Vali','Soli','Xoli']
```

```
print(len(ism))
```

Ro‘yxat elementlari soni bo‘luvchilarini chiqaruvchi dastur tuzing.

```
ism=['Ali','Vali','Soli','Xoli','Toir','Zoir']
for i in range(1,len(ism)+1):
    if len(ism)%i==0:
        print(i)
```

Ro‘yxatning oxiriga yangi element qo‘shish uchun **append()** funksiyasi ishlatiladi.

```
ism=['ali','vali','soli','xoli']
ism.append('toir')
print(ism)
```

a = [1, 2, 3, 4, 5, 6, 7] ro‘yxat berilgan. Quyidagi ko‘rinishda hosil bo‘luvchi dastur tuzing. b = [1, 4, 9, 16, 25, 36, 49]

```
a=[1,2,3,4,5,6,7]
b=[]
for i in a:
    b.append(i**2)
print('a={}\nb={}'.format(a,b))
```

yoki print(f'a={a}\nb={b}')

Ro‘yxat elementini istalgan o‘rniga element qo‘shishda **insert()** funksiyasidan foydalanamiz.

```
ism=['ali','vali','soli','xoli']
ism.insert(0,'toir')
print(ism)
```

meva=['olma','anor','uzum','gilos'] ro‘yxat elementlari berilgan. Ushbu ro‘yxatni quyidagi ko‘rinishda hosil qiluvchi dastur tuzing.

meva=['nok', 'olma', 'nok', 'anor', 'nok', 'uzum', 'nok', 'gilos']

```
meva=['olma','anor','uzum','gilos']
for i in range(len(meva)):
    meva.insert(2*i,'nok')
print(meva)
```

Ro‘yxatdan elementni o‘chirishda **remove(), pop(), clear()** funksiyalaridan hamda **del** xizmatchi so‘zidan foydalanamiz.

1) **remove()** funksiyasi orqali ro‘yxatdan elementni o‘chirishda uning indeksiga emas balki element nomiga murojaat qilinadi.

2) **pop()** funksiyasi orqali ro‘yxatdan elementni o‘chirishda uning indeksi bo‘yicha elementni ro‘yxatdan o‘chiradi. Agar ro‘yxat indeksi ko‘rsatilmasa avtomatik ravishda ro‘yxatning oxirgi elementini o‘chiradi.

3) **clear()** funksiyasi ro‘yxatdagi barcha elementlarni o‘chiradi.

4) **del** xizmatchi so‘ziga element indeksi ko‘rsatilgan holda ro‘yxat elementi o‘chiriladi.

```
rang=['qizil','sariq','yashil','qora']
rang.remove('sariq')
print(rang)
```
Javob: ['qizil', 'yashil', 'qora']

```
rang=['qizil','sariq','yashil','qora']
rang.pop(2)
print(rang)
```
Javob: ['qizil', 'sariq', 'qora']

```
rang=['qizil','sariq','yashil','qora']
rang.pop()
print(rang)
```
Javob: ['qizil', 'sariq', 'yashil']

```
rang=['qizil','sariq','yashil','qora']
del rang[0]
print(rang)
```
Javob: ['sariq', 'yashil', 'qora']

```
rang=['qizil','sariq','yashil','qora']
rang.clear()
print(rang)
```
Javob: [ ]

rang=['qizil','sariq','yashil','qora'] ro‘yxat elementlari berilgan. Ushbu ro‘yxatni quyidagi ko‘rinishda hosil qiluvchi dastur tuzing.
rang=['qizil', 'sariq', 'ko‘k', 'qora']

```
rang=['qizil','sariq','yashil','qora']
rang.pop(2)
rang.insert(2,'ko`k')
print(rang)
```

Ro‘yxatdan nusxa olishda **copy()** va **list()** funksiyalaridan foydalanish mumkin.

```
rang=['qizil','sariq','yashil','qora']
rang_2=rang.copy()
print(rang_2)

rang=['qizil','sariq','yashil','qora']
rang_2=list(rang)
print(rang_2)
```

a=[12,52,63,42,85,96] ro‘yxat elementlari berilgan. Ushbu ro‘yxatni b nomli ro‘yxatga nusxalab, quyidagi ko‘rinishda hosil qiluvchi dastur tuzing.

b=[12, 52, 63, 42, 85, 96, 100]

```
a=[12,52,63,42,85,96]
b=a.copy()
b.append(100)
print('b=',b)
```

Ro‘yxatlarni birlashtirish uchun **extend()** funksiyasi yoki “+” belgisidan foydalanish mumkin.

```
a=[1,2,3,4,5,6]
b=[7,8,9,10,11]
a.extend(b)
print(a)

a=[1,2,3,4,5,6]
b=[7,8,9,10,11]
print(a+b)
```

a=[1,2,3,4,5,6] va b=[7,8,9,10,11] ro‘yxatlarni birlashtirib, hosil bo‘lgan ro‘yxatning toq indeksli elementlarini c=[2, 4, 6, 8, 10] ro‘yxatda hosil qiluvchi dastur tuzing.

```
a=[1,2,3,4,5,6]
b=[7,8,9,10,11]
c=[]
a.extend(b)
for i in range(len(a)):
    if i%2!=0:
        c.append(a[i])
print(c)
```

Ro‘yxatni biror songa ko‘paytirish mumkin, natijada takrorlangan ro‘yxat hosil bo‘ladi.

```
b=[4,5,6]
c=b*2
print(c)
```

Natija: [4, 5, 6, 4, 5, 6]

```
a=[1,2,3,4,5]
c=[]
for i in a:
    c.append(i*2)
print(c)
```

Natija:[2, 4, 6, 8, 10]

Ro‘yxatdagi bir xil elementlar sonini aniqlashda **count()** funksiyasidan foydalanamiz. ishlatiladi. **index()** funksiyasi berilgan elementni indeksini aniqlaydi, agarda bunday elementlar bir nechta bo‘lsa, faqat birinchi uchragan element indeksini aniqlaydi.

```
a=[12,52,63,42,85,12]
b=a.count(12)
print(b)

a=[12,52,63,42,85,12]
b=a.index(12)
print(b)
```

a=['b','c','d','b','c','a','a'] ro‘yxat berilgan, ushbu ro‘yxatdagi o‘xshash elementlardan faqatgina bittasini olib quyidagi ro‘yxat ko‘rinishida hosil bo‘luvchi dastur tuzing. b=['b', 'c', 'd', 'a']

```
a=['b','c','d','b','c','a','a']
b=[]
for i in a:
    if not i in b:
        b.append(i)
print('b=',b)
```

sort() funksiyasi ro‘yxatni tartiblaydi. Berilgan ro‘yxat elementlari sonlardan tashkil topgan bo‘lsa, o‘sish tartibida, satr yoki harflardan iborat bo‘lsa, alifbo bo‘yicha tartiblaydi.
reverse() funksiyasi ro‘yxat elementlarining joriy holatini teskari holatga o‘zgartiradi.

```
a=['xoli','toir','vali','soli','ali']
a.sort()
print(a)

a=['xoli','toir','vali','soli','ali']
a.reverse()
print(a)
```

a=[5,4,3,7,1,2,8,9] ro‘yxatni tartiblang hamda 2 raqami indeksini aniqlovchi dastur tuzing.

```
a=[5,4,3,7,1,2,8,9]
a.sort()
b=a.index(2)
print(a)
print(b)
```

max() va min() funksiyalari

```
n=int(input('o‘quvchilar soni='))
x=[0]*n
s=0
for i in range(n):
    x[i]=int(input())
    s+=x[i]
a=min(x)
b=max(x)
c=s/n
print('eng past ball:',a)
print('eng yuqori ball:',b)
```

```
print('o‘rtacha ball:',c)
```

SAVOL VA TOPSHIRIQLAR

1. Ro‘yxat nima va u qanday e’lon qilinadi?
2. Indekslar manfiy son bo‘lishi mumkinmi?
3. range() funksiyasi yordamida qanday massivlar hosil qilinadi?
4. randint() funksiyasi yordamida qanday massivlar hosil qilinadi?
5. Ichma-ich joylashgan ro‘yxatlar qanday hosil qilinadi?
6. Ro‘yxatdagi elementlar o‘rnini qanday almashtirish mumkin?

4.2. Tuple kortejlar bilan ishlash

Ma’lumotlar tizimida shunday ma’lumotlar ham mavjudki, ularni har xil rejali va tasodifiy o‘zgarishlardan himoyalash zarur bo‘ladi. Avvalgi mavzuda o‘tilgan ro‘yxatlarni dasturning ixtiyoriy qismida o‘zgartirish, xususan, ularga yangi element qo‘shish, almashtirish yoki o‘chirish mumkin. Shu sababli, bunday holatlarda Python dasturlash tilida ro‘yxat vazifasini bajara oladigan, ammo tarkibini o‘zgartirib bo‘lmaydigan kortejlardan foydalaniladi.

Tuple – tartiblangan va o‘zgarmas ro‘yxat. Elementlarini dublikatlash mumkin.

Kortej - elementlar orasini vergul bilan ajratish orqali hosil qilinadi.

Tuple ro‘yxati tartiblangan, o‘zgarmas ro‘yxat. Uning elementlarini o‘zgartirib bo‘lmaydi. Bu ro‘yxatni oddiy qavslar bilan yoki tuple() konstruktori bilan hosil qilinadi:

Masalan:

```
a = ("olma", "anor", "shaftoli")
b = tuple(("gilos", "nok", "xurmo"))
print(a)
print(b)
```

Tuple elementiga murojaat qilishda uning indeksiga ko‘rsatiladi.

```
a = ("olma", "anor", "shaftoli")
print(a[0])
```

Natija: olma

```
a = ("olma", "anor", "shaftoli")
```

```
print(a[-1])
```
Natija: shaftoli
```
a = ("olma", "anor", "shaftoli","nok")
print(a[1:3])
```
Natija: ('anor', 'shaftoli')

Tuple ro‘yxati tartiblangan, o‘zgarmas ro‘yxat deb yuqorida ta’kidladik, lekin Tuple ro‘yxatini avval list ro‘yxatiga aylantirib, so‘ngra uning istalgan elementini almashtirish mumkin.
```
a = ("olma", "anor", "shaftoli","nok")
b=list(a)
b[2]="gilos"
a=tuple(b)
print(a)
```
Natija: ('olma', 'anor', 'gilos', 'nok')

Tuple to‘plamida ham for takrorlanishdan foydalanib elementlarni tanlab olish mumkin.
```
a = ("olma", "anor", "shaftoli","nok")
for i in a:
    print(i)
```
Natija:
```
olma
anor
shaftoli
nok
```
Ro‘yxat elementlarini yig’indisini hisoblash
```
a = (1,2,3,4,5,6,7)
s=0
for i in a:
    s+=i
print(s)
```
Natija: 28

Biror elementning to‘plamda bor yoki yo‘q ekanligini **in** kalit so‘zi orqali tekshiramiz.
```
a = ("olma", "anor", "shaftoli","nok")
if 'anor' in a:
    print('anor bor')
else:
    print('anor yo‘q')
```

Natija: anor bor

any() - agar kortej elementi mavjud bo‘lsa **True** qiymat qaytaradi, aks holda (kortej bo‘sh bo‘lsa) **False** qiymat

max() - kortejning maksimal elementini qaytaradi.

min() - kortejning minimal elementini qaytaradi.

sorted() - kortej elementlaridan iborat yangi tartiblangan ro‘yxatni qaytaradi.

sum() - kortej elementlari yig`indisini qaytaradi. qaytaradi.

all() - agarda hamma elementlar haqiqiy bo‘lsa (yoki ketma-ketlik bo‘sh bo‘lsa) True ni qaytaradi. ***Eslatma****: agar ro‘yxatdagi biror element yolg’on bo‘lsa, False natija qaytaradi.*

```
a = (1,2,3,4,5,6,7)
print(any(a))
```
Natija: True
```
a = ()
print(any(a))
```
Natija: False
```
a = (",0,False)
print(any(a))
```
Natija: False
```
a = (",0,True)
print(any(a))
```
Natija: True
```
a = (",0,1,2,3)
print(any(a))
```
Natija: True
```
a = ('men','sen','ular')
print(any(a))
```
Natija: True

max(), min(), sorted(), sum() funksiyasi
```
a = (15,26,12,-9,15)
print(max(a))
```
Natija: 26
```
a = (15,26,12,-9,15)
print(min(a))
```
Natija: -9
```
a = (15,26,12,-9,15)
print(sorted(a))
```

Natija: [-9, 12, 15, 15, 26]

```
a = (15,26,12,-9,15)
print(sum(a))
```

Natija: 59

all() funksiyasi

```
a = (15,26,12,-9,15)
print(all(a))
```

Natija: True

```
a = ()
print(min(a))
```

Natija: True

```
a=(1,2,3,4,5,None)
print(all(a))
```

Natija: False

```
a = ('olma','anor',0,5)
print(all(a))
```

Natija: False

```
a = ('olma','anor',False,5)
print(all(a))
```

Natija: False

```
a = ('olma','anor',[],5)
print(all(a))
```

Natija: False

Eslatma: ro‘yxat tarkibida **{}, [], (), 0, False, None** kelsa, all funksiyasi False natija qaytaradi.

```
print(all([]))
```

Natija: True

```
print(all([[]]))
```

Natija: False

```
print(all([[[]]]))
```

Natija: True

```
print(all([[[[]]]]))
```

Natija: True

……

```
print(all([[[[[[[[]]]]]]]]))
```

Natija: True

SAVOL VA TOPSHIRIQLAR

1. Kortej va ro‘yxatning bir-biridan farqi nimada?
2. Kortejga element qo‘shish mumkinmi?
3. Kortejdagi elementni o‘zgartirish mumkinmi?
4. Kortejni saralash nima?
5. Kortejni saralashning asosiy usullari qaysilar?
6. Kortejning eng kichik elementi qanday aniqlanadi?

4.3. Set to‘plamlar bilan ishlash

To‘plam matematikadagi to‘plam tushunchasiga ekvivalent bo‘lgan tushuncha bo‘lib, ma’lumotlar strukturasini anglatadi. U tartibi aniqlanmagan turli elementlardan tashkil topgan. To‘plamlarda element qo‘shish, o‘rin almashtirish va o‘chirish hamda elementlarni birlashtirish, o‘zaro almashtirish kabi amallarni bajarish mumkin. Shuningdek, elementning to‘plamga tegishli yoki tegishli emasligini aniqlash ham mumkin.

Set – tartiblanmagan va indekslanmagan to‘plam. Elementlari dublikatlanmaydi.

To‘plam – takrorlanmaydigan elementlardan ixtiyoriy tartibda tashkil topgan ma’lumotlar turi.

To‘plam elementi ixtiyoriy o‘zgarmas ma’lumotlar turi bo‘lishi mumkin: son, satr, kortej va h. k. O‘zgaruvchan turdagi ma’lumotlar to‘plam elementi bo‘la olmaydi. Masalan, ro‘yxat to‘plam elementi bo‘la olmaydi, lekin kortej to‘plam elementi bo‘la oladi. To‘plamlar, asosan, takrorlanuvchi elementlarni o‘chirish uchun qulay hisoblanadi.

To‘plamlarni e’lon qilish. To‘plamlar {}qavs yoki set() konstruktori yordamida e’lon qilinadi. To‘plam elementlari qiymatlari unikal bo‘lishi zarur. Agar bir xil qiymatlar takrorlansa, u holda takrorlanuvchi qiymatlar bitta qiymat deb hisoblanadi. To‘plamlarga elementlar qo‘shish, ya’ni to‘plamlar hosil qilish uchun ro‘yxatlar kabi turli usullardan foydalanish mumkin.

To‘plamlar (set) – bitta identifikator ostida har xil takrorlanmaydigan elementlar to‘plamini saqlovchi ma’lumotlar turi. To‘plamni hosil qilish uchun maxsus qavslardan yoki **set**() konstruktori ishlatiladi:

```
a = {1,2,3,5,6,9}
print(a)
```

Natija: {1, 2, 3, 5, 6, 9}

```
a = set((1,2,3,5,6,9))
print(a)
```

Natija:{1, 2, 3, 5, 6, 9}

add() fuksiyasi –to‘plamga **bitta element** qo‘shadi.

update() funksiyasi – B to‘plamdagi elementlarni A to‘plamga qo‘shadi. Masalan: A |= B A.update(B)

```
a = {1,2,3,5,6,9}
a.add(10)
print(a)
```

Natija: {1, 2, 3, 5, 6, 9, 10}

```
a = {1,2,3,5,6,9}
a.update([10,11,12])
print(a)
```

Natija: {1, 2, 3, 5, 6, 9, 10, 11, 12}

Elementni to‘plamdan **pop()** funksiyasi bilan ham o‘chirish mumkin. Ammo **pop()** funksiyasi xususiyatiga ko‘ra ro‘yxatning birinchi elementni o‘chiradi.

clear() – funksiyasi to‘plamni tozalaydi, ya’ni barcha elementlarini o‘chiradi.

```
a = {1,2,3,5,6,9}
a.pop()
print(a)
```

Natija: {2, 3, 5, 6, 9}

```
a = {1,2,3,5,6,9}
a.clear()
print(a)
```

Natija: set()

remove() – to‘plamdan elementni o‘chiradi.

discard() – to‘plamdan elementni o‘chiradi.

```
a = {1,2,3,5,6,9}
a.remove(9)
print(a)
```

Natija: {1, 2, 3, 5, 6}

```
a = {1,2,3,5,6,9}
a.discard(9)
print(a)
```

Natija: {1, 2, 3, 5, 6}

union() - A va B to‘plamini birlashtirib, yangi to‘plam hosil qiladi.
Masalan: A | B A.union(B)

```
a = {1,2,3,5,6,9}
b = {10,11,12,13,14}
c=a.union(b)
print(c)
```

Natija: {1, 2, 3, 5, 6, 9, 10, 11, 12, 13, 14}
copy() funksiyasi to‘plamdan nusxa oladi.

```
a = {1,2,3,5,6,9}
b=a.copy()
print(b)
```

Natija: {1, 2, 3, 5, 6, 9}

- **A.difference(B)** A va B to‘plamlarning ayirmasi, ya’ni A to‘plamda mavjud va B to‘plamida mavjud bo‘lmagan elementlarini qaytaradi. Masalan: A - B A.difference(B)
- **A.difference_update(B)** A to‘plamdan B to‘plamda mavjud elementlarni o‘chiradi. Masalan: A - B A.difference_update(B)

```
a = {1,2,3,5,6,9,12,16}
b = {5,2,6,3,8,12,11,14}
c=a.difference(b)
print(c)
```

Natija: {16, 1, 9}

```
a = {1,2,3,5,6,9,12,16}
b = {5,2,6,3,8,12,11,14}
a.difference_update(b)
print(a)
```

Natija: {1, 9, 16}

- **A.intersection(B) -** A va B to‘plamlari kesishmasi, ya’ni ikkala to‘plam uchun umumiy bo‘lgan elementlarni oladi.
  Masalan: A & B A.intersection(B)
- **A.intersection_update(B)** A to‘plamda B to‘plamda mavjud elementlarni qoldiradi. Masalan: A&= B A.intersection_update(B)

```
a = {1,2,3,5,6,9,12,16}
b= {5,2,6,3,8,12,11,14}
c=a.intersection(b)
print(c)
```

Natija: {2, 3, 5, 6, 12}

```
a = {1,2,3,5,6,9,12,16}
b= {5,2,6,3,8,12,11,14}
a.intersection_update(b)
print(a)
```
Natija: {2, 3, 5, 6, 12}

isdisjoint() funksiyasi agar a to‘plamdagi birorta ham element b to‘plamda mavjud bo‘lmasa, rost qiymat qaytaradi.

```
a = {9,12,16}
b = {5,2,6,3}
c=a.isdisjoint(b)
print(c)
```
Natija: True

```
a = {9,12,16,5}
b = {5,2,6,3}
c=a.isdisjoint(b)
print(c)
```
Natija: False

- **A.issubset(B)** A to‘plami B to‘plamining qism to‘plami ekanligini tekshiradi. Masalan: A<=B A.issubset(B)
- **A.issuperset(B)** B to‘plami A to‘plamining qism to‘plami ekanligini tekshiradi. Masalan: A>=B A.issuperset(B)

```
a = {2,6,3,5}
b = {5,2,6,3}
c=a.issubset(b)
print(c)
```
Natija: True

```
a = {2,6,3,5}
b = {5,2,6,3,12}
c=a.issuperset(b)
print(c)
```
Natija: False

symmetric_difference() funksiyasi ikkala to‘plamda ham mavjud bo‘lgan bir xil elementlardan tashqari barcha elementlarni olib yangi to‘plam hosil qiladi.

```
a = {2,6,3,5}
b = {5,2,6,3,12}
c=a.symmetric_difference(b)
```

```
print(c)
```
Natija: {12}

SAVOL VA TOPSHIRIQLAR

1. To‘plam nima maqsadda ishlatiladi?
2. To‘plamning ro‘yxat va kortejdan farqli jihati nimada?
3. To‘plam tarkibini o‘zgartirish mumkinmi?
4. To‘plamlar ustida qanday amallar bajariladi?
5. To‘plam va ro‘yxatning bir-biridan farqi nimada?
6. Python. Dastur natijasini toping.

```
x,y={0},{1}
print(min(x,y)==min(y,x))
print(min(x|y)==min(y|x))
print(min(x or y)==min(y or x))
print(min(x and y)==min(y and x))
```

4.4. Dictionary va ular bilan ishlash

Ro‘yxat yoki kortejlar raqamlangan elementlar to‘plami hisoblanadi. Biror elementga murojaat etish uchun shu element raqami (indeksi)ni ko‘rsatish lozim. Ammo barcha axborotlarni har doim ham raqamlar yordamida aniqlab bo‘lmaydi. Masalan, poyezd yoki samolyot reyslari to‘g‘risidagi ma’lumotni saqlash uchun raqamlar emas, identifikator sifatida raqam va harflardan iborat belgilar ishlatiladi. Bunday holatlarda Python dasturlash tilida lug‘atlar (dictionary) deb nomlanuvchi elementlar to‘plamidan foydalaniladi.

Lug‘atlar ro‘yxatlarga o‘xshash bo‘lib, farqli jihati shundaki, ularda indeks o‘rnida maxsus kalitlar ishlatiladi. Lug‘atlarda elementning tartibi muhim rol o‘ynamaydi. Lug‘atning har bir elementi kalit va qiymatga ega bo‘lib, uning unikal kaliti orqali qiymatga murojaat etiladi. Shu sababli lug‘atlar assotsiativ massivlar deb ham yuritiladi.

Lug‘atlarni e’lon qilish. Lug‘atlar {}qavs yoki dict() konstruktori yordamida e’lon qilinadi. Lug‘atga elementlarni qo‘shish, ya’ni lug‘atlar hosil qilish uchun ro‘yxatlar kabi turli usullardan foydalanish mumkin.

Dictionary – tartiblanmagan, o‘zgaruvchan va indeksiz to‘plam. Bu to‘plamda kalit-qiymat (key-value) tushunchasi mavjud, ya’ni

maxsus kalit va ularga mos keluvchi qiymatlar juftligidan tashkil topgan. Chap tarafda kalitlar, o‘ng tomonda esa ularga mos keluvchi qiymatlar joylashgan bo‘ladi.

```
talaba={"ism": "Vali",
        "familiya": "Aliyev",
        "yoshi": 25}
print(talaba)
```

Natija: {'ism': 'Vali', 'familiya': 'Aliyev', 'yoshi': 25}

dict() konstruktori bilan ham yangi to‘plam hosil qilish mumkin.

```
talaba = dict(ism="Vali", familiya="Aliyev", yoshi=25)
print(talaba)
```

Natija: {'ism': 'Vali', 'familiya': 'Aliyev', 'yoshi': 25}

Dictionary elementlariga murojaat qilish uchun ularning kalitlarini **kvadrat qavs** ichida ko‘rsatish yoki **get()** funksiyasidan foydalanish mumkin.

```
talaba={"ism": "Vali",
        "familiya": "Aliyev",
        "yoshi": 25}
a=talaba["yoshi"]
b=talaba.get("ism")
print(a,b)
```

Natija: 25 Vali

Qiymatni o‘zgartirish uchun unga kalit orqali murojaat qilamiz, so‘ngra qiymatini o‘zgartiramiz.

```
talaba={"ism": "Vali",
        "familiya": "Aliyev",
        "yoshi": 25}
talaba["yoshi"]=20
print(talaba)
```

Natija: {'ism': 'Vali', 'familiya': 'Aliyev', 'yoshi': 20}

Yangi elementni, ya’ni kalit va qiymatni qo‘shish.

```
talaba={"ism": "Vali",
        "familiya": "Aliyev",
        "yoshi": 25}
talaba["manzili"]="Buxoro"
print(talaba)
```

Natija: {'ism': 'Vali', 'familiya': 'Aliyev', 'yoshi': 25, 'manzili': 'Buxoro'}

copy() yoki **dict()** funksiyalari orqali to‘plam elementlarini nusxa olish mumkin.

```
talaba={"ism": "Vali",
        "familiya": "Aliyev",
        "yoshi": 25}
a=talaba.copy()
print(a)
```

Natija: {'ism': 'Vali', 'familiya': 'Aliyev', 'yoshi': 25}

```
talaba={"ism": "Vali",
        "familiya": "Aliyev",
        "yoshi": 25}
a=dict(talaba)
print(a)
```

Natija: {'ism': 'Vali', 'familiya': 'Aliyev', 'yoshi': 25}

pop() funksiyasi yoki **del** kalit so‘zi orqali to‘plam elementlarini o‘chirish mumkin.

```
talaba={"ism": "Vali",
        "familiya": "Aliyev",
        "yoshi": 25}
talaba.pop("familiya")
print(talaba)
```

Natija: {'ism': 'Vali', 'yoshi': 25}

```
talaba={"ism": "Vali",
        "familiya": "Aliyev",
        "yoshi": 25}
del talaba["yoshi"]
print(talaba)
```

Natija: {'ism': 'Vali', 'familiya': 'Aliyev'}

popitem() funksiyasi to‘plamga kiritilgan oxirgi elementni o‘chiradi.

```
talaba={"ism": "Vali","familiya": "Aliyev","yoshi": '25'}
talaba.popitem()
print(talaba)
```

Natija:

{'ism': 'Vali', 'familiya': 'Aliyev'}

clear() funksiyasi to‘plamdagi barcha elementlarni o‘chiradi.

```
talaba={"ism": "Vali","familiya": "Aliyev","yoshi": '25'}
talaba.clear()
print(talaba)
```

Natija:
{}
To‘plamdagi **kalitlarga** murojaat qilish uchun **for** operatoridan foydalanamiz.

```
talaba={"ism": "Vali","familiya": "Aliyev","yoshi": 25}
for i in talaba:
    print(i)
```

Natija:
ism
familiya
yoshi

```
a=dict(ism='Vali',familiya='Aliyev',yoshi=25)
print(a.keys())
```

Natija:
dict_keys(['ism', 'familiya', 'yoshi'])
To‘plam **qiymatlariga** murojaat qilish uchun **values** operatoridan foydalanamiz.

```
talaba={"ism": "Vali","familiya": "Aliyev","yoshi": 25}
for i in talaba.values():
    print(i)
```

Natija:
Vali
Aliyev
25

```
talaba={"ism": "Vali","familiya": "Aliyev","yoshi": 25}
for i in talaba:
    print(talaba[i])
```

Natija:
Vali
Aliyev
25
setdefault() fuksiyasi ko‘rsatilgan kalit bo‘yicha element qiymatini qaytaradi. Agar bunday kalit to‘plamda mavjud bo‘lmasa, shu kalit ko‘rsatgan qiymatni yangi element sifatida to‘plamga qo‘shadi.

```
talaba={"ism": "Vali","familiya": "Aliyev","yoshi": '25'}
print(talaba.setdefault('ism'))
```

Natija: Vali

```
talaba={"ism": "Vali","familiya": "Aliyev","yoshi": '25'}
```

```
talaba.setdefault('manzil','Buxoro')
print(talaba)
```
Natija:
```
{'ism': 'Vali', 'familiya': 'Aliyev', 'yoshi': '25', 'manzil': 'Buxoro'}
```
update() funksiyasi to‘plamga yangi element qo‘shadi.
```
talaba={"ism": "Vali","familiya": "Aliyev","yoshi": '25'}
talaba.update({'manzil': 'Buxoro'})
print(talaba)
```
Natija:
```
{'ism': 'Vali', 'familiya': 'Aliyev', 'yoshi': '25', 'manzil': 'Buxoro'}
```

SAVOL VA TOPSHIRIQLAR

1. Lug‘at nima maqsadda ishlatiladi?
2. Lug‘atning ro‘yxat va kortejdan farqli jihati nimada?
3. Lug‘at tarkibini o‘zgartirish mumkinmi?
4. Lug‘atlar qanday usullar bilan hosil qilinadi?
5. Lug‘at va ro‘yxatning bir-biridan farqi nimada?

5. Pythonda shart operatorlari
5.1. Pythonda if, else, elif operatorlari

Tarmoqlanuvchi algoritmlar – birorta shartga ko‘ra buyruqlar ketma-ketligining bajarilishi yoki bajarilmasligini belgilovchi algoritm. Tarmoqlanuvchi algoritmlarda bir yoki bir necha shartlar tekshiriladi hamda rost yoki yolg‘on qiymat qaytarishiga asoslanib, buyruqlar ketma-ketligi bajariladi.

Shartlarni tekshirish uchun barcha dasturlash tillari kabi Python dasturlash tilida ham shartli o‘tish operatorlari mavjud.

Sintaksisi:

```
if shart:
    buyruqlar_bloki
```

if operatori tarkibidagi shart True (rost) qiymat qaytarsa, buyruqlar_bloki bajariladi. Agar yolg‘on qiymat qaytarsa, buyruqlar_bloki bajarilmaydi.

buyruqlar_bloki if operatoridan keyingi satrda xat boshidan 4 ta probel qoldirib, keyin yoziladi.

buyruqlar_blokidagi buyruqlar alohida qatorda yoki bitta qatorda nuqta, vergul bilan ajratilgan holda yozilishi mumkin.

Pythonda **CASE** tanlash operatori mavjud emasligi sababli, ko‘p shartli masalalarni yechish uchun elif operatoridan foydalaniladi.

CASE operatori – shartdan kelib chiqib, har bir shartga mos buyruqlar ketma-ketligini bajaradigan if ning takomillashgan ko‘rinishi.

elif – else va **if** so‘zlarining kombinatsiyasi bo‘lib, **“aks holda agar”** ma’nosini anglatadi.

```
if shart:
    buyruqlar_bloki
elif shart1:
    buyruqlar_bloki1
....
else:
    buyruqlar_bloki2
```

Dastur natijasini aniqlang.

```
x=1
a=4
b=5
```

```
if a>0:
   if b<0:
      x=x+5
   elif a>4:
      x=x+4
   else:
      x=x+3
else:
      x=x+2
print(x)
```

if shartli o‘tish operatori tarkibida boshqa if shartli o‘tish operatori mavjud bo‘lishi mumkin. Bunday holatga ichma-ich joylashgan shartli o‘tish operatori deyiladi. Ichki ifni ifodalash uchun tashqaridagiga nisbatan bitta xat boshi (4 ta probel) tashlab yozilishi shart, aks holda ifoda ichma-ich joylashmagan, alohida shart operatori hosil qilingan hisoblanadi.

SAVOL VA TOPSHIRIQLAR

1. Shartni tekshirish uchun qaysi operatordan foydalaniladi?
2. Shartli o‘tish operatorining umumiy ko‘rinishi qanday?
3. Dastur tuzish jarayonida shart tekshirish nima uchun kerak?
4. Tarmoqlanish operatorining qisqa va to‘liq ko‘rinishi qanday?
5. Kiritilgan a soni musbat yoki manfiy ekanligini aniqlovchi dastur tuzing.
6. Kiritilgan a soni toq yoki juft ekanligini aniqlovchi dastur tuzing.
7. Tomonlari a va b ga teng to‘g‘ri to‘rtburchak kvadrat ekanligini aniqlovchi dastur tuzing.
8. Kiritilgan a soni to‘rt xonali son ekanligini aniqlovchi dastur tuzing.

5.2. Pythonda takrorlanuvchi operatorlar

Takrorlanuvchi algoritmlar – biror buyruqlar guruhining ma’lum marta yoki belgilangan shart bajarilgunga qadar takroran bajarilishi. Takrorlanuvchi algoritmlarga doir masalalarni dasturlashda sikl operatorlaridan foydalaniladi.

Masalan, n ta sonning musbatligini tekshirish uchun n marotaba bir xil amalni bajarish kerak bo‘ladi. Bunday hollarda bitta amalni n marta yozishdan ko‘ra, bitta kod blokida n ta sonni tekshirish uchun sikl

operatorlaridan foydalangan afzal. Sikl operatorlari kodning takrorlanadigan buyruqlari uchun xizmat qiladi. Bu buyruqlarning ketma-ketligiga siklning tanasi deyiladi. Har bir takrorlanish esa iteratsiya deb ataladi.

for - Takrorlanishlar soni avvaldan ma'lum bo'lganda qo'llaniladi.

while - Takrorlanishlar soni noma'lum bo'lganda, kod hatto bir marta ham ishga tushmasligi mumkin. Kodni ishga tushirishdan avval shart tekshiriladi. Agar shart noto'g'ri bo'lsa, unda sikldagi kod ishga tushmaydi.

Qo'yilgan masalani yechishda sikllarning har ikkala turidan foydalanish mumkin, lekin berilgan shart uchun eng mos keladigan turni to'g'ri tanlay olish dasturning samaraliroq ishlashini ta'minlaydi.

Sintaksisi:

```
for i in range(start, stop, step):
    sikl tanasi
```

i – takrorlanishlar (iteratsiyalar) soni;
start – i ning boshlang'ich qiymati (ko'rsatilmasa, 0 deb qabul qiladi);
stop – i ning oxirgi qiymati (ko'rsatilishi shart);
step – qadam (ko'rsatilmasa, 1 deb qabul qiladi);

Sikl ichida yana siklning ishlatilishiga ichma-ich joylashgan sikl deyiladi.

Sintaksisi:

```
for i in range(start1, stop1, step1):
    for j in range(start2, stop2, step2):
        sikl tanasi
```

i – 1-sikl takrorlanishlari soni;
j – 2-sikl takrorlanishlari soni;
start1– i ning boshlang'ich qiymati (ko'rsatilmasa, 0 deb qabul qiladi);
stop1 – i ning oxirgi qiymati (ko'rsatilishi shart);
step1 – i ning qadami (ko'rsatilmasa, 1 deb qabul qiladi);
start2– j ning boshlang'ich qiymati (ko'rsatilmasa, 0 deb qabul qiladi);
stop2 – j ning oxirgi qiymati (ko'rsatilishi shart);
step2 – j ning qadami (ko'rsatilmasa, 1 deb qabul qiladi).
Tashqi siklning har bir iteratsiyasi uchun j ta iteratsiya bajariladi.
Tashqi siklning i ta iteratsiyasi uchun ichki siklning i*j ta iteratsiyasi bajariladi.
Python. Dastur natijasini toping:

```
s=0
```

```
for i in range(63,21,-41):
    for j in range(1,31,13):
        s+=i//j
print(s)
```

WHILE OPERATORI

while operatori – berilgan shart rost bo‘lgan holda sikl tanasi bajariladigan sikl turi. Agar sikl boshida shart bajarilmasa, u holda sikl ishga tushmaydi. while sikl operatori shart ifodasi bajarilgan holatlar (**True** bo‘lsa) uchun davom etadi, agar shart bajarilmasa (**False** bo‘lsa), sikl o‘z ishini to‘xtatadi.

while shart ifodasi:
    sikl tanasi

while siklidagi shart ifodasi doimo bajarilsa (True qiymat qaytarsa), sikl hech qachon to‘xtamaydi, ya’ni cheksiz davom etishi mumkin.

Eslatma: IDLE interfaol muhitida cheksiz siklni to‘xtatish uchun ***Ctrl+C*** *birgalikda bir necha marta bosiladi*

Dastur natijasini aniqlang:

```
x = 90
while (x < 100):
    x+=3
print(x)
```

SAVOL VA TOPSHIRIQLAR

1. Qaysi operator Python dasturlash tilida hisoblagich ko‘rinishida ishlaydi?
2. Hisoblagich ko‘rinishida ishlovchi sikl operatorining sintaksisi qanday bo‘ladi?
3. Start, stop, step vazifalarini tushuntiring.
4. Qachon hisoblagich ko‘rinishidagi sikl operatorlaridan foydalanib bo‘lmaydi?
5. Ichma-ich joylashgan sikllarda ikkala for operatori bir chiziqda joylashsa, dastur
to‘g‘ri ishlaydimi?
6. a va b sonlar berilgan. a dan b gacha bo‘lgan barcha sonlarni chiqaruvchi dastur
tuzing. Bu yerda a <= b.
7. Natijasi 3;1;-1;-3; ko‘rinishida bo‘ladigan dastur kodini aniqlang.

a) for num in range(3,-3,-2): print(num, end=';')
b) for num in range(3,-5,-3): print(num, end=';')
c) for num in range(2,-5,-2): print(num, end=';')
b) for num in range(3,-5,-2): print(num, end=';')

5.3. Pythonda break va continue operatorlari

break operatori – takrorlanish ishini to‘xtatuvchi operator. Agar shart **True** qiymat qaytarsa ham, sikl ichida **break** operatoriga murojaat etilsa, u holda sikl ishi to‘xtatiladi. Sikl tarkibidagi ixtiyoriy buyruq breakga murojaat etilganidan keyin bekor qilinadi.

continue – joriy siklni o‘tkazib yuborib, keyingisiga o‘tuvchi operator.

Dastur natijasini aniqlang.

```
var = 10
for i in range(10):
   for j in range(2, 10, 1):
      if var % 2 == 0:
         continue
         var += 1
   var+=1
else:
   var+=1
print(var)
```

Python dasturida berilgan kod natijasini aniqlang.

```
a=[]
n=10
for i in range(2,n):
   for j in range(2,i):
      if i%j==0:
         break
      else:
        a.append(1)
print(a)
```

```
for s in 'dars':
    if s=='r':
        break
    print(s)
print('Tugadi')
```

```
for s in 'dars':
    if s=='r':
        continue
    print(s)
print('Tugadi')
```

```
for number in range(10):
    if number == 7:
        break
    print('Sonlar=' + str(number))
print('Takrorlanish tugadi')
```

```
for number in range(10):
    if number == 7:
        continue
    print('Sonlar=' + str(number))
print('Takrorlanish tugadi')
```

SAVOL VA TOPSHIRIQLAR

1. Foydalanuvchi tomonidan kiritilgan sonlar yig‘indisini hisoblash dasturini tuzing. Agar manfiy son kiritilsa, sikl o‘z ishini to‘xtatishi lozim.
2. “Salom” so‘zini 5 marta ekranga chiqarish dasturini tuzing. (break operatori ishtirokida)
3. 1 dan n gacha sonlar yig’indisini hisoblovchi dastur tuzing. Agar takrorlanish 10 ga teng bo‘lganda sikl ishini to‘xtashi lozim.
4. “4+1=?“ savolga to‘g’ri javob berilganda “Barakallo”, aks holda “O‘ylab ko‘ring” deb chiqaruvchi dastur tuzing.
5. Qaysi operatorlar sikl ishini boshqaradi
6. break operatorining vazifasi nima?
7. continue operatorining vazifasi nima?

6. Pythonda funksiya va modullar
6.1. Funksiyalar. Lambda funksiyasi

Dastur tuzish jarayonida ma’lum bir amallar majmuini dasturning turli qismlarida takrorlashga to‘g‘ri keladi. Dasturning mana shu amallar majmuini o‘z ichiga olgan qismi qism dastur deb ataladi. Qism dasturlar ma’lum bir vazifani bajaradi, lekin alohida tizimni tashkil etmaydi.

Qism dasturga murojaat etilganda, unga murojaat etgan asosiy dastur to‘xtaydi va boshqaruv qism dasturga o‘tadi. Qism dastur bajarilishi tugaganidan so‘ng, boshqaruv yana asosiy dasturga qaytadi.

Asosiy dasturda qism dasturlarni chaqirish quyidagi imkoniyatlarni beradi:

- Qism dastur zarurat tug‘ilganda chaqiriladi. U ayni bir kodni bir necha marta yozish zaruratini bartaraf qilib, butun dastur davomida ko‘p marta foydalanilishi mumkin. Bu kodning blokliligini oshiradi, tushunishni osonlashtiradi va xatolarni topishda yordam beradi.
- Xato bor yoki yo‘qligini bitta kod blokining o‘zida tekshirsa bo‘ladi. Agar xato qism dasturda bo‘lsa, faqat qism dasturning o‘zini tuzatishga zarurat tug‘iladi. Agar qism dasturdan foydalanmasdan, kod bir necha joyda takror-takror yozilsa, u holda butun dastur bo‘ylab xatolarni qidirishga to‘gri keladi.
- Kodni faqat bitta joyda yangilash kerak bo‘ladi: Kiritilgan barcha tuzatishlar qism dastur chaqirilishi bilan amal qila boshlaydi.

Qism dasturning turlari

Funksiya – ma’lum bir vazifani bajaruvchi, qandaydir nomga ega, bir yoki bir necha qiymatni qabul qiluvchi, ishni tugatganidan keyin esa asosiy dasturga bir yoki bir necha natija qiymatlarni qaytaruvchi qism dastur.

Protsedura – funksiyaga o‘xshash ko‘p marta foydalanilishi mumkin bo‘lgan qism dastur bo‘lib, yagona farqli jihati hech qanday qiymatni qaytarmaydi. Python dasturlash tilining har xil masalalarni yechishga mo‘ljallangan bir necha foydali standart funksiyalari mavjud.

Standart funksiyalar:

print() – foydalanuvchi uchun ma’lumotlarni chiqaradi. Masalan, turli ma’lumotlar va hisoblash natijalarini.

input() – print() funksiyasining zidi, foydalanuvchilar kiritgan ma’lumotlarni dasturga uzatadi.

randint() – tasodifiy sonni chiqaradi. Masalan, dasturda tasodifiy son kerak bo‘lib qolganda ishlatiladi.

Funksiyani e’lon qilish va chaqirish:

Har bir yaratilgan qism dasturga, xususan, funksiya hamda protseduraga albatta nom berish kerak va bu nom Pythonda define (ing. define – aniqlash) so‘zidan olingan def kalit so‘zi bilan boshlanadi.

```
def funksiya_nomi ([parametrlar ro‘yxati]):
    buyruqlar_bloki
```

def – funksiyani e’lon qiluvchi kalit so‘z.

funksiya_nomi – funksiya nomi.

parametrlar ro‘yxati – ushbu ro‘yxat bir necha parametrdan iborat bo‘lishi mumkin va ular vergul bilan ajratib yoziladi.

buyruqlar_bloki – funksiya tanasi boshqa operatorlar kabi bitta xat boshi tashlab yozilishi shart.

Funksiya nomi orqali chaqirilganda uning tarkibidagi buyruqlar ketma-ketligi bajariladi. Shundan so‘ng dastur funksiya chaqirilgan satrga qaytadi va shu satrdan keyingi buyruqlarga o‘tadi.

Rekursiya

Funksiyaning o‘zini o‘zi chaqirishiga rekursiya deyiladi va bunday funksiyalar rekursiv funksiyalar deb ataladi.

Rekursiv funksiyalar dasturlashning kuchli mexanizmi hisoblanadi, lekin ular har doim ham samarali emas. Chunki aksariyat hollarda xatolarga yo‘l qo‘yadi. Xatolar ichidan eng ko‘p tarqalgani – cheksiz rekursiya. Unda funksiyaning chaqiruv zanjiri cheksiz bo‘lib, kompyuter bo‘sh xotirasi tugamaguncha davom etaveradi. Cheksiz rekursiya ro‘y berishining sabablari:

– rekursiyada shartni noto‘g‘ri qo‘llash. Masalan, faktorialni hisoblashda if n==0 ni unutib qo‘ysak, factorial(0) funksiyasi factorial(–1)ni, factorial(–1) funksiyasi esa factorial(–2) va hokazolarni chaqiradi;

– rekursiv funksiyani noto‘g‘ri parametr bilan chaqirish. Masalan, factorial(n) funksiya factorial(n)ni chaqirsa, yana cheksiz zanjir yuzaga keladi.

Lokal o‘zgaruvchilar – bu o‘zlari e’lon qilingan qism dasturda faol bo‘ladigan o‘zgaruvchilar. Ularni muayyan funksiya doirasida

qo‘llash mumkin, shu sababli faqat shu funksiya doirasidagina amal qiladi. Lokal o‘zgaruvchilardan foydalanish o‘zgaruvchining qiymati dasturning boshqa qismlarida tasodifan o‘zgarib qolishi xavfini kamaytiradi.

Global o‘zgaruvchilar butun dastur davomida faol bo‘ladigan o‘zgaruvchilardir. Ular qism dasturdan tashqarida, ya’ni asosiy dasturda e’lon qilinadi. Odatda, ularni modullarni import qilgandan keyin, kodning boshlanishida e’lon qilish kerak. Ularni odatiy o‘zgaruvchilar kabi chaqirish mumkin.

1. Berilgan to‘rtta sonning eng kichigini topuvchi dastur tuzing. Buning uchun min4 (a, b, c, d) funksiyasini yarating.

```
def min(a,b):
    if a>b:
        return b
    else:
        return a
def min4(a,b,c,d):
    return min(min(min(a,b),c),d)
a=int(input('a='))
b=int(input('b='))
c=int(input('c='))
d=int(input('d='))
print('Eng kichik son=',min4(a,b,c,d))
```

2. Berilgan a (haqiqiy) sonining k (butun) darajasini topuvchi dastur tuzing. Buning uchun daraja (a, k) funksiyasini yarating.

```
def daraja(a,k):
    return a**k
a=float(input('a ni kiriting='))
k=int(input('k darajani kiriting='))
print(daraja(a,k))
```

3. a va b natural sonlari berilgan. a va b sonlaridan kattasini topish funksiyasini tuzing. Funksiyadan foydalanib, a, b va c sonlari ichidan kattasini topish dasturini tuzing.

```
def max(a,b):
    if a>b:
        return a
    else:
        return b
```

```
def max3(a,b,c):
    return max(max(a,b),c)
a=int(input('a='))
b=int(input('b='))
c=int(input('c='))
print('Eng katta son=',max3(a,b,c))
```

4. 0 bilan tugaydigan sonlar ketma-ketligi berilgan. Uning raqamlari yig‘indisini sikl ishlatmagan holda hisoblash dasturini tuzing. Masalan, 1 7 9 0 = 17

```
def raqam(n):
    if n == 0:
        return 0
    return (n % 10 + raqam(int(n / 10)))
n=int(input('son kiriting='))
print(f"{n} = {raqam(n)}")
```

5. Berilgan n sonini rim raqamlarida ifodalovchi dastur tuzing. Funksiyadan foydalaning.

```
a = [(1000, 'M'), (900, 'CM'), (500, 'D'), (400, 'CD'), (100, 'C'), (90, 'XC'),
     (50, 'L'), (40, 'XL'), (10, 'X'), (9, 'IX'), (5, 'V'), (4, 'IV'), (1, 'I')]
def rim(n):
    s = ''
    while n > 0:
        for i, r in a:
            while n >= i:
                s += r
                n -= i
    return s
n=int(input("Son kiriting="))
print(rim(n))
```

Lambda funksiyasi kichik anonim funksiya hisoblanadi. Unda bir qancha argument qatnashishi mumkin.

Ushbu dasturda berilgan sonni 10 marta oshiradigan lambda funksiya yozilgan. Lambda funksiyalarni funksiya ichida boshqa bir anonim funksiya sifatida ishlatish osonroq.

x - bu funksiya (ular lambda deb ataladi). Uning bitta parametri bor: a. Funksiya a ni o‘n baravar oshiradi va natijani qaytaradi.

```
d=lambda a,b:a*b
x=5
y=9
x=d(x,y)
print(x)
```

```
d=lambda a,b:a*b
t=lambda a,b:a+b
x=5
y=9
x=d(x,y)
x=t(x,y)
print(x)
```

```
d=lambda a:a*3
t=lambda a:a*4
x=2
x=d(x)
x=t(x)
x=d(x)
print(x)
```

```
d=lambda a:a*3
t=lambda a:a*4
k=lambda a:a*4
x=2
x=d(x)
x=t(x)
x=k(x)
print(x)
```

SAVOL VA TOPSHIRIQLAR

1. Qism dastur nima?
2. Dasturda protsedura va funksiyalar qanday maqsadda qo‘llaniladi?
3. Qism dasturning qanday turlari mavjud?
4. Qism dastur qanday afzalliklarga ega?
5. Protsedura va funksiyaning farqi nimada?
6. Qachon funksiyaning o‘rniga protsedurani qo‘llash mumkin?

6.2. Pythonda maxsus modullardan foydalanish

Har bir yangi dasturning kodini yozish ko‘p vaqt talab qiladigan jarayon hisoblanadi. Shu sababli, tayyor qism dasturlardan foydalanish har bir dasturchi uchun qulaydir. Zamonaviy dasturlash tillarida bu jarayonni yengillashtirish uchun tayyor dastur kodlarini saqlovchi kutubxonalar mavjud.

Modullar – alohida faylda yozilgan bo‘lib, turli dasturlarda qo‘llanilishi mumkin bo‘lgan kodlar majmui.

Boshqa dasturlash tillari kabi Python dasturlash tilining standart kutubxonasi ham ko‘plab tayyor kod fragmentlari (modullar, standart

funksiyalar va b.)dan tarkib topgan. Python dasturlash tilini yanada takomillashtirish uchun foydalanuvchi tomonidan yozilgan modullarni kutubxonaning alohida qismiga yuklash ham mumkin.

Python dasturlash tili o‘rnatgichidagi Batteries included (батарейки в комплекте – batareykasi bilan) izohi Python dasturlash tili majmuida ko‘plab tayyor kodlar mavjudligini anglatadi.

math - modul murakkab matematik ifodalarni hisoblash uchun qo‘llaniladi.

ceil(x) - eng yaqin katta butun songacha yaxlitlaydi.
round(x) - x sonini yaxlitlaydi.
round(x, n) - x sonini nuqtadan keyin n ta belgi qolgunga qadar yaxlitlaydi.
floor(x) - eng yaqin kichik butun songacha yaxlitlaydi.
log(a, b) - b asosga ko‘ra a logarifmini hisoblaydi.
log10(x) - x sonining o‘nli logarifmini hisoblaydi.
sqrt(x) - x ning kvadrat ildizini hisoblaydi.
pow(x, n) - x ning n-darajasini hisoblaydi.
factorial(x) - x faktorialni hisoblaydi.
abs(x) - *x ning modulini hisoblaydi.*
cos(x) - *x ning kosinusini hisoblaydi.*
degrees(x) - *Radiandan gradusga o‘tkazadi.*
radians(x) - *Gradusdan radianga o‘tkazadi.*

1. Berilgan burchak yoyining uzunligini hisoblovchi dastur tuzing. Yoyning burchagi (gradusda) hamda radiusi foydalanuvchi tomonidan kiritiladi.

```
from math import*
a=int(input('burchakni kiriting='))
r=int(input('radiusni kiriting='))
l=(pi*r**2*a)/180
print('burchak yoyi uzunligi',l)
```

2. Kvadrat tenglamaning ildizlarini hisoblash dasturini tuzing. a, b, c foydalanuvchi tomonidan kiritiladi.

```
#y=ax^2+bx+c
from math import*
a=float(input("a="))
b=float(input("b="))
c=float(input("c="))
if a!=0:
```

```
   d=b**2-4*a*c
if d>=0:
       x1=(-b-sqrt(d))/2*a
       x2=(-b+sqrt(d))/2*a
       print(f"x1={x1}, x2={x2}")
else:
   print("kvadrat tenglama ildizga ega emas")
```

3. Berilgan haqiqiy sonning kasr qismini 1 dan 4 gacha bo‘lgan aniqlikda yaxlitlang.

Berilgan son: 0.26598

Natija:

1-aniqlikda: 0.3

2-aniqlikda: 0.27

3-aniqlikda: 0.266

4-aniqlikda: 0.2660

```
       n=float(input("n="))
       for i in range(1,5):
          print(f'{i}-aniqlik:{round(n,i)}')
```

Random modulining funksiyalari

random() – 0 dan 1 gacha tasodifiy sonlarni hosil qiladi

randint(start, stop) – startdan stopgacha bo‘lgan oraliqdagi tasodifiy sonlarni hosil qiladi.

randrange(start, stop, step) – startdan stopgacha bo‘gan oraliqdagi step qadam bilan tasodifiy sonlarni hosil qiladi.

1. 0 va 1 ning oralig‘idan 10 ta tasodifiy sonni chiqarish dasturini tuzing.

```
       from random import*
       for i in range(1,11):
          print(random())
```

2. 10 va 10000 ning oralig‘idan 5 ta tasodifiy sonni chiqarish dasturini tuzing.

```
       from random import*
       for i in range(1,6):
          print(randint(10,10000))
```

3. 20 va 50 ning oralig‘idan 2 qadam bilan 7 ta tasodifiy sonni chiqarish dasturini tuzing.

```
       from random import*
       for i in range(1,8):
```

```
print(randrange(20,50,2))
```

SAVOL VA TOPSHIRIQLAR

1. Matematik hisoblashlarni amalga oshiruvchi modul qanday nomlanadi?
2. Modul nima va u nima maqsadda ishlatiladi?
3. Dasturlash tili kutubxonasi nima?

6.3. Pythonda fayllar bilan ishlash

Ma’lumotlar fayllarda saqlanadi. Python dasturlash tili yordamida kompyuterdagi har xil fayl turlari bilan ishlash imkoniyati mavjud bo‘lib, shartli ravishda ularni ikki turga bo‘lish mumkin: matn va binar fayllar.

Matn fayllari kengaytmasi cvs, txt, html va hokazolardan iborat, umuman olganda, matn shaklida ma’lumot saqlaydigan barcha fayllarlarni o‘z ichiga oladi.

Binar fayllarni esa tasvirli, audio, video, o‘yin fayllari va boshqalar tashkil etadi. Fayl turiga qarab, fayl bilan ishlash Pythonda biroz farq qilishi mumkin.

Python kutubxonasining ochiq funksiyalari yordamida fayllar bilan ishlash mumkin. Fayllar

bilan ishlaganda, amallar quyidagi ketma-ketlikda bajariladi:

1) open() metodi yordamida faylni ochish;

2) read() metodi yordamida faylni o‘qish yoki write() metodi yordamida faylga yozish;

3) close() metodi yordamida faylni yopish.

Faylni ochish

Faylga ma’lumot yozish yoki uni tarkibini o‘qish uchun fayl open() metodi yordamida ochiladi.

Sintaksisi:

open (file, mode)

open() funksiyasi Python tilida faylni ochish va uning tarkibini ekranda namoyish etish imkonini beradi. Ushbu funksiyaning birinchi parametri sifatida faylning qayerdaligini ko‘rsatuvchi yo‘l ko‘rsatiladi, masalan, D:/project/my.txt. Ikkinchi parametr mode yordamida faylni ochish rejimi, ya’ni fayl ustida qanday ish bajarilishi ko‘rsatiladi.

r (read) - Faylni o‘qish uchun ochish. Agar fayl topilmasa, FileNotFoundError
xatoligini beradi.
w (write) - Faylni qayta yozish uchun ochish. Fayl tarkibini o‘chirib, bo‘sh faylni ochadi. Agar fayl mavjud bo‘lmasa, uni yaratadi.
x (xwrite) - Agar fayl mavjud bo‘lsa, uni yozish uchun ochadi, aks holda ochmaydi.
a (append) - Faylning davomiga yozish uchun ochish. Agar fayl mavjud bo‘lmasa, uni yaratadi.
b (binary) - Faylni binar (ikkilik) rejimda ochish. w va r kabi holatlar bilan birgalikda ishlatiladi. Masalan, ‘rb’, ‘rt’
t (text) - Faylni matnli holatda ochish.+ Faylni o‘qish va yozish uchun ochish.
Fayl bilan ishlash yakunlanganidan so‘ng, uni close() metodi yordamida yopish kerak. Ushbu metod fayl bilan bog‘liq barcha resurs ishlarini yakunlaydi.
Faylga yozish. Matnli faylni yozish uchun ochishda ikki xil rejim ishlatiladi: **a(append)** – fayl davomiga yozuv qo‘shish, **w (write)** – fayl tarkibini o‘chirib, qayta yozish. Faylga yozish uchun write() metodidan foydalaniladi.
Sintaksisi:
write(str)
str parametri orqali satr kiritiladi. Agar boshqa turdagi ma’lumotlar kiritilishi lozim bo‘lsa, ular satr turiga o‘tkaziladi.
Fayldan ma’lumotlarni o‘qish uchun **r (read)** rejimidan foydalaniladi.
Fayl tarkibidagi satrlarni o‘qishda turli usullardan foydalaniladi:
read() – fayl tarkibidagi barcha ma’lumotlarni o‘qiydi;
readline() – faylning faqat birinchi satrini o‘qiydi;
readlines() – faylning barcha satrlarini ro‘yxat elementi sifatida o‘zlashtirib oladi.
Sintaksisi:
file_name.read()
file_name.readline()
file_name.readlines()
file_name o‘rnida ochilgan fayl o‘zlashtirilgan o‘zgaruvchi nomi ko‘rsatiladi.
file=open('d:/dars.txt', 'r')
text=file.readline()

```
text1=text[0]
text2=text[1]
text3=text[2]
text4=text[3]
print(text1,end='')
print(text2,end='')
print(text3,end='')
print(text4)
file.close()
```

with ..as operatori

Fayl bilan ishlashda turli holatlarga duch kelish mumkin. Masalan, fayldan foydalanishga ruxsat berilmagan bo‘lishi mumkin va h. k. Buyday holatlarda dasturning ma’lum bir qatorida xatolik yuz berishi va undan keyingi satrlardagi close() metodi orqali faylning yopilishi bajarilmasligi mumkin. Buning uchun Pythonda with ..as operatori mavjud bo‘lib, bu operator har qanday holatda ham fayl yopilishini ta’minlaydi.

Sintaksisi:

with open(file, mode) as file_name

commands

with orqali ochiq fayl file_name o‘zgaruvchi aniqlanadi hamda commands’da keltirilgan buyruqlar ketma-ketligi bajariladi. So‘ng qanday holat yuzaga kelishidan qat’iy nazar, fayl avtomatik ravishda yopiladi.

```
with open('d:/dars.txt', 'a') as text:
    text.write('\nBuxoro')
    print('\nYaxshi',file=text)
```

1. Faylga ixtiyoriy 5 ta son kiritish va ular yig‘indisini hisoblab, shu fayl davomiga yozish dasturini tuzing.

```
s=0
file=open('d:\dars.txt','w')
for i in range(5):
    k=int(input())
    text=file.write(str(k)+'\n')
    s+=k
text=file.write(str(s))
file.close()
```

2. Faylga kiritilgan she’r satrlarini teskari tartibda boshqa faylga yozish dasturini tuzing. Dasturda readlines() metodidan foydalaning.

```
file=open('d:/dars.txt', 'w')
for i in range(4):
    k=input()
    text=file.write(k+'\n')
file.close()
file2=open('d:/dars.txt', 'r')
file3=open('d:/dars2.txt', 'w')
text1=file2.readlines()
for i in range(4):
    text2=file3.write(str(text1[3-i]))
file2.close()
file3.close()
```

3. To‘rt qatorli she’rni faylga yozish va uni o‘qish dasturini tuzing.

```
file=open('d:/dars.txt', 'w')
for i in range(4):
    k=input()
    text=file.write(k+'\n')
file.close()
file2=open('d:/dars.txt', 'r')
text2=file2.readlines()
for i in range(4):
    print(text2[i],end='')
file2.close()
```

4. Fayldagi satrda @ belgisi mavjud yoki mavjud emasligini aniqlovchi dastur tuzing.

```
file=open('d:/dars.txt', 'w')
for i in range(4):
    k=input()
    text=file.write(k+'\n')
file.close()
file2=open('d:/dars.txt', 'r')
text2=file2.readlines()
for i in range(4):
    for j in range(0,len(text2[i])):
        if text2[i][j]=='@':
            print(f'{i}-satrda bor')
```

```
            break
        elif len(text2[i])-1==j:
            print(f'{i}-satrda yo‘q')
file2.close()
```

SAVOL VA TOPSHIRIQLAR

1. Fayllar nima maqsadda ishlatiladi?
2. Faylni ochish uchun qaysi buyruqdan foydalaniladi?
3. Faylning davomiga yozish imkoniyati mavjudmi, agar mavjud bo‘lsa, qaysi rejim orqali amalga oshiriladi?
4. Faylni avtomatik tarzda yopish uchun qaysi operatordan foydalaniladi?

7. Pythonda grafika
7.1. Grafika bilan ishlash operatorlari

Aksariyat dasturlash tillarida foydalanuvchi bilan o‘zaro aloqani o‘rnatish uchun boshqaruv elementlari: oyna, matnlar maydoni va tugmachalar ishlatiladi. Bular umumiy nom bilan foydalanuvchining grafik interfeysi (GUI – graphical user interface) deb ataladi.

Widget (vijet) – GUIga ega ilovani yaratish uchun foydalaniladigan tugmachalar yoki matnli maydonlar kabi interfeys elementlari.

Barcha elementlar joylashadigan oyna GUIning asosi hisoblanadi. Oyna va uning elementlari (vijetlar)ni yaratish uchun Python standart kutubxonasining Tkinter modulidan foydalaniladi.

Tkinter – Pythondagi standart grafik kutubxona. Pythonni o‘rnatganda kutubxona dasturning ichida birga taqdim etiladi. Python o‘rnatilishi bilan GUIga ega ajoyib ilovalarni yaratish uchun zarur obyekt va usullardan foydalanish imkoniyati vujudga keladi. GUI ilovalarni yaratish uchun:

• Tkinter modulini import qilish;
• Tkinter asosiy oynasini yaratish;
• ilovaga bir yoki bir necha vijetni qo‘shish;
• foydalanuvchi bajaradigan jarayonlarni tushunadigan va ularga javob qaytaradigan asosiy siklli kodga kirish lozim.

1. Rangi yashil, o‘lchami 100×100 bo‘lgan “Mening birinchi ilovam” nomli GUI oynasini yaratuvchi dastur kodini yozing. Oynada “Salom O‘zbekiston” xabarini chiqaruvchi tugmachasini joylashtiring.

```
from tkinter import*
window = Tk()
window.title('Mening birinchi ilovam')
window.geometry('300x200')
window.configure(background='green')
matn=Label(window, width=25,height=5,bg='green',text='')
matn.grid(row=0, column=0)
def oyna():
   matn.config(text='Salom O‘zbekiston!')
tugma=Button(window,text="Tanlang",width=10,command=oyna)
tugma.grid(row=1,column=0)
```

```
window.mainloop()
```

2. Rangi pushti, o‘lchami 250×150 bo‘lgan “Mevalar” nomli GUI oynasini yaratuvchi dastur kodini yozing. Oynada berilgan 4 ta mevadan birini tanlash imkonini beruvchi vijetni joylashtiring.

```
from tkinter import *
window = Tk()
window.title('Mevalar')
window.geometry('250x150')
window.configure(background='green')
options=('Olma','Anor','Nok','Gilos')
a=IntVar()
a.set('Tanlang:')
b=OptionMenu(window, a, *options)
b.grid()
window.mainloop()
```

SAVOL VA TOPSHIRIQLAR

1. Foydalanuvchi grafik interfeysi (GUI) nima?
2. Vijet nima va u nima maqsadda ishlatiladi?
3. GUI ilovalarga vijet qanday qo‘shiladi?
4. Ilova oynasida vijet qanday joylashtiriladi, qaysi usul yordamida?
5. Tugmachalar yordamida funksiya qanday ishga tushiriladi?

7.2. Grafika bilan ishlovchi funksiyalar

Dastur ilovalarida foydalaniladigan barcha vijetlar **window = Tk()** va window.mainloop() buyruqlari orasida kiritiladi.

Tkinter modulidagi **grid()** usuli katakli koordinatalar tizimidan foydalangan holda vijetlarni kerakli koordinataga joylashtirish imkonini beradi.

Label() - Matnli maydon

Text - Natijani chiqarish uchun matnli maydon

Entry() - Matn kiritiladigan maydon

OptionMenu() - Tanlanadigan maydon

1. GUIdan foydalanib, berilgan songa karrali 10 ta sonni chiqaruvchi dastur tuzing. Berilgan son foydalanuvchi tomonidan kiritiladi.

```
from tkinter import*
def karra():
   n=int(a.get())
   b.delete(0.0,END)
   for i in range (1,11):
      son=str(i*n)+'\n'
      b.insert(END,son)
window=Tk()
window.title('Karrali sonlar')
window.geometry('250x250')
window.configure(background='orange')
c=Label(window,text='Son: ', bg='orange')
c.grid(row=0, column=0)
d=Label(window, text='\n Natija', bg='orange')
d.grid(row=2, column=0)
a=Entry(window, width=5)
a.grid(row=1, column=0)
b=Text(window,height=10,width=6)
b.grid(row=3, column=0)
t=Button(window,text='Karrali sonlar', command=karra)
t.grid(row=1,column=1)
window.mainloop()
```

2. GUIdan foydalanib, berilgan songa karrali sonlarni chiqaruvchi dastur tuzing. Berilgan son va karrali sonlarning soni foydalanuvchi tomonidan kiritiladi.

```
from tkinter import*
def karra():
   n=int(a.get())
   m=int(b.get())
   c.delete(0.0,END)
   for i in range (1,m+1):
      son=str(i*n)+'\n'
      c.insert(END,son)
window=Tk()
window.title('Karrali sonlar')
window.geometry('250x350')
```

```
window.configure(background='yellow')
k=Label(window,text='Son: ', bg='yellow')
k.grid(row=0, column=0)
j=Label(window, text='\n Natija', bg='yellow')
j.grid(row=2, column=4)
a=Entry(window, width=5)
a.grid(row=1, column=0)
karra_soni=Label(window,text='Karra_soni: ', bg='yellow')
karra_soni.grid(row=0, column=4)
b=Entry(window, width=5)
b.grid(row=1, column=4)
c=Text(window,height=15,width=6)
c.grid(row=3, column=4)
t=Button(window,text='Karrali sonlar', command=karra)
t.grid(row=1,column=5)
window.mainloop()
```

3. GUIdan foydalanib, n ta tub sonni chiqaruvchi dastur tuzing (1 ga va o‘ziga bo‘linadigan songa tub son deyiladi). n foydalanuvchi tomonidan kiritiladi.

```
from tkinter import*
import math
def tub():
    n=int(m.get())
    d.delete(0.0,END)
    b=2
    s=0
    a=True
    while a:
        for i in range(2, int(math.sqrt(b) + 1)):
            if b%i==0:
                break
        else:
            s+=1
            satr=str(b)+'\n'
            d.insert(END,satr)
        b+=1
        if n==s:
            a=False
```

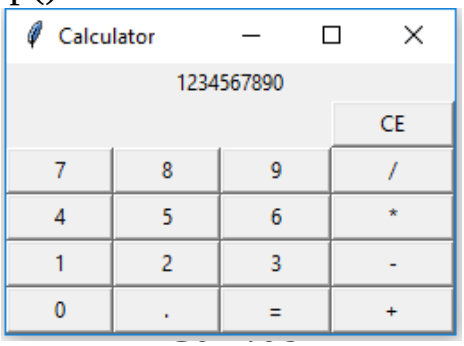
```
window=Tk()
window.title('Tub sonlar')
window.geometry('250x250')
window.configure(background='pink')
k=Label(window,text='Son kiriting: ', bg='pink')
k.grid(row=0, column=0)
j=Label(window, text='\n Natija', bg='pink')
j.grid(row=2, column=0)
m=Entry(window, width=5)
m.grid(row=1, column=0)
d=Text(window,height=10,width=6)
d.grid(row=3, column=0)
t=Button(window,text='Tub sonlar', command=tub)
t.grid(row=1,column=1)
window.mainloop()
```

4. . GUIdan foydalanib, kalkulyator hosil qiling.

```
from tkinter import *
from decimal import *
window = Tk()
window.title('Calculator')
buttons = (('7', '8', '9', '/'),
           ('4', '5', '6', '*'),
           ('1', '2', '3', '-'),
           ('0', '.', '=', '+')
           )
satr = ''
t = []
def calculate():
    global t
    global label
    natija = 0
    a = Decimal(t.pop())
    amal = t.pop()
    b = Decimal(t.pop())
    if amal == '+':
        natija = b + a
    if amal == '-':
        natija = b - a
```

```
    if amal == '/':
        natija = b / a
    if amal == '*':
        natija = b * a
    label.configure(text=str(natija))
def click(text):
    global satr
    global t
    if text == 'CE':
        t.clear()
        satr = ''
        label.configure(text='0')
    elif '0' <= text <= '9':
        satr += text
        label.configure(text=satr)
    elif text == '.':
        if satr.find('.') == -1:
            satr += text
            label.configure(text=satr)
    else:
        if len(t) >= 2:
            t.append(label['text'])
            calculate()
            t.clear()
            t.append(label['text'])
            satr = ''
            if text != '=':
                t.append(text)
        else:
            if text != '=':
                t.append(label['text'])
                t.append(text)
                satr = ''
                label.configure(text='0')
label = Label(window, text='0', width=35)
label.grid(row=0, column=0, columnspan=4, sticky="nsew")
button = Button(window, text='CE', command=lambda text='CE':
                click(text))
```

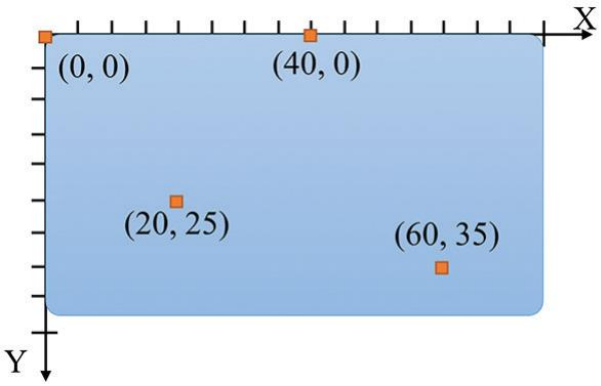
```
button.grid(row=1, column=3, sticky="nsew")
for row in range(4):
    for col in range(4):
        button = Button(window, text=buttons[row][col],
                command=lambda row=row, col=col:
                        click(buttons[row][col]))
        button.grid(row=row + 2, column=col, sticky="nsew")
window.mainloop()
```



Python dasturlash tilida tasvir chizish uchun Canvas deb nomlanuvchi chizish maydonidan foydalaniladi.

Canvas – Python dasturlash tilida X va Y koordinatalar tizimidan iborat maxsus tasvir chizish maydoni.

Tkinterda X koordinatalari chapdan o‘ngga, Y koordinatalari esa yuqoridan pastga qarab belgilanadi. (0,0) nuqta Canvasning yuqori chap burchagi hisoblanadi.



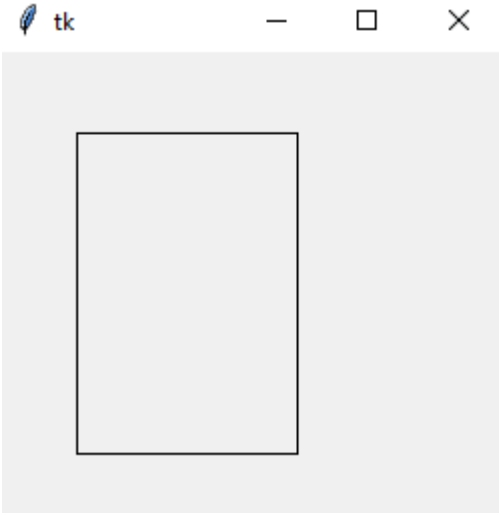
Python dasturlash tilida tasvir chizish uchun Canvas deb nomlanuvchi chizish maydonidan foydalaniladi.

Canvas – Python dasturlash tilida X va Y koordinatalar tizimidan iborat maxsus tasvir chizish maydoni.

Tkinterda X koordinatalari chapdan o‘ngga, Y koordinatalari esa yuqoridan pastga qarab belgilanadi. (0,0) nuqta Canvasning yuqori chap burchagi hisoblanadi.

create_rectangle() – funksiyasi to‘g‘ri to‘rtburchak chizadi.

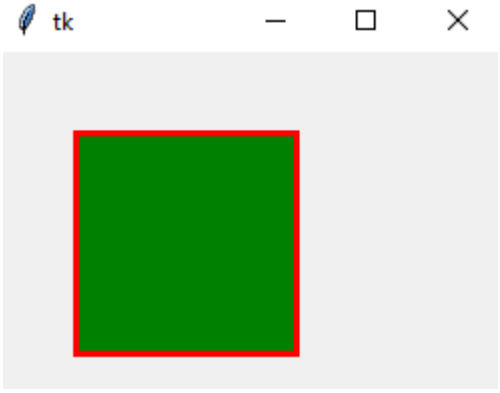
```
from tkinter import*
window=Tk()
a=Canvas(window, width=250, height=300)
a.pack()
a.create_rectangle(40,40,150,200)
```



fill – metodi orqali tanlangan sohani bo‘yash.

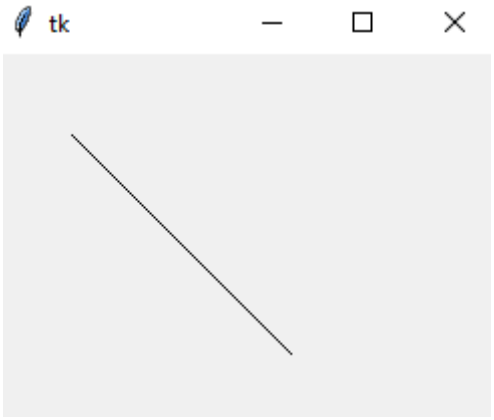
outline – metodi orqali soha chegara chizig’i rangini tanlash mumkin.

```
from tkinter import*
window=Tk()
a=Canvas(window, width=250, height=300)
a.pack()
a.create_rectangle(40,40,150,150, fill='green', outline='red', width=3)
```



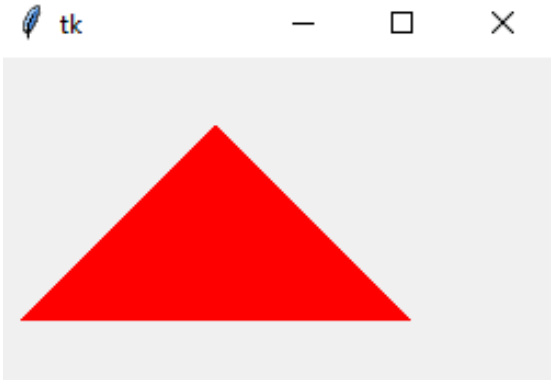
create_line() – *to‘g’ri chiziq chizadi*.

```
from tkinter import*
window=Tk()
a=Canvas(window, width=250, height=300)
a.pack()
a.create_line(40,40,150,150)
```



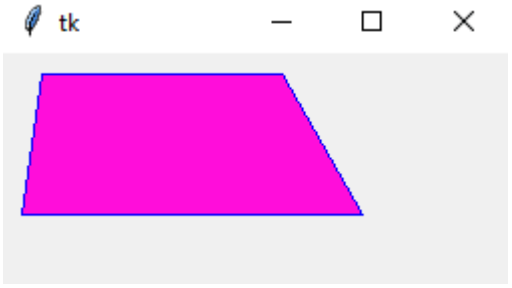
create_polygon() – *ixtiyoriy ko‘pburchak.*

```
from tkinter import*
window=Tk()
a=Canvas(window, width=250, height=300)
a.pack()
a.create_polygon(100,30,10,120,190,120, fill='#FF0000')
```



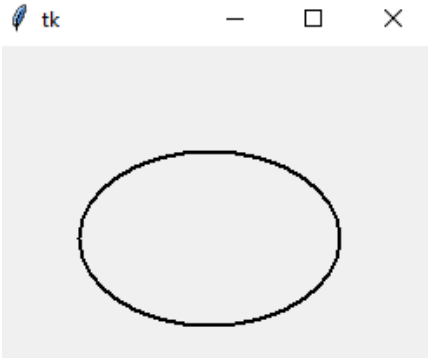
create_polygon() – *ixtiyoriy ko‘pburchak.*

```
from tkinter import*
window=Tk()
a=Canvas(window, width=250, height=300)
a.pack()
a.create_polygon(20, 10, 140, 10, 180, 80, 10, 80, fill='#FF0DDA',
outline='blue')
```



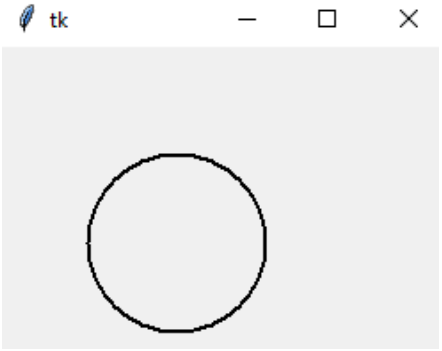
create_oval() – *oval chizadi.*

```
from tkinter import*
window=Tk()
a=Canvas(window, width=250, height=300)
a.pack()
a.create_oval(50,60,200,160, width=2)
```



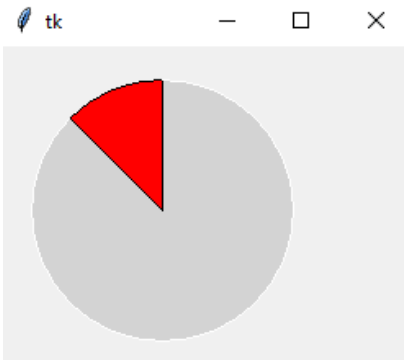
create_oval() – *oval chizadi.*

```
from tkinter import*
window=Tk()
a=Canvas(window, width=250, height=300)
a.pack()
a.create_oval(50, 60, 150, 160, width=2)
```



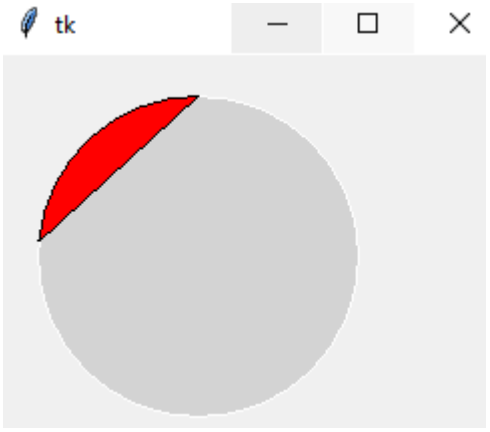
create_arc – *sektor, segment, yoy uzunligi chizadi. start - sektor boshlangan gradus, extent qo‘shilgan burchak gradusi.*

```
from tkinter import*

window=Tk()

a=Canvas(window, width=250, height=300)

a.pack()

a.create_oval(20,20,180,180, fill='lightgrey', outline='white')

a.create_arc(20,20,180,180,start=90, extent=45, fill='red')
```



style=CHORD segment ekanligini ifodalaydi

```
from tkinter import*
window=Tk()
a=Canvas(window, width=250, height=300)
a.pack()
a.create_oval(20, 20, 180, 180, fill='lightgrey', outline='white')
a.create_arc(20, 20, 180, 180, start=90, extent=85,style=CHORD, fill='red')
```



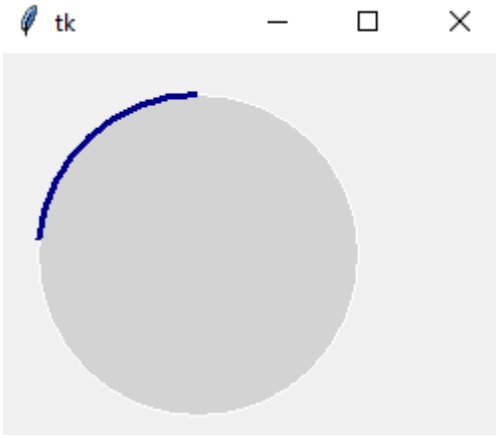
style=ARC yoy ekanligini ifodalaydi.

outline='darkblue' yoy chegarasi.

```
from tkinter import*
window=Tk()
a=Canvas(window, width=250, height=300)
a.pack()
a.create_oval(20, 20, 180, 180, fill='lightgrey', outline='white')
```

a.create_arc(20, 20, 180, 180, start=90, extent=85,style=ARC,

outline='darkblue', fill='red', width=3)



create_text() – *oval chizadi.*

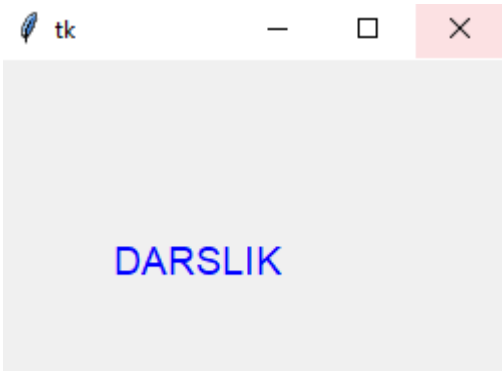
from tkinter import*

window=Tk()

a=Canvas(window, width=250, height=300)

a.pack()

a.create_text(100,100,text="DARSLIK",justify=CENTER,font="ARIAL 14",fill="Blue")



Sichqoncha ko‘rsatkichini chiziq ustiga olib kelganda, och yashil ko‘rinishda hosil bo‘ladi.

from tkinter import *

win = Tk()

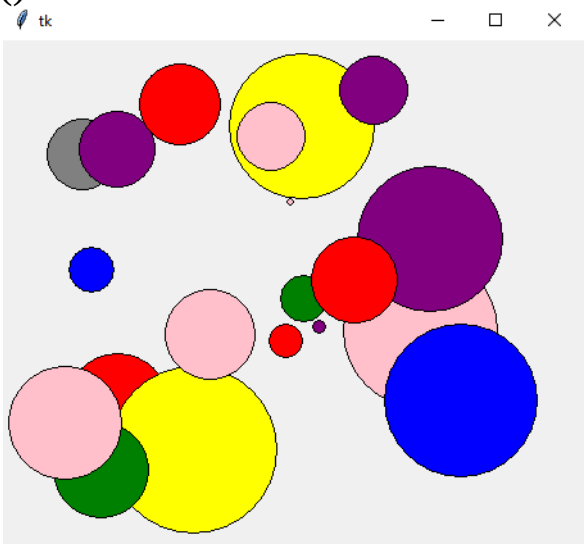
c = Canvas(win, width=200, height=200, bg='lightyellow')

c.pack()

```
c.create_line(20, 50, 100, 50,  fill='green',  width=5,
              arrow=LAST,          dash=(3,4),          activefill='lightgreen',
arrowshape="10 20 10")
win.mainloop()
```

Radiusi tasodifiy qiymatga teng 20 ta doira chizing. Doira ichini ranglar to‘plamidagi tasodifiy rangga bo‘yang.

```
from tkinter import *
from random import *
window=Tk()
canvas=Canvas(window, width=400, height=400)
canvas.pack()
for i in range(1,21):
    color=choice(['yellow', 'red','green', 'blue', 'pink', 'grey', 'purple'])
    x0=randint(0, 300)
    y0=randint(0, 300)
    d=randint(0, 150)
    canvas.create_oval(x0,y0,x0+d,y0+d, fill=color)
window.mainloop()
```



Chizish buyruqlari buyruqlari

width(n)
Toshbaqa izining kengligini n pikselga o'rnatish.
color(s)
Toshbaqa izi rangini s ga o‘rnating. s "qizil", "sariq", "yashil" va hokazo kabi rang nomi (ingliz tilida) bilan qo'shtirnoq ichiga olingan matn qatori bo'lishi kerak.

```
from turtle import*
window=Turtle()
width(3)
color('red')
forward(100)
```

```
from turtle import*
window=Turtle()
width(3)
color('red')
for i in range(1,9):
    forward(100)
    left(45)
    backward(100)
    left(90)
```

```
from turtle import*
window=Turtle()
width(5)
color('red')
for i in range(1,9):
    forward(100)
    left(45)
    backward(100)
    right(90)
```

```
from turtle import*
window=Turtle()
color('red', 'yellow')
begin_fill()
for i in range(1,5):
    forward(100)
    left(90)
end_fill()
```

fill(f)
To'ldirilgan joylarni chizish uchun ishlatiladi. To'ldirilgan maydonni chizishni boshlab uchun begin_fill() buyrug‘i, maydonni chizish tugatish uchun end_fill() buyrug'ini beriladi.

```
from turtle import*
window=Turtle()
color('red', 'yellow')
begin_fill()
a=1
while a<37:
    forward(200)
    left(170)
    a+=1
end_fill()
```

Takrorlanish operatorlariga doir dasturlar

1. 0 dan 20 gacha bo‘lgan juft sonlarni ekranga chiqaruvchi dastur tuzing.

```
a=1
while a<20:
    if (a%2==0):
        print(a,end=",")
    a+=1
```

2. n va k butun manfiy bo‘lmagan sonlar berilgan. n va k qatnashgan ushbu ifodani hisoblang. $\frac{n!}{k!(n-k)!}$

```
from math import factorial
n=int(input("n ni kiriting="))
k=int(input("k ni kiriting="))
print(factorial(n)/(factorial(k)*factorial(n-k)))
```

```
from math import factorial
n=int(input("n ni kiriting="))
k=int(input("k ni kiriting="))
if n>k:
    print(factorial(n)/(factorial(k)*factorial(n-k)))
else:
    print(f"{n} son {k} sondan kichik bo`ldi")
```

3. a va b natural sonlar berilgan. a dan b gacha bo‘lgan sonlar orasidan faqat juftlarini chiqaruvchi dastur tuzing. Bu yerda a <= b.

```
a=int(input("a ni kiriting="))
b=int(input("b ni kiriting="))
while a<=b:
    if (a%2==0):
        print(a,end=",")
    a+=1
```

5. n natural soni berilgan. Kvadrati n dan kichik bo‘lgan barcha natural sonlarni chiqaruvchi dastur tuzing.

```
import math
n=int(input("n ni kiriting="))
i=1
while i<math.sqrt(n):
    print(i)
    i+=1
```

5. S = 0,5 + 1,5 + 2,5 +…+ 98,5 + 99,5 ifodani hisoblash dasturini tuzing.

```
i=0
s=0
while i<100:
    s=s+(0.5+i)
    i+=1
print("yig`indi=",s)
```

6. n natural sonining barcha bo‘luvchilarini chiqaruvchi dastur tuzing.

```
n=int(input("n natural sonni kiriting="))
i=1
while i<=n:
    if n%i==0:
        print(i)
    i+=1
```

7. Kiritilgan n soni qancha raqamdan iborat ekanligini aniqlovchi dastur tuzing (ko‘rsatma: n = n//10 ifoda n = 0 bo‘lguncha bajariladi).

```
n=int(input("n natural sonni kiriting="))
i=0
while n>0:
    n=n//10
    i+=1
print(f"{i} ta raqamdan iborat")
```

8. Kiritilgan n soni raqamlari yig‘indisini hisoblovchi dastur tuzing.

```
n=int(input("n natural sonni kiriting="))
i=0
while n>0:
    m=n%10; n=n//10;
    i+=m
print(f"{i} ta raqamdan iborat")
```

9. Kiritilgan n soni juft raqamlari sonini hisoblovchi dastur tuzing.

```
n=int(input("n natural sonni kiriting="))
i=0
while n>0:
    m=n%10; n=n//10;
    if m%2==0:
        i+=1
print(f"Juft raqamlar soni= {i}")
```

10. n natural soni berilgan. 1 dan n gacha bo‘lgan natural sonlar ichida oxirgi raqami 3 ga karrali sonlarni chiqaruvchi dastur tuzing.

```
n=int(input("n natural sonni kiriting="))
i=1
while i<n:
    if (i%10)%3==0:
        print(i)
    i+=1
```

CONTINUE va BREAK

1) Foydalanuvchi tomonidan kiritilgan sonlar yig‘indisini hisoblash dasturini tuzing. Agar manfiy son kiritilsa, sikl o‘z ishini to‘xtatishi lozim.

```
s=0
a=int(input("Nechta son kiritmoqchisiz="))
for i in range(a):
    k=int(input())
    if k<0:
        break
    else:
        s=s+k
print("yig`indi=",s)
```

2) “Salom” so‘zini 5 marta ekranga chiqarish dasturini tuzing. (break operatori ishtirokida)

```
a=0
while a >= 0:
   if a == 5:
      break
   a += 1
   print("Salom")
print("Takrorlanish to`xtadi")
```

3) 1 dan n gacha sonlar yig’indisini hisoblovchi dastur tuzing. Agar takrorlanish 10 ga teng bo‘lganda sikl ishini to‘xtashi lozim.

```
n=int(input("n sonni kiriting="))
s=0
for i in range(1,n):
    if i==10:
        break
    s+=i
print(s)
```

4) “4+1=?“ savolga to‘g’ri javob berilganda “Barakallo”, aks holda “O‘ylab ko‘ring” deb chiqaruvchi dastur tuzing.

```
a=0
while a==0:
    print('4+1=')
    k=int(input())
    if k==5:
        print('Barakallo')
        break
    else:
        print('O`ylab ko`ring')
print('Takrorlanish tugadi')
```

5) Qo‘shish, ayirish, ko‘paytirish va bo‘lish amallaridan iborat sodda kalkulyator dasturini tuzing.

```
a=int(input("a="))
k=input("Bajarish amalini kiriting: ")
b=int(input("b="))
if k=="+":
    print(a+b)
elif k=="-":
    print(a-b)
elif k=="*":
    print(a*b)
elif k=="/":
    print(a/b)
else:
    print("Bajarish amali belgisini xato kiritildi")
```

6) Foydalanuvchi tomonidan kiritilgan sonlar yig‘indisini hisoblash dasturini tuzing. Agar manfiy son kiritilsa, sikl o‘z ishini to‘xtatishi lozim.

```
s=0
a=int(input("Nechta son kiritmoqchisiz="))
for i in range(a):
    k=int(input())
    if k<0:
        break
    else:
        s=s+k
print("yig`indi=",s)
```

Pythonda modullar

1. n va k butun musbat sonlar berilgan. n va k qatnashgan ushbu ifodani hisoblang. n!/(k!(n-k)!) (Funksiyadan foydalaning)

```
from math import factorial
def binomial(n,k):
      return factorial(n)/(factorial(k)*factorial(n-k))
n=int(input('n sonini kiriting='))
k=int(input('k sonini kiriting='))
print(binomial(n,k))
```

2. n natural son berilgan. Kvadrati n dan kichik bo‘lgan barcha natural sonlarni chiqaruvchi dastur tuzing.

```
import math
def kvadrat(n):
    i=1
    while i<math.sqrt(n):
        print(i)
        i+=1
    return  "Takrorlanish tugadi"
n=int(input("n ni kiriting="))
print(kvadrat(n))
```

3. Bir birlik uzunlik ‘-’ ga teng. Berilgan n uzunlikdagi ‘-’ belgidan iborat chiziq chizuvchi dastur tuzing. Protseduradan foydalaning.

Kiruvchi ma’lumot	Chiquvchi ma’lumot
n	*n* ta chiziq (‘-’)
5	-----

```
def msg():
 print('-'*n)
n=int(input('n ni kiriting='))
msg()
```

Funksiyalar va protseduralar

1. ‘*’ belgisidan tomoni n ga teng kvadrat chizuvchi dastur tuzing. Protseduradan foydalaning.

Kiruvchi ma’lumot	Chiquvchi ma’lumot
n	n x n ta (‘*’) dan iborat kvadrat
3	*** *** ***

```
def yulduz():
    for i in range(1,n+1):
        print('*'*n,end=" "+"\n")
n=int(input("n ni kiriting="))
yulduz()
```

2. Berilgan n sonining bo‘luvchilarini bitta qatorda probel orqali ajratib chiqaruvchi dastur tuzing. Protseduradan foydalaning.

Kiruvchi ma’lumot	Chiquvchi ma’lumot
6	1 2 3 6

```
def son():
    for i in range(1,n+1):
        if n%i==0:
            print(i,end=" ")
n=int(input('n ni kiriting='))
son()
```

3. Berilgan n sonini rim raqamlarida ifodalovchi dastur tuzing. Funksiyadan foydalaning.

Kiruvchi ma’lumot	Chiquvchi ma’lumot
125	CXXV

```
a = [(1000, 'M'), (900, 'CM'), (500, 'D'), (400, 'CD'), (100, 'C'), (90, 'XC'),
     (50, 'L'), (40, 'XL'), (10, 'X'), (9, 'IX'), (5, 'V'), (4, 'IV'), (1, 'I')]
def rim(n):
    s = ''
    while n > 0:
        for i, r in a:
            while n >= i:
                s += r
                n -= i
    return s
n=int(input("Son kiriting="))
print(rim(n))
```

4. Berilgan n soni xonalarining yig‘indisini hisoblash dasturini tuzing. Funksiyadan foydalaning.

Kiruvchi ma’lumot	Chiquvchi ma’lumot
125	8

```
def son(n):
    s=0
    while n>0:
        m=n%10
        n=n//10
        s=s+m
    return s
n=int(input('n sonini kiriting='))
print('yig`indi=',son(n))
```

Kiruvchi ma’lumot	Chiquvchi ma’lumot
1 2 3 4 5	15 3.00

5. Sport musobaqasida sportchilarning chiqishlari hakamlar tomonidan ballik tizimda baholandi. Yakuniy ballni olish uchun eng yuqori va eng past ball chiqarib tashlandi va qolgan uchta ballning o‘rtacha arifmetik qiymati hisoblandi. 5 nafar hakam tomonidan berilgan eng yuqori va

eng past ballni hamda sportchi olgan ballni chiqaradigan dastur tuzing. Funksiyadan foydalaning.

```
def sport(n):
    a=n[0]
    b=n[4]
    s=(int(n[1])+int(n[2])+int(n[3]))/3
    print(a+b)
    return s
n=input().split()
print(sport(n))
```

(sonlar probel bilan kiritiladi, masalan: 1 2 3 4 5)

6. Berilgan n soni xonalari raqamlarining sonini chiqaruvchi dastur tuzing. Funksiyadan foydalaning.

Kiruvchi ma’lumot	Chiquvchi ma’lumot
647521	6

```
def raqam(n):
    s=0
    while n>0:
        n=n//10
        s+=1
    return s
n=int(input('n sonini kiriting='))
print("Raqamlar soni=",raqam(n))
```

7. n natural soni berilgan. S=1*5+2*6+3*7+…+n*(n+4) ifodani hisoblash dasturini protsedura yordamida tuzing.

```
def hisoblash():
    s=0
    for i in range(1,n+1):
        s+=i*(i+4)
    print("Yig`indi=",s)
n=int(input("n sonni kiriting="))
hisoblash()
```

8. a va b natural sonlari berilgan. a va b sonlaridan kattasini topish funksiyasini tuzing. Funksiyadan foydalanib, a, b va c sonlari ichidan kattasini topish dasturini tuzing.

```
def max(a,b):
    if a>b:
        return a
    else:
        return b
def max3(a,b,c):
    return max(max(a,b),c)
a=int(input('a='))
b=int(input('b='))
c=int(input('c='))
print('Eng katta son=',max3(a,b,c))
```

9. Berilgan to‘rtta sonning eng kichigini topuvchi dastur tuzing. Buning uchun min4 (a, b, c, d) funksiyasini yarating.

```
def min(a,b):
    if a>b:
        return b
    else:
        return a
def min4(a,b,c,d):
    return min(min(min(a,b),c),d)
a=int(input('a='))
b=int(input('b='))
c=int(input('c='))
d=int(input('d='))
print('Eng kichik son=',min4(a,b,c,d))
```

10. Berilgan a (haqiqiy) sonining k (butun) darajasini topuvchi dastur tuzing. Buning uchun daraja (a, k) funksiyasini yarating.

```
def daraja(a,k):
    return a**k
a=float(input('a ni kiriting='))
k=int(input('k darajani kiriting='))
print(daraja(a,k))
```

11. Inglizcha harf va raqamlardan iborat satr berilgan. Ushbu satr qancha raqamdan iboratligini hisoblovchi dastur tuzing.

Kiruvchi ma’lumot	Chiquvchi ma’lumot
jdf423h4545b5213b8u58hkj2k32	17

```
s=input('satr kiriting=')
raqam = 0
harf = 0
for i in s:
    #  agar belgi raqamli bo'lsa (qaytarish True)
    if i.isnumeric():
        raqam += 1
    # agar belgi harf bo'lsa (False qaytish)
    else:
        harf += 1
print("Umumiy raqamlar :-", raqam)
print("Umumiy harflar :-", harf)
```

12. Berilgan sonni akslantiruvchi dastur tuzing. Masalan, 123 dan 321 ni hosil qiling. Funksiyadan foydalaning.

```
def akslantirish(n):
    s=''
    while n>0:
        m=n%10
        n=n//10
        s+=str(m)
    return s
n=int(input('n='))
print(akslantirish(n))
```

13. 0 bilan tugaydigan sonlar ketma-ketligi berilgan. Uning raqamlari yig‘indisini sikl ishlatmagan holda hisoblash dasturini tuzing. Masalan, 1 7 9 0 = 17

```
def raqam(n):
    if n == 0:
        return 0
    return (n % 10 + raqam(int(n / 10)))
n=int(input('son kiriting='))
print(f"{n} = {raqam(n)}")
```

14. Tonna, kilogramm va grammlarda berilgan birlikni grammga aylantirish funksiyasini yozing

Kiruvchi ma’lumot	Chiquvchi ma’lumot
tonna=14 kg=32 g=125	14032125 g

```
def gramm(t,k,g):
    g=1000000*t+1000*k+g
    return g
t=int(input('tonna='))
k=int(input('kilogramm='))
g=int(input('gramm='))
print(gramm(t,k,g),'g')
```

Random va Math modullar

1. 0 va 1 ning oralig‘idan 10 ta tasodifiy sonni chiqarish dasturini tuzing.

```
from random import*
for i in range(1,11):
    print(random())
```

2. 10 va 10000 ning oralig‘idan 5 ta tasodifiy sonni chiqarish dasturini tuzing.

```
from random import*
for i in range(1,6):
    print(randint(10,10000))
```

3. 20 va 50 ning oralig‘idan 2 qadam bilan 7 ta tasodifiy sonni chiqarish dasturini tuzing.

```
from random import*
for i in range(1,8):
    print(randrange(20,50,2))
```

4. Berilgan burchak yoyining uzunligini hisoblovchi dastur tuzing. Yoyning burchagi (gradusda) hamda radiusi foydalanuvchi tomonidan kiritiladi.

```
from math import*
a=int(input('burchakni kiriting='))
r=int(input('radiusni kiriting='))
l=(pi*r**2*a)/180
print('burchak yoyi uzunligi',l)
```

5. y = x * cos x funksiyasi qiymatini hisoblovchi dastur tuzing. x foydalanuvchi tomonidan kiritiladi.

```
from math import*
x=int(input('x ni kiriting='))
y=x*cos(x)
print('Natija:',y)
```

6. Kvadrat tenglamaning ildizlarini hisoblash dasturini tuzing. a, b, c foydalanuvchi tomonidan kiritiladi.

```
#y=ax^2+bx+c
from math import*
a=float(input("a="))
b=float(input("b="))
c=float(input("c="))
if a!=0:
    d=b**2-4*a*c
if d>=0:
        x1=(-b-sqrt(d))/2*a
        x2=(-b+sqrt(d))/2*a
        print(f"x1={x1}, x2={x2}")
else:
    print("kvadrat tenglama ildizga ega emas")
```

7. Berilgan haqiqiy sonning kasr qismini 1 dan 4 gacha bo‘lgan aniqlikda yaxlitlang.
Berilgan son: 0.26598
Natija:
1-aniqlikda: 0.3
2-aniqlikda: 0.27
3-aniqlikda: 0.266
4-aniqlikda: 0.2660

```
n=float(input("n="))
for i in range(1,5):
    print(f'{i}-aniqlik:{round(n,i)}')
```

8. Doira sektorining yuzasini hisoblash dasturini tuzing. Doiraning radiusi hamda sektorning burchagi (gradusda) foydalanuvchi tomonidan kiritiladi.

```
from math import*
b=int(input('burchakni kiriting='))
r=int(input('radiusni kiriting='))
s=(pi*r**2*b)/360
print(Doira sektori yuzi,s)
```

9. Rangi yashil, o‘lchami 100×100 bo‘lgan “Mening birinchi ilovam” nomli GUI oynasini yaratuvchi dastur kodini yozing. Oynada “Salom O‘zbekiston” xabarini chiqaruvchi tugmachasini joylashtiring.

```
from tkinter import*
window = Tk()
window.title('Mening birinchi ilovam')
window.geometry('300x200')
window.configure(background='green')
matn=Label(window, width=25,height=5,bg='green',text='')
matn.grid(row=0, column=0)
def oyna():
    matn.config(text='Salom O`zbekiston!')
tugma=Button(window,text="Tanlang",width=10,command=oyna)
tugma.grid(row=1,column=0)
window.mainloop()
```

10. Rangi pushti, o‘lchami 250×150 bo‘lgan “Mevalar” nomli GUI oynasini yaratuvchi dastur kodini yozing. Oynada berilgan 4 ta mevadan birini tanlash imkonini beruvchi vijetni joylashtiring.

```
from tkinter import *
window = Tk()
window.title('Mevalar')
window.geometry('250x150')
window.configure(background='green')
options=('Olma','Anor','Nok','Gilos')
a=IntVar()
a.set('Tanlang:')
b=OptionMenu(window, a, *options)
b.grid()
window.mainloop()
```

Foydalanuvchi grafik interfeysi

1. GUIdan foydalanib, 1 dan n gacha bo‘lgan 10 ta tasodifiy sonni chiqaruvchi dastur tuzing. n foydalanuvchi tomonidan kiritiladi.

```
from tkinter import*
from random import randint
def tasodifiy():
    son=int(a.get())
    b.delete(0.0,END)
    for i in range (1,11):
        s=str(randint(1,son))+'\n'
        b.insert(END,s)
window=Tk()
window.title('Tasodifiy son')
window.geometry('250x250')
window.configure(background='lightgreen')
k=Label(window,text='Son: ', bg='lightgreen')
k.grid(row=0, column=0)
j=Label(window, text='\n Natija', bg='lightgreen')
j.grid(row=2, column=0)
a=Entry(window, width=5)
a.grid(row=1, column=0)
b=Text(window,height=10,width=6)
b.grid(row=3, column=0)
t=Button(window,text='Tasodifiy son', command=tasodifiy)
t.grid(row=1,column=1)
window.mainloop()
```

2. GUIdan foydalanib, berilgan songa karrali 10 ta sonni chiqaruvchi dastur tuzing. Berilgan son foydalanuvchi tomonidan kiritiladi.

```
from tkinter import*
def karra():
    n=int(a.get())
    b.delete(0.0,END)
    for i in range (1,11):
        son=str(i*n)+'\n'
        b.insert(END,son)
window=Tk()
window.title('Karrali sonlar')
window.geometry('250x250')
window.configure(background='orange')
c=Label(window,text='Son: ', bg='orange')
c.grid(row=0, column=0)
d=Label(window, text='\n Natija', bg='orange')
d.grid(row=2, column=0)
a=Entry(window, width=5)
a.grid(row=1, column=0)
b=Text(window,height=10,width=6)
b.grid(row=3, column=0)
t=Button(window,text='Karrali sonlar', command=karra)
t.grid(row=1,column=1)
window.mainloop()
```

3. GUIdan foydalanib, berilgan songa karrali sonlarni chiqaruvchi dastur tuzing. Berilgan son va karrali sonlarning soni foydalanuvchi tomonidan kiritiladi.

```
from tkinter import*
def karra():
    n=int(a.get())
    m=int(b.get())
    c.delete(0.0,END)
    for i in range (1,m+1):
        son=str(i*n)+'\n'
        c.insert(END,son)
window=Tk()
window.title('Karrali sonlar')
window.geometry('250x350')
window.configure(background='yellow')
```

```
k=Label(window,text='Son: ', bg='yellow')
k.grid(row=0, column=0)
j=Label(window, text='\n Natija', bg='yellow')
j.grid(row=2, column=4)
a=Entry(window, width=5)
a.grid(row=1, column=0)
karra_soni=Label(window,text='Karra_soni: ', bg='yellow')
karra_soni.grid(row=0, column=4)
b=Entry(window, width=5)
b.grid(row=1, column=4)
c=Text(window,height=15,width=6)
c.grid(row=3, column=4)
t=Button(window,text='Karrali sonlar', command=karra)
t.grid(row=1,column=5)
window.mainloop()
```

4. GUIdan foydalanib, ikkita a va b sonni qabul qiladigan, ab ni hisoblovchi dastur tuzing.

```
from tkinter import*
def son():
    n=int(a.get())
    m=int(b.get())
    c.delete(0.0,END)
    satr=str(n)+str(m)
    c.insert(END,satr)
window=Tk()
window.title('Sonlar')
window.geometry('250x350')
window.configure(background='yellow')
k=Label(window,text='1-son: ', bg='yellow')
k.grid(row=0, column=0)
j=Label(window, text='\n Natija', bg='yellow')
j.grid(row=2, column=4)
a=Entry(window, width=5)
a.grid(row=1, column=0)
karra_soni=Label(window,text='2-son: ', bg='yellow')
karra_soni.grid(row=0, column=4)
b=Entry(window, width=5)
b.grid(row=1, column=4)
c=Text(window,height=15,width=6)
c.grid(row=3, column=4)
t=Button(window,text='Bajarish', command=son)
t.grid(row=1,column=5)
window.mainloop()
```

```
import math
n=int(input("n="))
b=2
s=0
a=True
while a:
    for i in range(2, int(math.sqrt(b) + 1)):
        if b%i==0:
            break
    else:
        s+=1
        print(b)
    b+=1
    if n==s:
        a=False
```

```
from tkinter import*
import math
def tub():
    n=int(m.get())
    d.delete(0.0,END)
    b=2
    s=0
    a=True
    while a:
        for i in range(2, int(math.sqrt(b) + 1)):
            if b%i==0:
                break
        else:
            s+=1
            satr=str(b)+'\n'
            d.insert(END,satr)
        b+=1
        if n==s:
            a=False
window=Tk()
window.title('Tub sonlar')
window.geometry('250x250')
window.configure(background='pink')
k=Label(window,text='Son kiriting: ', bg='pink')
k.grid(row=0, column=0)
```

```
j=Label(window, text='\n Natija', bg='pink')
j.grid(row=2, column=0)
m=Entry(window, width=5)
m.grid(row=1, column=0)
d=Text(window,height=10,width=6)
d.grid(row=3, column=0)
t=Button(window,text='Tub sonlar', command=tub)
t.grid(row=1,column=1)
window.mainloop()
```

6. GUIdan foydalanib, ikkita a va b sonni qabul qiladigan, ular EKUBini hisoblovchi dastur tuzing.

```
def ekub(a, b):
    while a != 0 and b != 0:
        if a > b:
            a %= b
        else:
            b %= a
    return a + b
a = int(input('a = '))
b = int(input('b = '))
print('EKUK:', ekub(a, b))
```

```
from tkinter import*
def ekub():
    a = int(n.get())
    b = int(m.get())
    i.delete(0.0,END)
    while a != 0 and b != 0:
        if a > b:
            a %= b
        else:
            b %= a
    k= a + b
    i.insert(END,str(k))
window=Tk()
window.title('EKUB ni hisoblash')
window.geometry('250x250')
window.configure(background='lightblue')
k=Label(window,text='Son_1: ', bg='lightblue')
k.grid(row=0, column=0)
```

```
j=Label(window, text='\n Natija', bg='lightblue')
j.grid(row=2, column=4)
n=Entry(window, width=5)
n.grid(row=1, column=0)
l=Label(window,text='Son_2: ', bg='lightblue')
l.grid(row=0, column=4)
m=Entry(window, width=5)
m.grid(row=1, column=4)
i=Text(window,height=15,width=6)
i.grid(row=3, column=4)
t=Button(window,text='EKUB ni hisoblash', command=ekub)
t.grid(row=1,column=5)
window.mainloop()
```

6. GUIdan foydalanib, ikkita a va b sonni qabul qiladigan, ularni EKUKini hisoblovchi dastur tuzing.

```
def ekuk(a, b):
    m = a * b
    while a != 0 and b != 0:
        if a > b:
            a %= b
        else:
            b %= a
    return m // (a + b)
a = int(input('a = '))
b = int(input('b = '))
print('EKUK:', ekuk(a, b))
```

```
from tkinter import*
def ekuk():
    a = int(n.get())
    b = int(m.get())
    i.delete(0.0,END)
    p = a * b
    while a != 0 and b != 0:
        if a > b:
            a %= b
        else:
            b %= a
    k=p // (a + b)
```

```
    i.insert(END,str(k))
window=Tk()
window.title('EKUK ni hisoblash')
window.geometry('250x250')
window.configure(background='lightblue')
k=Label(window,text='Son_1: ', bg='lightblue')
k.grid(row=0, column=0)
j=Label(window, text='\n Natija', bg='lightblue')
j.grid(row=2, column=4)
n=Entry(window, width=5)
n.grid(row=1, column=0)
l=Label(window,text='Son_2: ', bg='lightblue')
l.grid(row=0, column=4)
m=Entry(window, width=5)
m.grid(row=1, column=4)
i=Text(window,height=15,width=6)
i.grid(row=3, column=4)
t=Button(window,text='EKUK ni hisoblash', command=ekuk)
t.grid(row=1,column=5)
window.mainloop()
```

```
def ekuk(a, b):
    n = a * b
    while a != 0 and b != 0:
        if a > b:
            a %= b
        else:
            b %= a
    return n // (a + b)
def ekuk3(a,b,c):
    return ekuk(ekuk(a,b),c)
a = int(input('a = '))
b = int(input('b = '))
c = int(input('c = '))
print('EKUK:', ekuk3(a, b, c))
```

```
def ekuk(a, b):
    n = a * b
    while a != 0 and b != 0:
        if a > b:
            a %= b
        else:
            b %= a
    return n // (a + b)
def ekuk4(a,b,c,d):
    return ekuk(ekuk(ekuk(a,b),c),d)
a = int(input('a = '))
b = int(input('b = '))
c = int(input('c = '))
d = int(input('d = '))
print('EKUK:', ekuk4(a, b, c,d))
```

7. 1 dan n gacha bo‘lgan natural sonlar kubining yig‘indisini topish dasturini tuzing.

```
n=int(input('n sonini kiriting='))
s=0
for i in  range(1,n+1):
    s+=i**3
print(s)
```

8. P=2*4*6*….*40 ko‘paytmani hisoblash dasturini tuzing.

```
#P=2*4*6*...*40
p=1
for i in  range(2,41,2):
    p*=i
print(p)
```

9. 0 bilan tugovchi sonlar ketma-ketligi berilgan. Uni akslantiruvchi dastur tuzing. Masalan, 1230 dan 0321 hosil qiling.

```
def akslantirish(n):
    s=''
    if n%10==0:
        while n>0:
            m=n%10
            s+=str(m)
            n=n//10
    return s
n=int(input('n='))
print(akslantirish(n))
```

10. Protsedura yordamida berilgan matndagi ‘k’ belgini ‘q’ belgiga, ‘t’ belgini ‘d’ belgiga, ‘n’ belgini ‘m’ belgiga almashtiruvchi dastur tuzing.

```
def almashtirish():
    b = ''
    for i in satr:
            if i == 'k':
                i = 'q'
            elif i=='t':
                i='d'
            elif i=='n':
                i='m'
            b += i
    print(b)
satr=input('satr kiriting: ')
almashtirish()
```

5. Berilgan ikki yoki undan ortiq sonning eng kichik umumiy karralisi (EKUK)ni topish dasturini tuzing. Funksiyadan foydalaning.

Kiruvchi ma’lumot	Chiquvchi ma’lumot
18 24	72

```
def ekuk(a, b):
    n = a * b
    while a != 0 and b != 0:
        if a > b:
            a %= b
        else:
            b %= a
    return n // (a + b)
a = int(input('a = '))
b = int(input('b = '))
print('EKUK:', ekuk(a, b))
```

6. Soat, minut va sekundda berilgan birlikni sekundga aylantirish funksiyasini yozing.

Kiruvchi ma'lumot	Chiquvchi ma'lumot
soat=4 minut=15 sekund=60	15360 s

```
def sekundlar(soat,minut,sekund):
    return soat*3600+minut*60+sekund
soat=int(input('soat='))
minut=int(input('minut='))
sekund=int(input('sekund='))
print(sekundlar(soat,minut,sekund),'s')
```

7. Canvas maydonida matn va rasmdan iborat otkritka hosil qiling.

```
from tkinter import*
window=Tk()
window.title('Rasmlarni joylashtirish')
canvas = Canvas(window, height=350, width=700, bg='lightgreen')
a = PhotoImage(file='d:\\2.png')
image = canvas.create_image(350,200,image=a)
canvas.create_text(300,100,text='Navro`z bayrami muborak bo`lsin')
canvas.grid(row=0,column=0)
window.mainloop()
```

8. Shilliqqurt balandligi h metr daraxtning yuqorisiga sudralib chiqmoqda. U kunduzi a metr ko‘tarilsa, kechasi b metr pastga tushadi. U daraxt uchiga necha kunda yetib boradi? h, a va b (a>b) qiymatlar foydalanuvchi tomonidan kiritiladi.

```
h=int(input('balandlikni kiriting='))
a=int(input('ko`tarilish metr='))
b=int(input('tushish metri='))
n=((h-a)/(a-b))+1
print(f'{n} kunda ko`tariladi')
h=10, a=4, b=3
 0 1 2 3 4 5 6 7 8 9 10
   1 1 1 1 1 1          1
```

9. Ikkita a va b son berilgan. Ulardan kattasini aniqlash dasturini tuzing. (Funksiyasidan foydalanib)

```
def ikki(a,b):
    if a>b:
        return a
    else:
        return b
a=int(input('a='))
b=int(input('b='))
print(ikki(a,b))
```

10. Belgilar yordamida hosil qilingan pingvinni n marta chiqarish dasturini tuzing. n –1 dan 4 gacha bo‘lgan natural sonlarni qabul qiladi.

```
n=int(input('n='))
for i in range(0,n):
    print(" "*7+"~")
    print(" "*6+"- "*2)
    print(" "*4+("","O","O","")")
    print(" "*4+"/ ","V"," \ ")
    print(" "*3+"/"+"( ","-"," )"+"\ ")
    print(" "*5+"^"*2,"^"*2)
```

11. n ta kvadratni ekranga chiqaruvchi dastur tuzing. n –1 dan 5 gacha bo‘lgan natural sonlarni qabul qiladi.

```
n=int(input('n='))
for i in range(0,n):
    print("+"*5,"\n"+"+"*5,"\n"+"+"*5,"\n"+"+"*5,"\n")
```

QUYIDA KELTIRILGAN DASTURLAR NATIJASINI ANIQLANG.

Python. Dastur natijasini aniqlang.

```
for i in range(5):
    if i%2==0:
        continue
    print(i)
```

Python. Dastur natijasini aniqlang.

```
n=12
i=1
s=0
while i<=n:
    if (i%2==0 and i-5>4):
        print("Natija: ",i)
        break
    i+=1
```

Ushbu berilgan dasturda n=123, m=90 bo‘lganda ekranga chiqadigan natijani aniqlang.

```
n=int(input("n="))
m=int(input("m="))
k=0
while n>m:
    n-=m
    k+=1
print("Natija:",n*k)
```

Ushbu berilgan dasturda n=143, m=23 bo‘lganda ekranga chiqadigan natijani aniqlang.

```
n=int(input("n="))
m=int(input("m="))
k=0
while n>m:
    n-=m
    k+=1
print("Natija: {0} va {1}".format(n,k))
```

Dastur natijasini aniqlang.

```
a=45
b=27
c=a*b
while a!=b:
    if a>b:
        a=a-b
    else:
        b=b-a
print(c/a)
```

Dastur natijasini 16 chiqishi uchun n ga qanday qiymat berish lozim.

```
n=int(input("n="))
s=0
k=1
while s<n:
    s=s+k
    k=k+2
print(s)
```

Dastur natijasi necha marta ekranga chiqariladi.

```
i=5
b=4
x=i*b
while True:
    print(i)
    i=i-1
    if i<=2:
        break
    else:
        continue
print(x/i)
```

Python dasturida berilgan kod natijasini aniqlang.

```
def f(n):
    if n<=3:
        return 2
    return 3*f(n-2)-f(n-3)
print(f(9))
```

Python dasturida berilgan kod natijasini aniqlang.

```
d=lambda p:p*2
t=lambda a:a*3
x=2
x=d(x)
x=t(x)
x=d(x)
print(x)
```

Python dasturida berilgan kod natijasini aniqlang.

```
def F(n):
    if n==1:
        return 0
    elif n==2:
        return 1
    else:
       return F(n-1)+F(n-2)
print(F(9))
```

Python. Dastur natijasini toping:

```
def F(n):
    if n==1:
        return 1
    elif n==2:
        return 2
    else:
        return F(n-1)+F(n-2)
print(F(6))
```

Python. Dastur natijasini toping:

```
def r(q):
    return q*2
def s(q):
    return q*3
x=12
x=r(x)
```

```
x=s(x)
x=r(x)
print(x)
```

BITWISE OPERATORLAR

& - operator natijasi har bir bit pozitsiyasining mantiqiy ko‘paytmasidan iborat bo‘ladi. (Rost – 1, Yolg’on – 0)

Dastur natijasini aniqlang.

```
x=True
y=False
z=True
if not x or y:
   print(12 & 3)
elif not x or y and z:
   print(18 & 14)
elif x and y or z:
   print(23 & 14 )
else:
   print(5 & 2)
```

| - operator natijasi har bir bit pozitsiyasining mantiqiy yig’indisidan iborat bo‘ladi.

Dastur natijasini aniqlang.

```
z = 17
if z % 2 == 0 :
 print(15 |  2)
elif z % 3 == 0 :
 print(27 | 19)
else :
 print(33 |  26)
```

^ - xor (maxsus qo‘shish) operatori.

Dastur natijasini aniqlang.

```
x = 6
y = -11
if x > 0 and y > 0:
   print(18 ^ 17)
elif x > 0 and y == 0:
   print(29 ^ 24)
```

```
elif  not(x < 0) and y < 0:
    print(41 ^ 17)
else:
    print(89 ^ 12)
```

~ - operator natijasi har bir bit pozitsiyasining mantiqiy inkorini hisoblaydi.

Dastur natijasini aniqlang.

```
x=1
a=4
b=5
if a>0:
    if b<0:
        x=x+5
    elif a>4:
        x=x+4
    else:
        x=x+3
else:
        x=x+2
print(~x)
```

<< - ikkilik chapga surish operatori.

Dastur natijasini aniqlang.

```
a=11
b=24
if a%2==0 and b%2==0:
    print(a<<1, b<<1)
elif a%2==1 and b%2==0:
    print(a<<1, b<<2)
elif a%2==0 and b%2==1:
    print(a<<2, b<<1)
else:
    print(a<<2, b<<2)
```

>> - ikkilik o‘ngga surish operatori.

Dastur natijasini aniqlang.

```
a=51
```

```
b=a//10
c=a%10
if b > a or b <= c and a < c:
   print(a >> 1)
else:
   print(a >> 2)
```

Dastur natijasini aniqlang.

```
a=85
b=16
if a>0 and b<0 and (a+b)>100:
   print(a or b)
elif a>0 and b>0 and (a+b)>100:
   print(a and b)
else:
   print(100 and 50)
```

Dastur natijasini aniqlang.

```
a=7
if bool(a%2==0):
   print(bool(71 or 56))
elif bool(a%3==0):
   print(bool(33 and 28 or 1))
else:
   print(bool(42 and 35 or 14))
```

is – berilgan ikkita ifoda bir-biriga teng bo‘lsa rost, aks holda yolg’on qiymat qaytaradi.

Dastur natijasini aniqlang.

```
a=7
b=6
c=7
if (a and b) is c:
   print(1)
elif a is not c:
   print(2)
elif (a or c) is (b and c):
   print(3)
```

```
else:
   print(4)
```

Python. Dastur natijasini aniqlang.

```
x,y =0,1
print(x|y==y|x)
print(x-y==y-x)
print(x&y==y&x)
```

SAVOL VA TOPSHIRIQLAR

1. Foydalanuvchining grafik interfeysi nima?
2. Grafik interfeysli ilovalar qanday yaratiladi?
3. GUIdan foydalanib, 1 dan n gacha bo‘lgan 10 ta tasodifiy sonni chiqaruvchi dastur tuzing. n foydalanuvchi tomonidan kiritiladi.
4. GUIdan foydalanib, ikkita a va b sonni qabul qiladigan, ular EKUBini hisoblovchi dastur tuzing.
5. Berilgan ikki yoki undan ortiq sonning eng kichik umumiy karralisi (EKUK)ni topish dasturini tuzing. Funksiyadan foydalaning.
6. Canvas maydonida matn va rasmdan iborat otkritka hosil qiling.
7. 0 bilan tugovchi sonlar ketma-ketligi berilgan. Uni akslantiruvchi dastur tuzing. Masalan, 1230 dan 0321 hosil qiling.

XULOSA

Ushbu darslik 60110600 – “Matematika va informatika” bakalavr ta’lim yo‘nalishi talabalarida zamonaviy dasturlash tillarini puxta egallashda muhim o‘rin egallaydi. Yana shuni alohida ta’kidlash lozimki, dasturlash tillarini o‘qitishda ilg‘or xorijiy tajribalardan foydalanish, dasturlashda raqamli texnologiyalarning o‘rni va robot texnologiyalar haqida tushunchalarni yetkazish talabalarni dasturlash tillariga bo‘lgan qiziqishini ortishiga asos bo‘ladi.

Darslarning xilma-xil shakli talabalarning bilim olish qobiliyatini oshiradi. O‘yin ko‘rinishidagi darslar, amaliy topshiriqlar, turli darajadagi topshiriqlardan foydalanish, farqlangan vazifalar, raqobatbardosh vazifalarni tashkil etish fanga qiziqishni uyg‘otadi. O‘z-o‘zini nazorat qilish va guruhli ishlash uchun vazifalar o‘zlashtirishi past va iqtidorli talabalar bilan ishlashni tashkil etish muammosini hal qiladi. Dasturlash tillariga mo‘ljallangan fanlardan olingan bilimlarni darsdan tashqari mashg‘ulotlarda qo‘llash talabalarning ushbu sohadagi bilimlarini chuqurlashtirish, ijodkorlik, zukkolikni namoyon etish va qobiliyatlarni rivojlantirishga imkon beradi.

Oliy o‘quv yurtlarida dasturlash tillarini o‘qitishda ta’limning interfaol metodlari va ta’limning zamonaviy texnologiyalaridan foydalanish talabalar faolligini oshiradi, mustaqil izlanishga undaydi va ta’lim samaradorligini yaxshilashga xizmat qiladi.

ATAMALARNING IZOHLI LUG`ATI

Prototip – butun tizimning ishlashini tahlil qilish uchun asosiy funksional imkoniyatlarni tezda bajarish.

Propedevtik – tizimli ravishda qisqa va elementar shaklda bayon etilgan fanga kirish.

Freelance – shtatda bo‘lmagan, rasmiy ravishda xodim sanalmaydigan ishchi bo‘lib, internetda masofadan turib ishlashni nazarda tutishadi.

Scratch – bu bolalar uchun oddiy, tushunarli va nihoyatda qiziqarli dasturlash tilidir. Unda yodda tutish va xatosiz yozish lozim bo‘lgan kodlar uchramaydi. Bunda faqat o‘qish va hisoblashni bilish yetarlidir. «Lego» konstruktori bilan bo‘lgani kabi, Scratch yordamida turli rangdagi «g‘ishtchalar» — bloklardan dasturlarni yig‘ish mumkin.

IOS (iPhone OS) - bu Apple kompaniyasi tomonidan ishlab chiqilgan mobil operatsion tizim.

Dastur – oldindan tayyorlangan algoritm va hisoblash texnikasi vositalariga asoslangan holda bajariladigan ishning shart-sharoiti, maqsadi va vazifasini e’tiborga olib, uning biror bir formal (shartli) algoritmik tildagi aniq va to‘liq ifodalanishi.

Dastur – bajariladigan ish yoki biror faoliyat rejasi. Dastur – biror masalani yechishda kompyuterda bajarishi lozim bo‘lgan amallarning izchil tartibi.

Dasturlash tili – kompyuterlar uchun dasturlar (ko‘rsatmalar yig‘indisi) yoziladigan, uni u yoki bu harakatlarni bajarishga majbur qiladigan rasmiy til.

Dasturlash bu – kompyuterlar va boshqa mikroprotsessorli elektron mashinalar uchun mo‘ljallangan dasturlarni yaratish, tajriba orqali sinovdan o‘tkazish hamda xatolarni tuzatib borish jarayonidan iboratdir. Boshqacha aytganda, kompyuter uchun dastur tuzish jarayoni dasturlash va dastur tuzadigan kishi dasturchi deyiladi. Kompyuter tushunadigan til dasturlash tili deb ataladi.

Dasturlash muhiti bu – dasturchi yozadigan kodlarni aynan qanday tilda va muhitda yozishi tushuniladi. Masalan: keng tarqalgan va ko‘plab foydalanuvchiga ega muhitlarni misol qilish mumkin. PHPStorm — asosan PHP dasturchilarga, VisualStudio — .Net dasturchilarga, NetBeans —Java, PHP dasturchilarga, PHPDesigner —

asosan veb (PHP) dasturchilarga mo‘ljallangan. Hozirgi kunda dastur tuzish yuqori darajadagi dasturlash tillari (Delphi, Java, C++, S#, Python) vositasida amalga oshiriladi. Bu dasturlash tillarining semantikasi inson tiliga yaqinligi bois dastur tuzish jarayonini osonlashtiradi.

Ma’lumotlar bazasi - bu turli xil alohida yozuvlar yoki yozuvlar guruhlarini olish mumkin bo'lgan tarzda tashkil qilingan, o'zaro bog'liq ma'lumotlar to'plamidir.

S.rfind(str,[start],[end]) - Satrdan satr ostini axtarish. Oxirgi kirish raqamini yoki 1 ni qaytaradi

S.index(str,[start],[end]) - Satrdan satr ostini axtarish. Birinchi kirish raqamini qaytaradi yoki ValueError istisnosini chaqiradi

S.rindex(str,[start],[end]) - Satrdan satr ostini axtarish. Oxirgi kirish raqamini qaytaradi yoki ValueError istisnosini chaqiradi.

S.isdigit() - Satrda raqamlar ishtirok etganligini tekshirish.

S.isalpha() - Satr faqat harflardan iboratligini tekshirish.

S.isalnum() - Satr harf yoki raqamlardan iboratligini tekshiradi.

S.islower() - Satr quyi registrdagi belgilardan iboratligini tekshiradi.

S.isspace() -Satrda ko‘rinmaydidan belgilar borligini tekshirish (probel, sahifani o‘tkazish belgisi(‘\p’), yangi satrga o‘tish(‘\n’), koretkani qaytarish(‘\r’), gorizontal tabulyatsiya(‘\t’) va vertikal tabulyatsiya)

S.istitle() - Satrda so‘zlar bosh harf bilan boshlanishini tekshirish.

S.starswith(str) - S satr str shablonidan boshlanishini tekshirish.

S.endswith(str) - S satr str shabloni bilan tugashini tekshirish.

S.join(ro‘yxat) - S ajratuvchiga ega ro‘yxatdan qatorni yig`ish.

Ord(belgi) - Belgiga mos ASCII kodni qaytaradi.

Chr(son) - ASCII kodga mos belgini qaytaradi.

S.capitalize() - Satrning birinchi belgisi yuqori registrda qolganlarini quyi registrga o‘tkazadi.

S.center(width,[fill]) - Chegaralari bo‘yicha fill (jimlik holatida probel) belgisi turuvchi markazlashtirilgan satrni qaytaradi.

S.expandtabs(tabsize) - Joriy ustungacha bir yoki bir qancha probellar bilan tabulyatsiyaning hamma belgilari almashtirilgan satr nusxasini qaytaradi. Agarda TabSize ko‘rsatilmagan bo‘lsa tabulyatsiya hajmi 8 probelga teng bo‘ladi.

S.lstrip([chars]) - Satr boshidagi probel belgilarini olib tashlash.

S.rstrip([chars]) - Satr oxiridan probel belgilarini olib tashlash.

S.strip([chars]) - Satr boshidan va oxiridan probel belgilarini olib tashlash.

S.partition(shablon) - Birinchi shablon oldida turuvchi qismni keyin shablonni o‘zini va shablondan keyin turuvchi qismga ega kortejni qaytaradi. Agarda shablon topilmasa satrga ega bo‘lgan kortejni qaytaradi, avval ikki bo‘sh satr keyin satrni o‘zini.

S.rpartition(sep) - Oxirgi shablon oldida turuvchi qismni keyin shablonni o‘zini va shablondan keyin turuvchi qismni qaytaradi. Kortej qator o‘zidan va undan keyin ikkita bo‘sh qatordan iborat bo‘ladi.

S.swapcase() - Quyi registrdagi belgilarni yuqori registrga, yuqorilarni esa quyiga o‘tkazadi.

S.title() - Har bitta so‘zning birinchi harfini yuqori registrga qolganlarini esa quyi registrga o‘tkazadi.

S.zfill(width) - Qator uzunligini Widthdan kam qilmaydi agar kerak bo‘lsa birinchi belgilarni nollar bilan to‘ldiradi.

List – tartiblangan va o‘zgaruvchan ro‘yxat. Elementlarini dublikatlash mumkin.

Tuple – tartiblangan va o‘zgarmas ro‘yxat. Elementlarini dublikatlash mumkin.

Set – Tartiblanmagan va indekslanmagan to‘plam. Elementlari dublikatlanmaydi.

Dictionary – tartiblanmagan,o‘zgaruvchan va indekslangan to‘plam. Elementlari

dublikatlanmaydi.

List.append(x) - Ro‘yxat oxiridan element qo‘shish

List.extend(L) - Oxiriga hamma elementlarni qo‘shib list ro‘yxatini kengaytiradi.

List.clear() - Ro‘yxatni tozalaydi.

TEST SAVOLLARI.

Dastur natijasini aniqlang. int(round(4.56666)+7//4)=?

a) 6
b) 5
c) 3
d) 2

Dastur natijasini aniqlang: x = 90 while (x < 100): x+=3 print(x)

a) 102
b) 103
c) 99
d) 100

Dastur natijasini aniqlang:
x = 1 a = 4 b = 5 if a > 0: if b < 0: x = x + 5 elif a > 4: x = x +4 else:x=x+3 else: x = x + 2 print(x)

a) 4
b) 5
c) -3
d) 6

Dastur natijasini aniqlang: x=0 a=4 b=5 if a>0: if b<0: x=x+5 elif a>5:x=x+4 else:x=x+3 else:x=x+2 print(x)

a) 3
b) 4
c) 5
d) 6

Quyidagi kod natijasini ko‘rsating: x = 45 // 4 % (3 + 2) ** 4 + 25//6 print(x)

a) 15
b) 25
c) 16
d) 12

Quyidagi kod natijasini ko‘rsating: print(2 or 3)

a) 2
b) 3

c) 1
d) 0

Quyidagi kod natijasini ko‘rsating: print(100 and 50)
a) 50
b) 100
c) 150
d) 0

Quyidagi kod natijasini ko‘rsating: print(-500 or 500)
a) -500
b) 500
c) 150
d) 0

Dastur natijani aniqlang. n=5 m=4 a=[[0]*m]*n a[0][0]=4 print(a[1][0])
a) 4
b) 5
c) 20
d) 0

Dastur natijani toping. x=20 num=20 if x>0: print(num+10) else: print(num)
a) 30
b) 20
c) 0
d) -15

Quyidagi kod natijasini ko‘rsating: print(bool(-10))
a) True
b) False
c) 1
d) 0

Quyidagi kod natijasini ko‘rsating: print(bool(‘abc’))
a) True
b) False

c) 0
d) 1

Quyidagi kod natijasini ko‘rsating: print(bool(0))
a) False
b) True
c) 0
d) 1

Quyidagi kod natijasini ko‘rsating: print(bool(1111))
a) True
b) False
c) 0
d) 1

Dastur natijasini aniqlang. a=[1,2,3,4,5,6] a[3:5]=[7,8,9] print(a)
a) [1, 2, 3, 7, 8, 9, 6]
b) [1, 2, 3]
c) [3, 7, 8, 9, 6]
d) [1, 2, 3, 7, 8, 9, 6,3]

Dastur natijasini aniqlang. x=True y=False z=False if not x or y: print(1) elif not x or not y and z: print(2)
elif x and y or not z: print(3) else: print(4)
a) 3
b) 4
c) 1
d) 2

Dastur natijasini aniqlang. i=5 if i%2==0: print(bool(12)) else: print(bool(0))
a) False
b) True
c) 0
d) 1

Dastur natijasini aniqlang. x=85 while(x<100): x+=4 print(x)

a) 101
b) 85
c) 100
d) 104

Dastur natijasini toping. d=lambda p:p*3 t=lambda p:p*5 x=2
x=d(x) x=t(x) x=d(x) print(x)

a) 90
b) 100
c) 12
d) 82

Dastur natijasini aniqlang. x=True y=False z=False if not x or y:
print(1) elif not x or not y and z: print(2)
elif x and y or not z: print(3) else: print(4)

a) 3
b) 4
c) 2
d) 1

Dastur natijasini aniqlang. i=5 if i%2==0: print(bool(12)) else:
print(bool(0))

a) False
b) True
c) 1
d) 0

Dastur natijasini aniqlang. x=80 while (x<100): x+=4 print(x)

a) 100
b) 101
c) 85
d) 80

Dastur natijasini aniqlang. a=[1,2,3,4,5,6,7] a[3:5]=[7,8,9] print(a)

a) [1, 2, 3, 7, 8, 9, 6, 7]
b) [2, 3, 7, 8, 9, 6, 7]
c) [3, 7, 8, 9, 6, 7]

d) [1, 2, 3, 7, 8, 9, 6, 7,3,5]

Dastur natijasini aniqlang. n=3 m=4 a=[[0]*n]*m a[0][0]=5 print(a[1][0])

a) 5
b) 3
c) 4
d) 2

Dastur natijasini aniqlang. d=lambda a:a*4 t=lambda a:a*4 x=3 x=d(x) x=t(x) x=d(x) print(x)

a) 192
b) 202
c) 156
d) 104

Dastur natijasini aniqlang: x = 50 while (x < 80): x+=2 print(x)

a) 80
b) 50
c) 75
d) 96

Dastur natijasini aniqlang: x = 10 while (x < 100):   x*=2   print(x)

a) 160
b) 320
c) 80
d) 40

Takrorlanishlar sonini aniqlang: x = 10 while (x < 100):   x*=2 print(x)

a) 4
b) 10
c) 8
d) 16

Takrorlanishlar sonini aniqlang: x=5 s=0 while (x <200): x*=2 s+=1 print(s)

a) 6

b) 12
c) 8
d) 16

Takrorlanishlar sonini aniqlang: for i in range(-5,-10,-2): print(s)
a) 3
b) 6
c) 8
d) 10

Takrorlanishlar sonini aniqlang: for i in range(1,10,-3): print(s)
a) 0
b) 6
c) 8
d) 10

Takrorlanishlar sonini aniqlang: for i in range(-5,15,3): print(s)
a) 7
b) 6
c) 9
d) 10

Takrorlanishlar sonini aniqlang: for i in range(-5,15,-2): print(s)
a) 0
b) 6
c) 9
d) 10

Takrorlanishlar sonini aniqlang: for i in range(1,15,2): for i in range(2,12,3): print(s)
a) 28
b) 14
c) 18
d) 19

Dastur natijasini aniqlang. x=15//8%(16-14)**5+17//7 print(x)
a) 3
b) 5

c) 6
d) 9

Dastur natijasini aniqlang. x=34%len('dars')**2-(7*3)//2**2*7
print(x)
a) -33
b) -60
c) 61
d) 38

Dastur natijasini aniqlang. a='O‘zbekiston' print(a[3:6])
a) Bek
b) O‘zbek
c) O‘z
d) ton

Dastur natijasini aniqlang. a=‘Biologiya' b=‘Kimyo' c=‘Adabiyot'
print(a[:4]+b[1:3]+c[1:6:3])
a) Biolimdi
b) Kimyola
c) Adabio
d) Biolidi

Dastur natijasini aniqlang. a='Maktab' b='bilim bulog`i' c='bilim' in b
print(len(a)==5 and c==True or len(b)>12)
a) True
b) False
c) Maktab
d) 1

Dastur natijasini aniqlang. a='Attestatsiya' b='Informatika'
print(('test' in a) and (b[2:9:3]=='fmi')==False)
a) False
b) True
c) 0
d) 1

Dastur natijasini aniqlang. a='15+1835' b='183'
print(len(a[5:])==len(b[:2]) and not(True==(b in a)))

a) False
b) True
c) 1
d) 0

Dastur natijasini aniqlang. x=1 a=-4 b=10 if a>0: if b<0: x=x+5 elif a>4: x=x+4 else: x=x+3 else: x=x+2 print(x)

a) 3
b) 6
c) 8
d) 7

Takrorlanishlar sonini aniqlang: for i in range(-21,21,5): print('4782')

a) 9
b) 8
c) 14
d) 6

Dastur natijasini aniqlang. x=0 for i in range(10): for j in range(-1,-10,-1): x+=1 print(x)

a) 90
b) 100
c) 85
d) 150

Dastur natijasini aniqlang. var = 10 for i in range(10): for j in range(2, 10, 1): if var % 2 == 0: continue var += 1 var+=1 else: var+=1 print(var)

a) 21
b) 20
c) 0
d) 12

Dastur natijasini aniqlang. s=0 for i in range(5,12,2): for i in range(1,-15,-3): s+=1 print(s)

a) 24

b) 26
c) 14
d) 10

Dastur natijasini toping. a=9 b=-6 s=0 if a<0: for i in range(1,10,-2): s+=i*i else: for i in range(9,3,-2): s+=i*i print(s)

a) 155
b) 160
c) 190
d) 147

Dastur natijasini toping. for s in 'dars': if s=='r': break print(s)

a) da
b) ars
c) sa
d) ar

Dastur natijasini toping. for s in 'dars': if s=='r': continue print(s)

a) das
b) ars
c) sad
d) ar

Dastur natijasini aniqlang. a=85 b=16 if a>0 and b<0 and (a+b)>100: print(a or b) elif a>0 and b>0 and (a+b)>100: print(a and b) else: print(100 and 50)

a) 16
b) 85
c) 100
d) 50

Dastur natijasini aniqlang. a=15 b=3 print(a==15 or b==32)

a) True
b) False
c) 1
d) 0

Dastur natijasini aniqlang. a=5 b=4 c=2 a-=b a*=c a+=(b*c+b) print(a)

a) 14
b) 15
c) 20
d) 24

Dastur natijasini aniqlang. a=8 b=-5 c=-1 a+=b a-=c a-=(b+c*b) print(a)

a) 4
b) 14
c) 15
d) 6

Dastur natijasini aniqlang. a=-7 b=4 c=a%b a+=b//c print(a)

a) -3
b) -5
c) 6
d) 2

Dastur natijasini aniqlang. a=15 b=2 c=a//b**3 a+=(b+a)**c b//=a print(b)

a) 0
b) 1
c) -3
d) 5

Dastur natijasini aniqlang. a=7 b=-4 c=a//b b-=(b%c) print(b)

a) -4
b) 5
c) -3
d) -2

Dastur natijasini aniqlang. a=32%-12//-2%8//-2%2 print(a)

a) 1
b) -2
c) -1
d) 0

Dastur natijasini aniqlang. a=-15 b=11 print(a%b)

a) 7
b) -7
c) 4
d) -4

Dastur natijasini aniqlang. a=15 b=-11 print(a//b)

a) -2
b) -1
c) 3
d) -4

Dastur natijasini aniqlang. a=7 b=-4 a*=(b+a) b+=(a*b) b+=(b-a) print(b)

a) -197
b) -180
c) 165
d) 123

Dastur natijasini aniqlang. satr = 'instagram' print (satr[1:3])

a) ns
b) ins
c) tg
d) ram

Dastur natijasini aniqlang. print(type(6/3))

a) <class 'float'>
b) <class 'int'>
c) <class 'str'>
d) <class 'bool'>

Dastur natijasini aniqlang. var= "Kutubxona" print(var[-3:-1])

a) on
b) xon
c) tub
d) Kut

Dastur natijasini aniqlang. var= "odobli bola" print(var[1::2])

a) dbibl
b) odobl
c) li bola
d) bola

Dastur natijasini aniqlang. a= "Informatika" print (a[-4::2])

a) tk
b) for
c) ka
d) tika

Dastur natijasini aniqlang. var= "O‘zbekiston" print(var[7::-3])

a) se`
b) tez
c) ton
d) O‘z

Dastur natijasini aniqlang. a='Qudratli' print (a[-4::-2])

a) adQ
b) uri
c) dti
d) ldQ

Dastur natijasini aniqlang. a= "Matematika" print (a[::-3])

a) ateM
b) Mate
c) Matik
d) aaea

Dastur natijasini aniqlang. var= "Kutubxona" print(var[5::])

a) xona
b) tubno
c) bxon
d) utubx

Dastur natijasini aniqlang. var= "Samsung" print(var[6::-2])

a) gumS

b) Sams
c) sung
d) msung

a=['bir','ikki','uch','to‘rt','besh','olti'] natija ['besh', 'uch', 'bir'] chiqadigan javobni belgilang.

a) print(a[-2::-2])
b) print(a[2::-3])
c) print(a[-4::-2])
d) print(a[0::-2])

Python. Natija False bo‘lgan dastur kodini aniqlang.

a) x = [1, 2, 3] print(x in x)
b) print(False == False in [False])
c) print((-9.0).is_integer())
d) print(3 in [1, 2, 3, 4, 5])

Python. Natijasi False bo‘lgan dastur kodini aniqlang.

a) print(all([1, (None), 2, None]))
b) print((-9.0).is_integer())
c) print((True==True) and (False in [False]))
d) print(False == False in [False])

ADABIYOTLAR RO‘YXATI

ASOSIY ADABIYOTLAR

1. A. R. Azamatov, B. Boltayev. Algoritmlash va dasturlash asoslari. O‘quv qo‘llanma. T. : “Cho‘lpon”, 2013 y.
2. Sh. Madraximov va boshq. C++ tilida programmalash bo‘yicha masalalar to‘plami. Toshkent, “Universitet” nashriyoti, -2014. -160 bet.
3. M.M. Aripov, N.A. Otaxanov, Dasturlash asoslari, O‘quv qo‘llanma. -T.: “Tafakkur bo‘stoni”, 2015. -240 bet.
4. F. M. Fayziyeva, D. M. Sayfurov, R. K. Atamuratov, L. K. Bagbekova, M. M Tilovova "Informatika va axborot texnologiyalari": umumiy o‘rta ta'lim maktablarining 10-sinfi uchun darslik: — Toshkent: Respublika ta'lim markazi, 2021. — 160 b.
5. Sh.A. Nazarov, C.G. Ivanova, S.M. Gaynazarov, Dasturlash texnologiyalari. Darslik. T.: “O‘zbekiston faylasuflari milliy jamiyati” nashriyoti, 2014. -280 bet.
6. H. Thomas Cormen. Intrudiction to algorithms. Third Edition. Massachusetts Instutite of Technology. The MIT Press. London 2009, 1292 p.
7. M. R. Fayziyeva, D. M. Sayfurov, N. S. Xaytullayeva “Informatika va axborot texnologiyalari”, umumiy o‘rta ta’lim maktablarining 9-sinfi uchun darslik: - Toshkent: Tasvir, 2020. – 112 b.
8. Fabio Nelli. Python data analytics. Rome, Library of Congress. 2018. 576 p.

QO‘SHIMCHA ADABIYOTLAR

1. Mirziyoyev SH.M. “Erkin va farovon, demokratik O‘zbekiston davlatini birgalikda barpo etamiz”. O‘zbekiston Respublikasi Prezidenti lavozimiga kirishish tantanali marosimiga bag`ishlangan Oliy Majlis palatalarining qo‘shma majlisidagi nutq, Toshkent, 2016. 56-b.
2. Mirziyoyev SH.M. “Qonun ustivorligi va inson manfaatlarini ta`minlash-yurt taraqqiyoti va xalq farovonligining garovi”. O‘zbekiston Respublikasi Konstitutsiyasi qabul qilinganining 24 yilligiga bag`ishlangan tantanali marosimidagi ma`ruza. 2016 yil 7 dekabr –Toshkent, O‘zbekiston, 2017. 48-b.

3. M. R. Fayziyeva, D. M. Sayfurov, N. S. Xaytullayeva Informatika va axborot texnologiyalari: umumiy o‘rta ta’lim maktablarining 9-sinfi uchun darslik: - Toshkent: Tasvir, 2020. – 112 b.

4. Boltayev B., Azamatov A., Asqarov A., Sodiqov M., Azamatova G. Informatika va hisoblash texnikasi asoslari. Umumiy o‘rta ta’lim maktablarining 9-sinfi uchun darslik. Toshkent: “Cho‘lpon” nomidagi NMIU, 2015. – 160 b.

5. Chris Roffey. Computer science. Programming book for Python. – USA: Cambridge university press. 2017, – p. 204

6. Chris Roffey. Python basics. Coding club. Level 1,2. – USA: Cambridge university press. 2012, – p. 85

7. Eric Matthes. Python crash course: a hands-on, project-based introduction to programming. – San-Francisco: No Starch Press, 2015. – p. 562

8. Matt Harrison. Illustrated guide to Python 3. 2017, – p. 267

9. Dan Bader. Python tricks the book. Anja Pircher Design, 2017, – p. 299

10. https://pythonru.com/

11. https://python-scripts.com/

12. https://www.rupython.com/

N.N. Zaripov

ZAMONAVIY DASTURLASH TILLARI

Muharrir: **E.Eshov**
Tex. muharrir: **D.Abduraxmonova**
Musahhih: **M.Shodiyeva**
Badiiy rahbar: **M.Sattorov**

Nashriyot litsenziyasi № 022853. 04.03.2022.
Original maketdan bosishga ruxsat etildi: 04.07.2024. Bichimi 60x84. Kegli 16 shponli. “Times New Roman” garnitura 1/16.
Elektrografik usulda. Oddiy bosma qog‘ozi.
Bosma tabog‘i 9. Adadi 100. Buyurtma № 420

KAMOLOT

“BUXORO DETERMINANTI” MCHJ
bosmaxonasida chop etildi.
Buxoro shahar Namozgoh ko‘chasi 24 uy
Tel:. + 998 91 310 27 22

ISBN 978-991072941-6