

**O‘ZBEKISTON RESPUBLIKASI OLIY VA O‘RTA MAXSUS
TA‘LIM VAZIRLIGI**

SAMARQAND DAVLAT UNIVERSITETI

AXATOV AKMAL, NAZAROV FAYZULLO

**PYTHON
TILIDA DASTURLASH ASOSLARI
(I-QISM)**

*(Amaliy matematika, kompyuter ilmlari va dasturlash texnologiyalari
yo‘nalishlari uchun o‘quv qo‘llanma)*



SAMARQAND - 2020

UDK: 681.14(075)
BBK 32.97

Python tilida dasturlash asoslari. O‘quv qo‘llanma.– Samarqand: SamDU nashri, 2020 yil, – 180 bet.

Mazkur qo‘llanma dasturlash asoslari fanini Python dasturlash tili misolida qarab o‘tilgan. Python dasturlash tili tarkibidagi barcha turdagi operatorlar, kalit so‘zlar va qoidalar batafsil keltirib o‘tilgan. Operatorlar, kalit so‘zlar va buyruqlarning umumiy ko‘rinishlari va yozilishlari bir nechta dasturlar yordamida tushuntirib o‘tilgan. Qo‘llanmada Python tilining barcha imkoniyatlari bo‘yicha nazariy tushunchalar hamda bu tushunchalarni o‘zlashtirish uchun masalalar yechimlari keltirib o‘tilgan. Har bir mavzu bo‘yicha, mavzuni mustahkamlash uchun nazariy savollar hamda mustaqil ishlash uchun topshiriqlar ham keltirib o‘tilgan. Mazkur qo‘llanma oliy o‘quv yurtlari talabalari va magistrantlari, litsey kasb hunar kollej o‘quvchilari hamda mustaqil o‘rganuvchilar uchun qulay vosita hisoblanadi.

Mualliflar

Axatov Akmal Rustamovich – *O‘zbekiston milliy universiteti Jizzax filiali ilmiy va innovatsiyalar bo‘yicha direktor o‘rinbosari, texnika fanlari doktori, professor.*

Nazarov Fayzullo Maxmadiyarovich – *Samarqand Davlat Universiteti “Axborotlashtirish texnologiyalari” kafedrasi o‘qituvchisi.*

Ma’sul muharrir

Jumanov Isroil Ibragimovich – *Samarqand Davlat Universiteti, texnika fanlar doktori, professor.*

Taqrizchilar

Bobomuradov Ozod Jo‘rayevich – *Toshkent axborot texnologiyalari universiteti professori, texnika fanlar doktori.*

Raximov Nodir – *O‘zbekiston milliy universiteti Jizzax filiali o‘quv ishlari bo‘yicha direktor o‘rinbosari, texnika fanlar doktori.*

Qobilov Samijon Soliyevich – *Samarqand Davlat Universiteti “Axborotlashtirish texnologiyalari” kafedrasi dotsenti, texnika fanlar nomzodi.*

ISBN 978-9943-6646-8-5

©Samarqand davlat universiteti, 2020

MUNDARIJA

Soʻz boshi.....	6
Kirish.....	7
I-BOB. Algoritmash va dasturlash, python dasturlash tili.....	8
1.1 Algoritm, algoritmning berilish usullari, xossalari va turlari	8
Algoritmning tasvirlash usullari.....	9
Algoritmning xossalari.....	11
Chiziqli algoritmlar.....	12
Tarmoqlanuvchi algoritmlar.....	14
Takrorlanuvchi algoritmlar.....	15
1.2 Dasturlash tillarining klassifikatsiyasi va python dasturlash tili...	19
Dasturlash tillarining klassifikatsiyasi.....	19
Python dasturlash tili va uning imkoniyatlari	20
1.3 Python dasturlash tilida interaktiv rejim, birinchi dastur, kiritish va chiqarish operatorlari.....	23
Python dasturlash tilining IDLE rejimida interaktiv dastur yaratish.....	23
Python dasturlash tilida faylli dastur yaratish	27
1.4 Python dasturlash tili tarkibidagi arifmetik amallar va mantiqiy amallar	30
Arifmetik amallari.....	30
Taʼminlash operatori.....	33
Mantiqiy amallar.....	34
1.5 Python dasturlash tili tarkibidagi matematik funksiyalar va ifodalar.....	38
Python tilida ifodalar.....	38
Python dasturlash tilida matematik funksiyalar.....	39
II-BOB. Python tilida chiziqli, tarmoqlanuvchi va takrorlanuvchi jarayonlarni dasturlash.....	45
2.1 Python dasturlash tilida chiziqli jarayonlarni dasturlash.....	45
Python tilida chiziqli dasturlar.....	45
type() va help() funksiyalari.....	47
2.2 Python dasturlash tilida tarmoqlanuvchi jarayonlarni dasturlash.....	50
Qisqa shartli operator va uning umumiy koʻrinishi.....	50
Toʻliq shartli operator va uning umumiy koʻrinishi.....	52
elif operatori va uning umumiy koʻrinishi.....	53
2.3 Python dasturlash tilida takrorlanuvchi jarayonlar va parametr boʻyicha dasturlash.....	56
Takrorlanuvchi jarayonlarni dasturlash.....	56
for(sikl) operatori va uning umumiy koʻrinishi.....	58
Ichma ich sikllarni tashkil qilish.....	

2.4 Sikl qadamlarini tashlab o‘tish va sikllarni muddatidan oldin tugatish.....	64
Break operatori va uning umumiy ko‘rinishi.....	70
Continue operatori va uning umumiy ko‘rinishi.....	73
2.5 Python dasturlash tilida shartli takrorlanuvchi jarayonlar dasturlash.....	74
Shartli sikl operatori.....	75
while operatori va uning umumiy ko‘rinishi.....	76
III-BOB. Python dasturlash tilida funksiyalar.....	83
3.1 Python dasturlash tilida funksiyalar python dasturlash tilida funksiyalarni yaratish va ulardan foydalanish.....	83
Qism dasturlar.....	83
Funksiya tanasini faollashtirish.....	85
Global va lokal o‘zgaruvchilar.....	89
Funksiyaga argument berishni soddalashtirish.....	92
3.2 Python dasturlash tilida ko‘p qiymat qaytaruvchi funksiyalar va ulardan foydalanish.....	95
Ko‘p qiymat qaytaruvchi funksiyalar.....	95
Ko‘p qiymat qaytaruvchi funksiyalarni shakllantirish.....	96
3.3 Rekursiv funksiyalar.....	100
Rekursiv funksiyalar.....	100
IV-BOB. Python dasturlash tilida murakkab turlar ro‘yxat, kortej, lug‘at,to‘plam va massivlar.....	103
4.1 Python dasturlash tilida ro‘yxat va kortejlar.....	103
Ro‘yxatlar.....	103
Kortejlar.....	110
4.2 Python dasturlash tilida lug‘at va to‘plamlardan foydalanish.....	112
Lug‘atlar.....	112
To‘plamlar.....	114
4.3 Python dasturlash tilida massivlar va ulardan foydalanish.....	117
Bir o‘lchovli massivlar.....	119
Massiv elementlarini funksiyalar orqali va klaviatura yordamida hosil qilish.....	121
Massiv elementlari ustida aniqlangan amallar.....	123
Ikki o‘lchovli massivlar.....	125
Random funksiyasi.....	128
V-BOB. Python dasturlash tilida satrlar va fayllar.....	133
5.1 Python dasturlash tilida satrlar va ulardan foydalanish.....	133
Satrlar.....	133

Satrlar ustida aniqlangan amallar.....	135
5.2 Python dasturlash tilida fayllar va ulardan foydalanish.....	141
Fayllarni faollashtirish.....	142
Fayllar ustida amallar bajarish.....	145
Fayldan ma'lumot o'qish.....	145
Fayl tarkibiga ma'lumotlarni yozish.....	149
Fayl tarkibidagi ma'lumotlarni o'chirish.....	151
VI-BOB. Python dasturlash tilida grafika.....	156
6.1 Python dasturlash tili tarkibida grafiklar chizish va ularni qayta ishlash.....	156
Grafik muhitini faollashtirish.....	156
Tekislikda chizma va shakllar chizish.....	157
Chizmalarni alohida faylda saqlash.....	160
Grafikga ma'lumot yozish.....	161
Matematik funksiyalar grafiklarini chizish.....	162
6.2 Python dasturlash tili tarkibida diagrammalar va uch o'lchovli grafiklar chizish.....	166
Diagrammalar.....	166
Uch o'lchovli grafika.....	168
Xulosa.....	171
Glossariy.....	172
Foydalanilgan adabiyotlar ro'yxati.....	175
Ilovalar.....	176

SO‘Z BOSHI

Hozirgi vaqtda barcha sohalarni axborot texnologiyalarisiz tasavvur qilib bo‘lmaydi. Jamiyatning barcha sohalariga axborot texnologiyalarini jadallik bilan kirib borayotgan bir vaqtda, ta‘lim sohasida ham axborot texnologiyalarining o‘rni keskin ortib bormoqda. Hozirda raqamli iqtisodiyot va raqamli texnologiyalarning asosiy bo‘g‘ini hisoblangan dasturlash texnologiyalarini rivojlantirish bugungi kunning dolzarb masalasi hisoblanadi. Bugungi kunda ma‘lumotlar oqimining ko‘pligi tufayli ularni qisqa vaqt ichida qayta ishlash muommosi ham ortib bormoqda. Ma‘lum bir sohaga oid muommolarni hal etadigan avtomatlashtirilgan tizimlar yaratish shu soha mutaxassislarining asosiy vazifasi hisoblanadi. Zamonaviy jamiyatda tobora o‘sib borayotgan axborot oqimi, axborot texnologiyalarining turlit-umanligi, kompyuterda yechiladigan masalalarning murakkablashuvi, ushbu texnologiyalardan foydalanuvchining oldiga bir qator vazifalarni qo‘ydi. Bu vazifalarni bosqichma bosqich raqamli texnologiyalar ya‘ni avtomatlashtirilgan tizimlar orqali hal etish kerak. Axborot tizimi va axborot texnologiyalarining avtomatlashtirilgan elementlarini qo‘llash va avtomatlashtirish asosida yangi axborot texnologiyasini yaratish, jarayonlarni loyihalashtiruvchilarning asosiy vazifalaridan biri hisoblanadi. Mazkur qo‘llanma yuqorida belgilangan vazifalarni hal etish uchun, asosiy uslubiy ta‘minot vazifasini bajaradi. Bu qo‘llanma nafaqat matematik va muhandis yo‘nalishidagi talabalar, balki mustaqil o‘rganuvchilar uchun ham asosiy qo‘llanma hisoblanadi. Python dasturlash tili, dastur tuzishni o‘rganuvchi talablarning barchasiga yaxshiroq javob beradi, shuning uchun hozirgi vaqtda ko‘plab universitetlarning va tadqiqot institutlarining olimlari hamda dasturchi muhandislar orasida bu til mashxur bo‘lib kelmoqda. Qo‘llanma Amaliy matematika va informatika, axborot tizimlarining matematik va dasturiy taminoti yo‘nalishlarining o‘quv rejasi hamda o‘quv dasturi asosida yaratildi. Bunda python dasturlash tilining asoslari, ya‘ni chiziqli dasturlardan boshlab, grafik ma‘lumotlarni qayta ishlashgacha keng yoritib o‘tildi.

KIRISH

Mazkur kitob o'quvchiga dasturlashni boshidan boshlab, qanday dastur tuzishni tez va samarali o'rganishga imkonini beradi. Qo'llanmada python dasturlash tili tarkibidagi amallar, operatorlar, murakkab turlar, fayllar va grafiklar bo'yicha ma'lumotlarni qayta ishlash jarayoni ko'rsatib o'tilgan. Biz bu qo'llanmani tayyorlashda ongli ravishda tushunarli bo'lishi uchun eng sodda jarayonlardan, murakkab jarayonlarga tomon o'rganishga amal qildik. Python dasturlash tili uzoq tarixga bormasada, lekin uning rivojlanishi hozirgi vaqtda eng istiqbolli sohalardan hisoblanadi. Python dasturlash tili o'tgan asrning 80-yillari oxirlarida ishlab chiqila boshlandi. Gido Van Rossum Python dasturlash tilini 1980-yillarda yaratgan va u til 9 yildan so'ng, ya'ni 1989 yil dekabrda Gollandiyadagi matematika va informatika laboratoriya markazida ishlab chiqilgan. Python istisno holatlarini ko'rib chiqishga va Amoeba operatsion tizimiga ta'sir ko'rsatishga qodir bo'lgan ABC dasturlash tilining avlodi hisoblanadi. Van Rossum Pythonning asosiy muallifidir va u 2018 yilgacha tilni rivojlantirish bo'yicha bir qancha ishlar olib borgan.

Python dasturlash tili boshqa dasturlash tillaridan farqli ravishda, mukammal darajada ishlab chiqilgan. Python dasturlash tili, boshqa dasturiy vositalarni boshqarish va ularning tarkibiy qismlarini mustaqil boshqarishni amalga oshiradi. Aslida, Python ko'p maqsadli dasturlash tili sifatida o'rganilishi mumkin, bu dasturlash tili yordamida bir qancha jarayonlarni dasturlash imkoni yaratiladi. Python dasturlash tilida, bir vaqtning o'zida, boshqa dasturlash tillaridan farqli ravishda, bir nechta turdagi dasturlar yaratish inkoniyati mavjud, ya'ni:

- amaliy dasturiy maxsulotlar;
- web ilovali dasturiy maxsulotlar;
- 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. Bunday imkoniyatlar, albatta, dastur yaratuvchilar ish faoliyati samaradorligini keskin ortishiga xizmat qiladi.

I-BOB. ALGORITMLASH VA DASTURLASH, PYTHON DASTURLASH TILI

1.1 ALGORITM, ALGORITMNING BERILISH USULLARI, XOSSALARI VA TURLARI

Reja:

1. *Algoritm;*
2. *Algoritmning tasvirlash usullari;*
3. *Algoritmning xossalari;*
4. *Chiziqli algoritmlar;*
5. *Tarmoqlanuvchi algoritmlar;*
6. *Takrorlanuvchi algoritmlar.*

Tayanch soʻzlar: *Algoritm, algoritm xossalari, algoritm turlari chiziqli algoritmlar, tarmoqlanuvchi algoritmlar, takrorlanuvchi algoritmlar.*

Axborot texnologiyalarini rivojlanishining asosiy boʻgʻini bu algoritmlash jarayonidir. Biror bir masala yoki muommoni elektron hisoblash mashinasida hisoblash uchun, albatta, berilgan masalani matematik modeli, algoritmi va biror bir algoritmlash tili asosida dasturini yaratish kerak boʻladi. Muommo yoki masalani matematik modelini tuzish deganda berilgan masalani matematik tushunchalar bilan ifodalanishiga aytiladi. Muommo yoki masalani yechish qadamlari albatta algoritm bilan bogʻliqdir. Algoritm soʻzi va tushunchasi IX asrda yashab ijod etgan buyuk vatandoshimiz alloma Muhammad al-Xorazmiy nomi bilan uzviy bogʻliq. Algoritm soʻzi Al-Xorazmiy nomini Yevropa olimlari tomonidan oʻzgacha talaffuz qilinishidan yuzaga kelgan. Al-Xorazmiy birinchi boʻlib oʻnlik sanoq sistemasining tamoyillarini va undagi toʻrtta amallarni bajarish qoidalarini asoslab bergan. Masala yoki muommoni hal etish uchun yuqorida keltirib oʻtilgandek, uni algoritmini tuzish kerak boʻladi.

Taʼrif: Algoritm- bu biror bir masala yoki muommoni hal etish uchun kerak boʻlgan chekli sondagi buyruqlar ketma- ketligi.

Masala algoritmini tuzish tartibini quyidagi masala orqali koʻrib chiqamiz.

Misol. *a,b,c uchburchakning yuzasini hisoblash algoritmi tuzilsin*

1. *a,b,c kesmalar kiritilsin;*

2. Berilgan kesmalar yordamida $P=(a+b+c)/2$ hisoblansin;

3. $S = \sqrt{p(p-a)(p-b)(p-c)}$ hisoblansin.

Demak, masalani yechish ketmaketligini algoritm sifatida qarash mumkin.

Algoritmning tasvirlash usullari

Masala algoritmini tuzish vaqtida, bu masala qanday sohaga tegishliligiga qarab algoritm bir necha xil ko‘rinishda tasvirlanishi mumkin. Algoritmni bir nechta tasvirlash usullari mavjud. Iqtisodiy masalalar algoritmi jadvallar ko‘rinishida, matematik masalalar formula yoki grafik usullarda va sanoatda esa so‘zlar yordamida tasvirlanish mumkin.

1.Algoritmning so‘zlar orqali ifodalanishi. Bu usulda ijrochi uchun beriladigan har bir ko‘rsatma, jumlar, so‘zlar orqali buyruq shaklida beriladi.

Misol. To‘g‘ri to‘rtburchakning yuzasi va perimetrini hisoblash algoritmi tuzilsin.

1. boshlanish
2. a, b lar kiritilsin;
3. $S=ab$ va $P=2(a+b)$ hisoblansin;
4. S va P chiqarilsin;
5. Tamom.

2.Algoritmning grafik shaklida tasvirlanishida algoritm maxsus geometrik figuralar yordamida tasvirlanadi va bu grafik ko‘rinishi blok-sxema deyiladi. Algoritmning blok-sxema ko‘rinishida tasvirlanishi beriladigan buyruqlar qandaydir shakllar orqali ifodalanadi.

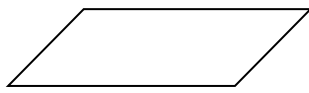
Algoritmning blok-sxemalar yordamida tuzishda foydalaniladigan asosiy sodda geometrik figuralar quyidagilardan iborat.



– Ellips u algoritmning boshlanishi yoki tugallashini belgilaydi;



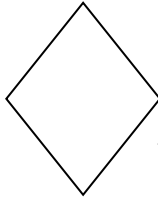
– To‘g‘ri burchakli to‘rtburchak, qiymat berish yoki tegishli ko‘rsatmalarni bajarish jarayonini belgilaydi;



– Parallelogramm, ma'lumotlarni kiritish yoki chiqarishni belgilaydi.



– Yordamchi algoritmgaga murojatni belgilaydi.

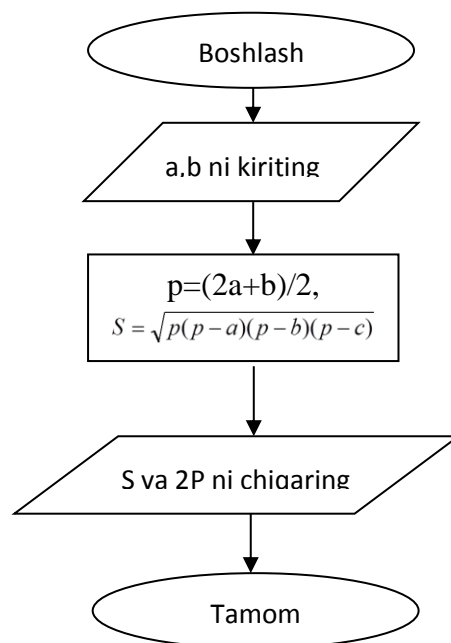


– Romb, shart tekshirishni belgilaydi va shart bajarilsa "ha", tarmoq bo'yicha, aks holda "yo'q"-tarmog'i bo'yicha amallar bajarilishini ta'minlaydi.

- \longrightarrow Strelka - amallar ketma-ketligining bajarilish yo'nalishini ko'rsatadi.

Yuqorida ko'rilgan algoritmlarning tasvirlash usullarining asosiy maqsadi - qo'yilgan masalani yechish uchun zarur bo'lgan amallar ketma-ketligining eng qulay holatini aniqlash va shu bilan foydalanuvchi tomonidan programma yozishni yanada osonlashtirishdan iborat. Aslida programma ham algoritmnining boshqa bir ko'rinishi bo'lib, u insonning kompyuter bilan muloqotini qulayroq amalga oshirish uchun mo'ljallangan.

Misol. Teng yonli uchburchakning tomonlari berilganda uning yuzi va perimetrini topish algoritmi blok-sxemasini yozing.



Algoritmning xossalari

Masala yoki mummoni hal etish jarayoni uchun keltirilgan algoritmlar ma'lum bir xususiyatlarga bo'ysinish kerak. Bu xususiyatlarni e'tiborga olib algoritmlar quyidagi xossalarga ega.

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.

Tuzilayotgan algoritmlar doimo bir xil yo'nalishlarda bo'lmaydi. Masala yoki muommoni hal etish ma'lum bir algoritm bo'yicha amalga oshiradi. Masalalar va ularning algoritmlari ham ma'lum bir turlarga bo'linadi. Har qanday murakkab masalani ham uchta asosiy strukturaga keltirish mumkin. Algoritmlarni umumlashtirgan holda quyidagi turlarga ajratamiz:

- Chiziqli algoritmlar;
- Tarmoqlanuvchi algoritmlar;
- Takrorlanuvchi algoritmlar.

Chiziqli algoritmlar

Masalani hal etish uchun tuzilgan algoritm tarkibidagi buyruqlar ketma ketligi uzluksiz bo'lishi mumkin yoki qandaydir holatlarda shartlar asosida uzluksizlik tarqatilishi mumkin. Chiziqli algoritmlarda esa buyruqlar ketma-ketligi doim uzluksiz bo'ladi.

*Tarif: Algoritm bajarilish vaqtida hech qanday to'siqqa uchramasdan buyruqlar ketma-ketligi uzluksiz bajarilsa bunday algoritmlar **chiziqli algoritm** deyiladi.*

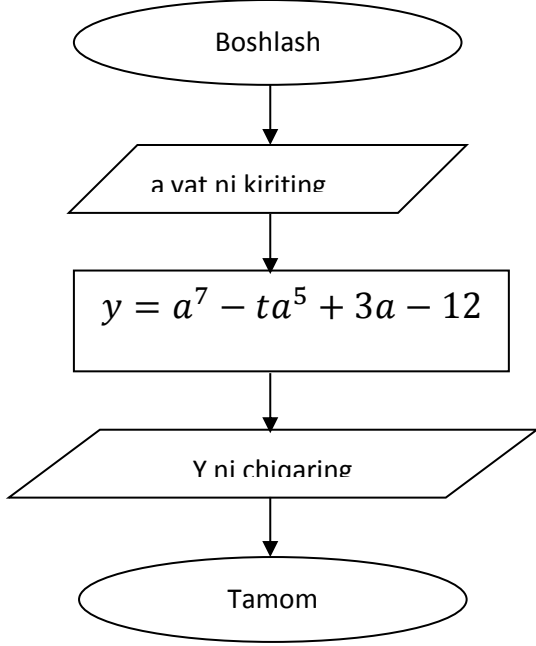
Demak, algoritm bajarilishida hech qanday shart bo'lmaslik va uzluksizlik yo'qolmaslik kerak. Algoritm tuzish vaqtida uning turini

aniqlash uchun masala tarkibida hech qanday shart yoki takrorlanish bo‘lmaslik kerak. Har qanday masala algoritmini ham uchta chiziqli, shartli va takrorlanuvchi algoritmlar yordamida tasvirlash mumkin. Chiziqli algoritmlar bajarilish vaqtida buyruqlar ketma-ketligi buzilmasdan davom etadi.

Misol: a ni qiymati berilganda quyidagi funktsiyani hisoblash algoritmini keltiring.

$$y = a^7 - ta^5 + 3a - 12$$

Bu masala algoritmini tuzish jarayoni a va t noma'lumning qiymati berilganda y funktsiyaning natijasi hisoblanish kerak. Demak, faqat a va t ning qiymati kiritilib y funktsiyaning natijasi hisoblanish kerak bo'ladi. Berilgan masala uchun algoritmning quyidagicha ya'ni so'zlar va blok-sxema ko'rinishida ko'rinishida bo'ladi.

Algoritmni so'zlar yordamida tasvirlanishi	Algoritmni blok-sxema yordamida tasvirlanishi
<ol style="list-style-type: none"> 1. boshlanish 2. a va t kiritilsin; 3. $y = a^7 - ta^5 + 3a - 12$ hisoblansin; 4. Tamom. 	 <pre> graph TD Start([Boshlash]) --> Input[/a va t ni kiriting/] Input --> Process[y = a^7 - ta^5 + 3a - 12] Process --> Output[/Y ni chiqaring/] Output --> End([Tamom]) </pre>

Blok-sxemalar bilan ishlashni yaxshilab o'zlashtirib olish zarur, chunki bu usul algoritmlarni ifodalashning qulay vositalaridan biri bo'lib, programma tuzishni osonlashtiradi, programmalash qobiliyatini mustahkamlaydi. Algoritmik tillarda blok - sxemaning asosiy strukturalariga maxsus operatorlar mos keladi. Shuni aytish kerakki, blok-

sxemalardagi yozuvlar odatdagi yozuvlardan katta farq qilmaydi. Faqat ketma-ket bajariladigan amallardan tashkil topgan algoritmlarga-chiziqli algoritmlar deyiladi. Bunday algoritmni ifodalash uchun ketma-ketlik strukturasi ishlatiladi. Strukturada bajariladigan amal mos keluvchi shakl bilan ko'rsatiladi. Aslida programma ham algoritmning boshqa bir ko'rinishi bo'lib, u insonning kompyuter bilan muloqotini qulayroq amalga oshirish uchun mo'ljallangan.

Tarmoqlanuvchi algoritmlar.

Masala yoki muommoni hal etish jarayonida qandaydir shartlarga duch kelinsa, bu masala shartlarning bajarilishi asosida amalga oshiriladi. Masala yechimi aniqlanish jarayonida tuzilayotgan algoritmlar shartlar asosida tarmoqlanishi mumkin, ya'ni shart bajarilish asosida chin qiymat qabul qilganda algoritmning bir qismi yolg'on qiymat qabul qilganda esa algoritmning boshqa qismi bajariladi.

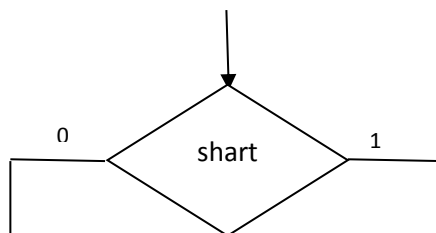
Algoritm bajarilish davomida har doim ham buyruqlar ketma-ketligi bajarilavermaydi, shunday holatlar ham mavjudki, algoritm tarkibida shartlar asosida buyruqlar ketma-ketligi tarmoqlanib ketadi.

*Ta'rif: Algoritm bajarilish vaqtida buyruqlar ketma-ketligi shartlar asosida u yoki bu qismga tarmoqlanishiga **tarmoqlanuvchi algoritmlar** deyiladi.*

Tarmoqlanuvchi algoritmlar quyidagicha so'zlar yordamida tasvirlanadi.

Agar (shartli ifoda) **u holda** (hisoblansin) **aks holda** (hisoblansin)

Tarmoqlanuvchi algoritmlar quyidagicha blok-sxema yordamida tasvirlanadi.



Shart chin bo'lganda algoritm 1 tomonga yolg'on bo'lganda esa 0 tomonga harakatlanadi. 1 yoki 0 o'rniga ha yoki yolg'on, + yoki - larni yozish mumkin. Algoritmni ifodalovchi blok-sxema ko'rinishiga e'tibor

qaratsangiz romb belgisi tarkibiga algoritm sharti keltiriladi. Agar algoritm sharti natijasi chin qiymat qabul qilsa, algoritm + tarafdagi buyruqlarga o'tadi aks holda – tarafdagi buyruqlar ketma-ketligiga o'tadi.

Misol. Kvadrat tenglamaning a, b, c koeffitsientlari berilganda, uning ildizlarini hisoblash algoritmini tuzing.

Algoritmni soʻzlar yordamida tasvirlanishi	Algoritmni blok-sxema yordamida tasvirlanishi
<p>1. boshlanish;</p> <p>2. a, b, c lar kiritilsin;</p> <p>3. $D = b^2 - 4ac$ hisoblansin;</p> <p>4. Agar $D > 0$ u holda $x_1 = (-b + \sqrt{d}) / (2a)$ $x_2 = (-b - \sqrt{d}) / (2a)$ x_1 va x_2 chiqarilsin</p> <p>aks holda agar $d < 0$ u holda (yechim yuq) aks $x = -b / (2a)$ x chiqarilsin;</p> <p>5. Tamom.</p>	<pre> graph TD Start([Boshlanish]) --> Input[/a, b, c kiritilsin/] Input --> Calc[D = b^2 - 4ac] Calc --> Dec1{D > 0} Dec1 -- 1 --> CalcSol1["x1 = (-b + sqrt(d)) / (2a) x2 = (-b - sqrt(d)) / (2a)"] Dec1 -- n --> Dec2{D < 0} Dec2 -- 1 --> CalcSol2["x = -b / (2a)"] Dec2 -- n --> NoSol[/Yechim yuq/] CalcSol1 --> Output[/x1 va x2/] CalcSol2 --> Output NoSol --> End([Tamom]) Output --> End </pre>

Takrorlanuvchi algoritmlar.

Masala yoki muommoni hal etishda ba'zi jarayonlar bir necha marta takrorlanish mumkin. Algoritm tarkibida biror parametr qandaydir shartga bog'liq ravishda oshishi yoki kamiyish hisobiga takrorlanish jarayonlari vujudga kelishi mumkin. Bunda bu jarayonlar takrorlanishlar orqali amalga oshiriladi.

Ta'rif: Algoritmning ma'lum bir qismi qandaydir shartlar asosida ikki va undan ortiq bajarilishiga **takrorlanuvchi algoritmlar** deyiladi.

Takrorlanuvchi algoritmlar bajarilish vaqtida uning qandaydir qismi bir necha marta takrorlanadi. Algoritm bajarilish davomida har doim ham buyruqlar ketma - ketligi bajarilavermaydi, shunday holatlar ham

mavjudki, algoritm tarkibida shartlar asosida buyruqlar ketma-ketligi bir necha marta takrorlanish ham mumkin. Masalan 1 dan n gacha sonlarning kvadratlar yig'indisi yoki yig'indisi x ga teng sinuslarning kvadratlar yig'indisi kabi masalalar algoritmi shartlar asosida takrorlanishlar yordamida tuziladi.

Takrorlanuvchi algoritmlar asosan ikki xil ko'rinishda bo'ladi.

-Takrorlanishlar soni oldindan aniq;

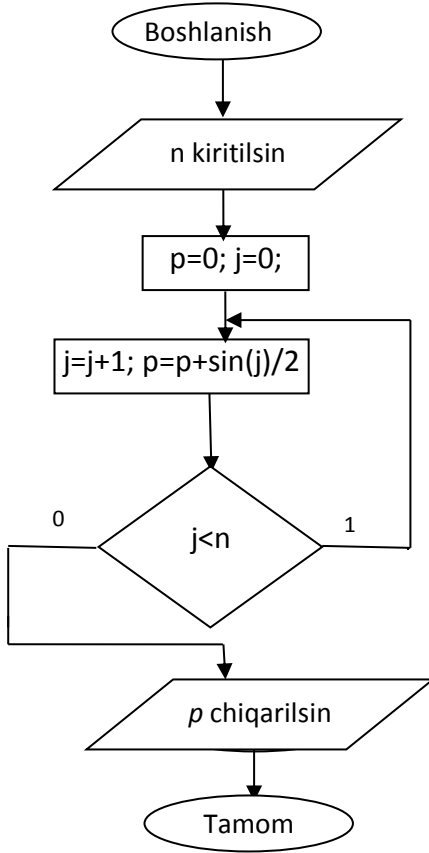
-Takrorlanishlar soni qandaydir shartlarga bog'liq.

Takrorlanishlar soni oldindan aniq bo'lgan masalalarda algoritm bajarilishini bitta parameter soni bilan bog'lanadi. Takrorlanishlar soni qandaydir shartlarga bog'liq bo'lgan holatlarda takrorlanishni shartli jarayon bilan ifodalanadi.

Misol: Quyidagi yig'indini hisoblang.

$$p = \sin(1) + \sin(2)/2 + \dots + \sin(n)/n$$

Berilgan masalani yechish algoritmi tarkibida takrorlanish soni oldindan ma'lum chunki bu *n ga* bog'liqdir.

Algoritmni so'zlar yordamida tasvirlanishi	Algoritmni blok-sxema yordamida tasvirlanishi
<p>1.boshlanish; 2.n soni kiritilsin; 3.p=0,j=0; 4. j=j+1; p=p+sin(j)/2; 5.Agar j<n uholda 4-ga qaytilsin aks holda p chiqarilsin; 6.tamom.</p>	 <pre> graph TD Start([Boshlanish]) --> Input[/n kiritilsin/] Input --> Init[p=0; j=0;] Init --> Loop[j=j+1; p=p+sin(j)/2] Loop --> Decision{j < n} Decision -- 0 --> Loop Decision -- 1 --> Output[/p chiqarilsin/] Output --> End([Tamom]) </pre>

Yuqorida keltirilgan algoritmning turlari asosida ixtiyoriy masala yoki muommolarni hal etish imkoniyati yaratiladi.

Nazariy savollar

- 1 Algoritm so'zining ma'nosi?
- 2 Algoritmning ta'rifi?
- 3 Algoritm necha xil usulda tasvirlanish mumkin?
- 4 Blok-sxema buyruqlar shakllarini tushuntirib bering?
- 5 Algoritmning xossalari va ularning ta'riflari?
- 6 Algoritm turlari?
- 7 Chiziqli algoritm deb nimaga aytiladi?
- 8 Tarmoqlanuvchi algoritm deb nimaga aytiladi?
- 9 Takrorlanuvchi algoritm deb nimaga aytiladi?

Mustaqil ishlash uchun topshiriqlar

1. Aylananing uzunligi L berilgan. Uning radiusi R va yuzasi S aniqlansin.
 $L=2\cdot\pi\cdot R$,
 $S=\pi\cdot R^2$.
2. Aylananing yuzasi S berilgan. Uning diametri D va uzunligi L aniqlansin. $L=2\cdot\pi\cdot R$, $S=\pi\cdot R^2$.
3. Sonlar o'qida ikkita nuqta orasidagi masofa aniqlansin. $|x_2-x_1|$.
4. Sonlar o'qida A , B , C nuqtalar berilgan. AC va BC kesmalarning uzunligini va kesmalar uzunligining yig'indisini topuvchi algoritm tuzilsin.
5. Sonlar o'qida A , B , C nuqtalar berilgan. C nuqta A va B nuqtalar orasida joylashgan AC va BC kesmalar uzunligining ko'paytmasini toping.
6. To'g'ri to'rtburchakning qarama-qarshi uchlari koordinatlari berilgan. Uning tomonlari koordinata o'qiga parallel. To'g'ri to'rtburchakning perimetri va yuzasi aniqlansin.
7. Tekislikda berilgan ikki nuqta (x_1, y_1) va (x_2, y_2) orasidagi masofa topilsin. $\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$
8. Uchburchakning uchta tomoni uchlari koordinatlari berilgan (x_1, y_1) , (x_2, y_2) , (x_3, y_3) . Ikki nuqta orasidagi masofa topilsin.
- 9 da berilgan. Uchburchakning yuzasi va perimetrini toping.
 $S = \sqrt{p\cdot(p - a)\cdot(p - b)\cdot(p - c)}$,

$$p=(a+b+c)/2$$

10. Uchta A, B, C butun sonlar berilgan. Jumlani rostlikka tekshiring: “ A, B, C sonlarining faqat bittasi musbat son”.

11. Uchta A, B, C butun sonlar berilgan. Jumlani rostlikka tekshiring: “ A, B, C sonlardan ikkitasi musbat son”.

12. Musbat butun son berilgan. Jumlani rostlikka tekshiring: “Berilgan son ikki xonali juft son”.

13. Musbat butun son berilgan. Jumlani rostlikka tekshiring: “Berilgan son uch xonali toq son”.

14. Jumlani rostlikka tekshiring: “Berilgan uchta butun sonlarning hech bo‘lmaganda 2 tasi bir biriga teng”.

15. Jumlani rostlikka tekshiring: “Berilgan uchta butun sonlarning hech bo‘lmaganda bir jufti o‘zaro qarama-qarshi”.

16. Uch xonali son berilgan. Jumlani rostlikka tekshiring: “Ushbu sonning barcha raqamlari xar xil”

15. N ($N>0$) butun son berilgan. Shu sonning kvadratini quyidagi mula asosida hisoblovchi algoritm tuzilsin:

$$N^2 = 1+3+5+ \dots + (2N-1).$$

har bir qo‘shiluvchidan keyin natijani ekranga chiqarib boring. Natijada ekranda 1 dan N gacha bo‘lgan sonlarning kvadratlari chiqariladi.

16. A xaqiqiy va N ($N>0$) butun sonlari berilgan. A ning N - darajasini aniqlovchi algoritm tuzilsin: $A^N = A * A * \dots * A$

17. A xaqiqiy va N ($N>0$) butun sonlari berilgan. Bitta ssikldan foydalanib A ning 1 dan N gacha bo‘lgan barcha darajasini chiqaruvchi algoritm tuzilsin.

18. A xaqiqiy va N ($N>0$) butun sonlari berilgan. Bitta ssikldan foydalanib quyidagi A ning 1 dan N gacha bo‘lgan barcha darajalarini chiqaruvchi va yig‘indini hisoblash algoritmi tuzilsin:

$$S=1+A+ A^2+ \dots + A^N.$$

19. X xaqiqiy va N ($N>0$) butun sonlari berilgan. Bitta ssikldan foydalanib quyidagi X ning 1 dan N gacha bo‘lgan barcha darajalarini chiqaruvchi va yig‘indini hisoblovchi algoritm tuzilsin:

$$S=1-X+ X^2- X^3+ \dots +(-1)^N X^N.$$

Shartli operatoridan foydalanmang.

20. N ($N > 0$) butun sonlari berilgan. Bitta ssikldan foydalanib quyidagi yig'indini hisoblash algoritmi tuzilsin:

$$S = 1! + 2! + 3! + \dots + N!$$

($N!$ ifoda - N faktorial - 1 dan N gacha bo'lgan butun sonlari ko'paytmasini bildiradi: $N! = 1 * 2 * \dots * N$).

1.2 DASTURLASH TILLARINING KLASSIFIKATSIYASI

Reja:

1. Dasturlash tillarining klassifikatsiyasi;
2. Python dasturlash tili va uning imkoniyatlari.

Tayanch so'zlar: axborot tizimi, quyi darajadagi til, o'rta darajadagi til, yuqori darajadagi til, kompyutor, translyator, interpretator.

Dasturlash tillarining klassifikatsiyasi

Axborot tizimi va axborot texnologiyalarining avtomatlashtirilgan elementlarini qo'llash va avtomatlashtirish asosida yangi axborot texnologiyalarini yaratish avtomatlashtirish tizimlarini loyihalashtiruvchilarning asosiy vazifalaridan biridir. Avtomatlashtirilgan tizimlarni yaratish uchun albatta birinchi navbatda muammo obektini infologik yoki diskretli modelini qurish dolzarb hisoblanadi. Infologik yoki diskretli modelni muammo obektiga qarab algoritmlash tillarini qaysi biri asosida yaratish kerakligini tanlab olinish kerak. Elektron hisoblash mashinalarini birinchi avlodlari yaratilishi bilan algoritmlash tillarining rivojlanishi ham boshlandi. Avval algoritmlash tuzuvchi mutaxassislar eng sodda mashina tilini o'zida ifodalovchi kompyuter buyruqlari bilan ishlaganlar.

Bu buyruqlar nol va birlar ketma-ketligidan iborat matnlardan tashkil topgan edi. Keyinchalik insonlar uchun tushunarli bo'lgan mashina buyruqlarini o'zida saqlovchi assembler tili yaratildi. Keyinchalik FORTRAN, BASIC, PASKAL va COBOL singari yuqori darajali tillar ham paydo bo'ldiki, bu tillar yordamida so'z va gaplarning mantiqiy konstruksiyasidan foydalanib algoritmlash imkoniyati yaratildi. Ular buyruqlarni mashina tiliga interpretatorlar va kompilyatorlar yordamida o'tkazar edi.

Algoritmlash tillari yaratilishi bo'yicha uchta turga ajratiladi:

- quyi darajadagi;
- oʻrta darajadagi;
- yuqori darajadagi.

Maʼlumki, maʼlum bir masalani yechish uchun buyruqlar ketma-ketligi yaʼni, algoritm algoritmlash tilida yozilayotganda kamroq buyruqlardan foydalanilsa, bunday tillar darajasi yuqoriroq hisoblanadi.

Quyi darajadagi algoritmlash tillari bevosita kompyuter qurilmalari bilan bogʻliq boʻlib, buyruqlar ularning kodlari bilan yoziladi. Bu kabi buyruqlardan tashkil topgan algoritmlar katta hajmli boʻlib, ularni taxrirlash katta mehnat talab qiladi. Dastlabki kompyuterlar(*ENIAK*, *MESM* va boshqalar) ana shunday tillarda ishlagan.

Oʻrta darajadagi algoritmlash tillari buyruqlarida faqat raqamlar emas, balki insonlar tushunadigan baʼzi soʻzlar ishlatila boshlandi(*Assembler*).

Yuqori darajadagi algoritmlash tillari quyidagicha bosqichlarga boʻlinadi:

Algoritmik(Basic, Pascal, C va b.)

Mantiqiy(Prolog, Lisp va b.)

Obyektga moʻljallangan(Object Pascal, PYTHON, Java va b.)

Algoritmlash tillarida yaratilgan algoritmlar mashina tiliga *translyatorlar* yordamida oʻtkaziladi.

Translyator(translator-tarjimon) biror bir algoritmlash tilida yozilgan algoritmni mashina tiliga tarjima qiladi.

Translyatorlar ikki turda boʻladi:

- Kompilyatorlar(*compiler-yigʻuvchi*) biror bir algoritmlash tilida yozilgan algoritmni mashina tiliga toʻliq oʻqib olib tarjima qiladi;
- Interpretatorlar(*interpreter* - izohlovchi, ogʻzaki tarjimon) biror bir algoritmlash tilida yozilgan algoritmni mashina tiliga satrma - satr tarjima qiladi.

Python dasturlash tili va uning imkoniyatlari

Python dasturlash tilining tarixi oʻtgan asrning 80-yillari oxirlarida boshlangan. Gido Van Rossum Python dasturlash tilini 1980-yillarda yaratgan va u til 1989 yil dekabrda Gollandiyadagi matematika va informatika laboratoriya markazida ishlab chiqilgan. Python istisno holatlarini koʻrib chiqishga va Amoeba operatsion tizimiga taʼsir

ko'rsatishga qodir bo'lgan ABC dasturlash tilining avlodi bo'lgan. Van Rossum Pythonning asosiy muallifidir va u 2018 yilgacha tilni rivojlantirish bo'yicha bir qancha ishlar olib borgan.

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. Bu jarayoning

eng keng tarqalgan vositasi Django web platformasi hisoblanadi. Bu platforma orqali bir qancha keng ommaga tarqalgan mashhur tizimlar ishlab chiqilgan, jumladan, Instagram, Mozilla va hakoza. Django juda ko'p turli xil funktsiyalarni, shu jumladan avtomatik ravishda hosil qilinadigan ma'lumotlar bazasini yaratish imkoniyatlarini taqdim etadi. Python dasturlash tili ko'plab mashhur o'yinlarni ishlab chiqish uchun ishlatiladi, 2000 - yillarning birinchi yarmida Python dasturlash tili sivilizatsiyasida, to'rtinchi o'yinning ichki mantiqiy tuzilishini yozish uchun asosiy vosita sifatida ishlatildi.

Python dasturlash tilining matematik va ilmiy hisoblash jarayonlariga keng qo'llanilish imkoniyati yaratilgan. Python umumiy maqsadlar uchun mo'ljallangan til bo'lib, u matematik paketlar bilan muvaffaqiyatli moslashuvni amalga oshiradi. Python dasturlash tilining asosiy xususiyati uning kengayish imkoniyatidir. Bu esa Python uchun nafaqat C va Fortrandagi algoritmlarning ko'p sonli kutubxonalarini yozildi va moslashtirildi. Python dasturlash tilining boshqa dasturiy vositalar va paketlardan foydalanish imkoniyati mavjud. Pythonni matematik paketga aylantirishning asosiy modullari ishlab chiqilgan.

Python dasturlash tilining eng muhim afzalliklaridan biri shundaki, uning barcha amaliy kutubxonalarini 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.

Nazariy savollar

1. Dasturlash tillarining sinflari va ularning tarkibi?

2. Quyi, o'rta va yuqori darajali dasturlash tillarining vazifalari?

3. *Quyida, o'rtta va yuqori darajali dasturlash tillari?*
4. *Yuqori darajali dasturlash tillarining bosqichlari?*
5. *Python dasturlash tilining imkoniyatlari?*
6. *Python dasturlash tilining boshqa dasturlash tillari bilan bog'liqligi?*
7. *Python dasturlash tilining keng tarqalish sabablari?*
8. *Python dasturlash tilining amaliy, web va matematik paket dasturlarini yaratish imkoniyati?*
9. *Python dasturlash tilining sohalarda qo'llanilishi?*

1.3 PYTHON DASTURLASH TILIDA INERAKTIV REJIM, BIRINCHI DASTUR, KIRITISH VA CHIQRISH OPERATORLARI

Reja:

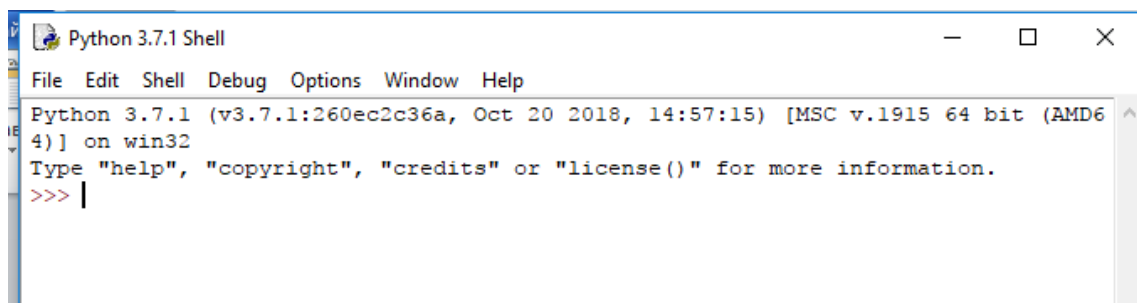
1. *Python dasturlash tilining IDLE rejimida interaktiv dastur yaratish;*
2. *Python dasturlash tilida faylli dastur yaratish;*

Tayanch so'zlar. *IDLE rejimi, interaktiv, faylli dastur, ENTER, restart, int, float, str.*

Python dasturlash tilining IDLE rejimida interaktiv dastur yaratish va o'zgaruvchi turlaridan foydalanish

Python dasturlash tili shaxsiy kompyuterga o'rnatilgandan so'ng IDLE rejimi yuklanadi. IDLE rejimi bu interaktiv rejim hisoblanadi, interaktiv rejimning qulayligi qisqa jarayonlarni dasturlashga juda qo'l keladi. Bunda boshqa dasturlash tillari kabi dasturning umumiy tuzilishini to'liq keltirish shart emas. O'zgaruvchilar tabaqalashtirilgan holda e'lon qilish shartlari keltirilmaydi, xotira bo'yicha muammolar e'tiborga olinmaydi.

Python dasturlash tili IDLE rejimi yuklangandan so'ng quyidagi oyna hosil bo'ladi.



1.2.1-rasm. Python dasturlash tili IDLE rejimining asosiy oynasi.

Python dasturlash tili IDLE rejimida birinchi dastur doimiy an'anaga muvofiq HELLO WORLD so'zini ekranga chiqarishni qarab o'tamiz. Satrli ma'lumotlar bittalik qo'shtirnoq ichiga olib yoziladi.

```
Type "help", "copyright", "credits" or "license()" for more information.
```

```
>>> print('Hello, World');
```

```
    Hello, World
```

```
>>>
```

Bunda >>> belgidan so'ng sonlarni kiritish va hisoblash jarayonlarini to'g'ridan to'g'ri amalga oshirish mumkin. Har bir son yoki amal kiritilgandan so'ng ENTER tugmasi bosiladi va natijaga ega bo'linadi.

Python tilida interaktiv rejim quyidagicha ishlatiladi.

```
Type "help", "copyright", "credits" or "license()" for more information.
```

```
>>> 2.5
```

```
2.5
```

```
>>> 3.2+25
```

```
28.2
```

```
>>> 255656565956565+595659555656565
```

```
851316121613130
```

```
>>>
```


Bu jarayon bizga kalkulyator vazifasini ham bajaradi, kalkulyatordagi xotira muammosi bu yerda mavjud emas. Python dasturlash tili IDLE rejimida o'zgaruvchilar bilan ishlash quyidagicha amalga oshiriladi:

```
Type "help", "copyright", "credits" or "license()" for more information.
>>> a=52
>>> b=2.5
>>> a+b
54.5
>>> a*b+a/b
150.8
>>> a='maktab'
>>> a
'maktab'
>>> 2*a
'maktabmaktab'
>>> 3*a
'maktabmaktabmaktab'
>>>
```

Yuqoridagi dasturga e'tibor bersak, bunda o'zgaruvchilar e'lon qilinmaydi interpretator qiymatga qarab turlarni aniqlaydi. Python dasturlash tilida buyruqlar boshqa dasturlash tillari kabi ; bilan tugatish shart emas. Satrli ma'lumotlarga ko'paytirish amali juda keng imkoniyatda ishlatilishi mumkin. Ixtiyoriy sonni satr ko'rinishga satrni esa son ko'rinishida foydalanish mumkin.

Int(satr) – bu funksiya satrni butun songa aylantiradi

Float(satr) – bu funksiya satrni haqiqiy songa aylantiradi

Str(son) – bu funksiya sonni satrga aylantiradi

```
Type "help", "copyright", "credits" or "license()" for more information.
>>> a='123'
>>> int(a)+2
```

```
125
```

```
>>> a=123
>>> b=str(a)
>>> 2*b
'123123'
>>>
```

Haqiqiy sonlar faqat nuqta bilan yoziladi, vergul esa sonlarni bir biridan ajratish uchun xizmat qiladi. Python dasturlash tili IDLE rejimining qulayligi dasturlashni o'rganayotganda yoki masala kodining ma'lum bir qismini sinab ko'rish vaqtida juda qulaydir. Agar boshqa kompilyatsiya qilinadigan dasturlash tilda ishlasangiz, avval kodni asl dasturlash tilida yozishingiz, keyin kompilyatsiya qilishingiz kerak bo'lgan faylni ishga tushirishingiz kerak bo'ladi.

Python dasturlash tilida sonlarni uch xil turlari aniqlangan:

- Butun son
- Haqiqiy
- Kompleks

Python dasturlash tilida butun va haqiqiy sonlardan foydalanish tajribalarini yuqoridagi misollarda keltirib o'tdik, complex sonlardan python dasturlash tilida quyidagicha foydalanamiz.

```
Type "help", "copyright", "credits" or "license()" for more information.
```

```
>>> a=5
>>> b=complex(a)
>>> b
(5+0j)
>>> c=5+2j
>>> a+c
(10+2j)
>>>
```

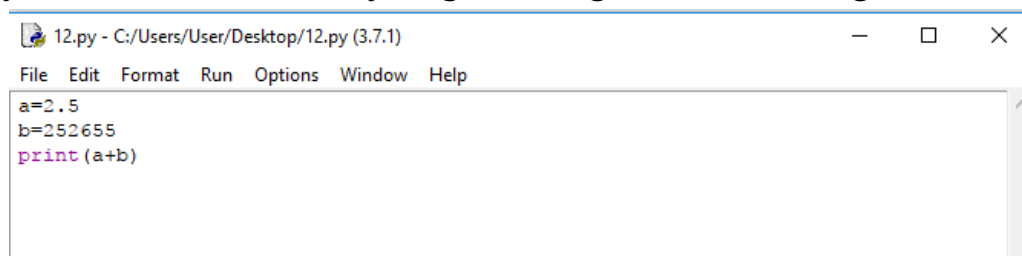
Complex(son) – bu funksiya sonni kompleks songa aylantiradi

Python dasturlash tilining boshqa dasturlash tillaridan ustunligi kompleks sonlar ustida to‘g‘ridan to‘g‘ri amal bajarish imkoniyati mavjudligi.

Python dasturlash tilida faylli dastur yaratish, kiritish va chiqarish operatorlari

Python dasturlash tilida ko‘pgina hollarda, dasturchi masala kodini faylga saqlaydi va natijani fayl kod orqali amalga oshiradi. Bu jarayon boshqa dasturlash tillari kabi alohida fayl yaratish orqali datur tuziladi va yaratilgan dastur RUN tugmasi orqali ishga tushiriladi. Bu jarayon skript yozish deb nomlanadi.

Python dasturlash tilida yaratilgan fayl **.py** kengaytmaga ega bo‘ladi. Python dasturlash tilida skript yozish uchun IDLE interaktiv rejimida **File** → **New File** (yoki **<Ctrl> + N** tugmachalarini bosing) ni tanlash orqali amalga oshiriladi. Yaratilgan faylga dastur tuzib **RUN** menyusi tarkibidan **Run Module F5** komandasi tanlanadi, natijada dastur natijasi interaktiv rejim oynasida aks etadi. Quyidagi dasturga e’tibor bering.



```
12.py - C:/Users/User/Desktop/12.py (3.7.1)
File Edit Format Run Options Window Help
a=2.5
b=252655
print(a+b)
```

1.2.2-rasm. Faylli dastur oynasi

Dastur natijasi quyidagicha bo‘ladi.

```
===== RESTART: C:/Users/User/Desktop/12.py
=====
252657.5
>>>
```

Skript dastur yaratish jarayonida ma’lumotlarni kiritish(o‘qish) va chiqarish(yozish) operatorlari ishlatiladi. Bunda kiritish operatori ma’lumotlarni faqat satr ko‘rinishida qabul qiladi. Sonli ma’lumotlarni qayta ishlash uchun yuqoridagi `int()`; yoki `float()`;funksiyasi yordamida

amalga oshiriladi. Kiritish operatori o'zgaruvchiga birlashtiriladi, kiritish operatorining umumiy ko'rinishi quyidagicha.

o'zgaruvchi = input() yoki **o'zgaruvchi = input('izoh')**

Kiritish operatorining umumiy ko'rinishi quyidagicha.

Print('izoh', o'zgaruvchi)

Kiritish va chiqarish operatorlarini yozilishi quyidagi dastur tarkibida keltirilgan.

Misol. Ikkita a haqiqiy va b butun son berilgan bu sonlarni ko'paytmasini ekranga chiqaring.

```
a=input('a=');
```

```
b=input()
```

```
print('a+b=',float(a)*int(b));
```

Dastur natijasi

```
=====
```

RESTART:

C:/Users/User/Desktop/12.py

```
=====
```

```
a=78545855.254454
```

```
5452154545
```

```
a*b= 4.282441417164835e+17
```

```
>>>
```

Yuqoridagi dasturda **a=input('a=');** izohni chiqargan holda a ga qiymat qabul qiladi, **b=input()** esa izohsiz b ga qiymat qabul qiladi. **Input()** funksiyasi tarkibida bir vaqtni o'zida izohli va izohsiz kiritishni ishlatish mumkin. Kiritish operatori klaviaturadan kiritilgan ma'lumotlarni o'qiydi va o'zgaruvchan nomga yozadi. Chiqarish operatori esa o'zgaruvchidagi ma'lumotni ekranga chop etadi.

Nazariy savollar

1. Python dasturlash tilining IDLE rejimini ishga tushirish?
2. Python dasturlash tilining IDLE rejimida interaktiv dastur tuzish bosqichlari?
3. Int(), satr (), float() va complex() funksiyalarining vazifalari?
4. Komleks sonlardan foydalanish ular ustida amallar?
5. Python faylining kengaytmasi?
6. Pythonda faylli dastur tuzish bosqichlari?

7. Kiritish va chiqarish operatorlarining ko'rinishlari va vazifalari?

Mustaqil ishlash uchu topshiriqlar

1. Kvadratning tomoni a berilgan. Uning perimetri $P=4\cdot a$ va yuzasi aniqlansin. $S=a^2$
2. To'g'ri to'rtburchakning tomonlari a va b berilgan. Uning yuzasi $S=a\cdot b$; va $P=2\cdot(a+b)$ perimetri aniqlansin.
3. Aylananing diametri d berilgan. Uning uzunligi aniqlansin $L=\pi\cdot d$.
4. Kubning yon tomoni a berilgan. Uning hajmini $V=a^3$ va to'la sirti $S=6\cdot a^2$ aniqlansin.
5. Paralelepipedning tomonlari a , b , c berilgan. Uning hajmi $V=a\cdot b\cdot c$ va to'la sirti $S=2\cdot(a\cdot b+b\cdot c+a\cdot c)$ aniqlansin.
6. Doiraning radiusi R berilgan. Uning uzunligi L va yuzasi S aniqlansin. $L=2\cdot\pi\cdot R$, $S=\pi\cdot R^2$.
7. Ikkita son a va b berilgan. Ularning o'rta arifmetigi aniqlansin. $(a+b)/2$ va o'rta geometrigi aniqlansin. $\sqrt{a\cdot b}$.
8. Nolga teng bo'lmagan ikkita son berilgan. Ularning yig'indisi, ko'paytmasi va har birining kvadrati aniqlansin.
9. Nolga teng bo'lmagan ikkita son berilgan. Ularning yig'indisi, ko'paytmasi va har birining moduli aniqlansin.
10. To'g'ri uchburchakning katetlari a va b berilgan. Uning gipotenuzasi c va perimetri P aniqlansin. $c = \sqrt{a^2 + b^2}$, $P = a + b + c$.
11. Umumiy markazga bo'lgan ikkita aylana radiusi berilgan. R_1 va R_2 ($R_1 > R_2$). Ularning yuzalari S_1 va S_2 , ularning ayirmasi S_3 aniqlansin. $S_1=\pi\cdot(R_1)^2$, $S_2=\pi\cdot(R_2)^2$, $S_3=S_1-S_2$.
12. Aylananing uzunligi L berilgan. Uning radiusi R va yuzasi S aniqlansin. $L=2\cdot\pi\cdot R$, $S=\pi\cdot R^2$.
13. Aylananing yuzasi S berilgan. Uning diametri D va uzunligi L aniqlansin. $L=2\cdot\pi\cdot R$, $S=\pi\cdot R^2$.
14. Sonlar o'qida A , B , C nuqtalar berilgan. AC va BC kesmalarning uzunligini va kesmalar uzunligining yig'indisini topuvchi algoritm tuzilsin.
15. To'g'ri to'rtburchakning qarama-qarshi uchlari koordinatlari berilgan. Uning tomonlari koordinata o'qiga parallel. To'g'ri to'rtburchakning perimetri va yuzasi aniqlansin.

16. Tekislikda berilgan ikki nuqta (x_1, y_1) va (x_2, y_2) orasidagi masofa topilsin. $\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$

17. Uchburchakning uchta tomoni uchlari koordinatlari berilgan (x_1, y_1) , (x_2, y_2) , (x_3, y_3) . Ikki nuqta orasidagi masofa topilsin.

18. A , B va C sonlari berilgan. A ni qiymati B ga, B ni qiymati C ga, C ni qiymati A ga almashtirilsin. A , B va C ning yangi qiymatlari ekranga chiqarilsin.

1.4 PYTHON DASTURLASH TILI TARKIBIDAGI ARIFMETIK AMALLAR VA MANTIQUIY AMALLAR

Reja:

1. Arifmetik amallari;
2. Ta'minlash operatori;
3. Mantiqiy amallar.

Tayanch so'zlar: *unar, binar, amal, round, div, mod, true, false.*

Python dasturlash tili tarkibida dastur tuziladigan vaqtda albatta matematik ifodalar, amal ishoralar va mantiqiy amallar ishtirok etishi mumkin. Dastur dasturlash tillarida amallar matematikadan yozilishi bilan farq qiladi. Dasturlash asoslarida amallarni ikki turga ajratamiz:

- arifmetik amallar;
- mantiqiy amallar;

Arifmetik amallar

Berilganlarni qayta ishlash uchun dasturlash tillarida amallarning juda keng majmuasi aniqlangan. Amal - bu qandaydir harakat bo'lib, u bitta (unar) yoki ikkita (binar) operandlar ustida bajariladi, hisob natijasi uning qaytaruvchi qiymati hisoblanadi.

Tayanch arifmetik amallar dasturlash tilida quyidagicha yoziladi.

Matematik ifodasi	Python dasturlash tilida ifodasi	Izoh
+	+	qo'shish
-	-	Ayirish
.	*	ko'paytirish
:	/	bo'lish
Qoldiqli bo'lish	%	Qoldiqli bo'lish

Butun bo'lish	//	Butun bo'lish
Darajaga ko'tarish	**	Darajaga ko'tarish

Dasturlash asoslarida arifmetik amallar matematikadagi amallarni yozilishi bir oz farq qiluvchi holatlari ham mavjud. Amallarni ishlash jarayoni tushunarli bo'lishi uchun, ularni interaktiv rejimda sinab ko'ramiz.

Misol. Arifmetik amallarni bajarilishi

Type "help", "copyright", "credits" or "license()" for more information.

```

>>> y=5
>>> x=2
>>> x+y
7
>>> x-y
-3
>>> x*y
10
>>> y/x
2.5
>>> y%x
1
>>> y//x
2
>>> y**x
25
>>>
>>> y=12.3
>>> x=7
>>> y//x
1.0
>>> y=12
>>> x=2.3
>>> y//2.3
5.0
>>>

```

Yuqoridagi misollarga e'tibor bersangiz // butun bo'lish amali bo'luvchi va bo'linuvchi butun son bo'lsa natija ham butun bo'ladi. Agar bo'luvchi va bo'linuvchining kamida bittasi haqiqiy son bo'lsa ham natija haqiqiy bo'ladi.

Python dasturlash tilida amallarni funksiyalar orqai ham amalga oshirish imkoniyati mavjud.

Python dasturlash tilida ifodasi	Izoh
abs(x)	Modul
round(x)	Yaxlitlash
round(x, n)	n – xonagacha yaxlitlash
pow(x, y)	Darajaga ko'tarish
divmod(x, y)	Butun va qoldikli bo'lish

Round(x) funksiya sonning butun qismigacha yaxlitlaydi, *round(x,n)* funksiyasi sonning *n* – xonasigacha yaxlitlaydi, *pow(x,y)=x**y* ga teng kuchli va *a,b=divmod(x,y)* funksiyasi bir vaqtda *x* ni *y* ga bo'lib butun va qoldiq qismlarini oladi. Funksiyali amallarni ishlash jarayoni tushunarli bo'lishi uchun, ularni interaktiv rejimda sinab ko'ramiz.

Misol. Funksiyali amallarni bajarilishi

```
Type "help", "copyright", "credits" or "license()" for more information.
>>> y=-5
>>> abs(y)
5
>>> x=12.32568
>>> round(x)
12
>>> y=13.652
>>> round(y)
14
>>> round(13.652,2)
13.65
```



```
>>> a=2
>>> b=3
>>> pow(a,b)
8
>>> a=5
>>> b=2
>>> x,y=divmod(a,b)
>>> x
2
>>> y
1
>>>
```

Ta'minlash operatori

Ma'lum bir ifodaning natijasi biror o'zgaruvchiga ta'minlash uchun Python dasturlash tilida "=" belgisi bilan ishlatiladi va uning umumiy ko'rinishi quyidagicha:

<o'zgaruvchi>=<ifoda>;

Python dasturlash tilida taminlash operatori amallar yordamida ham ishlatiladi. Qo'shish qiymat berish bilan (+=); ayirish qiymat berish bilan (-=); ko'paytirish, qiymat berish bilan (*=); bo'lish qiymat berish bilan (/=); bo'lish qoldig'ini olish qiymat berish bilan (%=) va boshqalar. Bu holatlarning umumiy ko'rinishi:

<o'zgaruvchi><amal>=<ifoda>;

$s+=x$; ning ma'nosi $s=s+x$;

Ta'minlash operatorini ishlash jarayoni tushunarli bo'lishi uchun, ularni interaktiv rejimda sinab ko'ramiz.

Misol. Ta'minlash operatorida foydalanish

Type "help", "copyright", "credits" or "license()" for more information.

```
>>> x=5
>>> y=2
>>> x*=y
>>> x
```

```
10
```

```
>>> x/=2
```

```
>>> x
```

```
5.0
```

```
>>> x%=y
```

```
>>> x
```

```
1.0
```

Python dasturlash tilida $s=+x$ amali $s=x$ amaliga teng kuchli hisoblanadi, $s=x+$ va $s=x++$ amallari python dasturlash tilida aniqlanmagan.

Mantiqiy amallar

Mantiqiy turdagi o'zgaruvchi xotiradan 1 bayt joy egallaydi va 0 (false, yolg'on) yoki (true, rost) qiymat qabul qiladi. Mantiqiy tur o'zgaruvchilar qiymatlar o'rtasidagi munosabatlarni ifodalaydigan mulohazalarni rost (true) yoki yolg'on (false) ekanligi tavsifida qo'llaniladi va ular qabul qiladigan qiymatlar matematik mantiq qonuniyatlariga asoslanadi. Mantiqiy o'zgaruvchini quyidagicha faollashtiramiz.

<o'zgaruvchi> = qiymat

Bu yerda qiymat *True* yoki *False* bo'lishi mumkin.

Taqqoslash amallari python dasturlash tilida quyidagi jadvalda berilgan ko'rinishida bajariladi.

Nomi	Pythonda ifodalanishi	Misol	Natija
Tenglik	==	12==50 5==5	False True
Teng emas	!=	100!=50 50!=50	True False
Katta	>	100>50 50>50	True False

Katta yoki teng	\geq	$100 \geq 50$ $50 \geq 50$	True True
Kichik	$<$	$100 < 50$ $50 < 50$	False False
Kichik yoki teng	\leq	$100 \leq 50$ $50 \leq 50$	False True

Python dasturlash tilida uchta mantiqiy bog'lash mulohazalari mavjud, mantiqiy mulohazalar ustida amallar quyidagicha:

- inkor;
- konyunksiya;
- dizyunksiya;

1) inkor – A mulohazani inkori deganda A rost bo'lganda yolg'on yoki yolg'on bo'lganda rost qiymat qabul qiluvchi mulohazaga aytiladi. Python tilida inkor – *not A* bilan beriladi. Masalan, A mulohaza inkori *not A* ko'rinishida yoziladi;

2) konyunksiya- ikkita A va B mulohazalar konyunksiyasi yoki mantiqiy ko'paytmasi «A and B» ko'rinishga ega. Bu mulohaza faqat A va B mulohazalar rost bo'lgandagina rost bo'ladi, aks holda yolg'on bo'ladi (odatda «and» amali «va» deb o'qiladi).

3) dizyunksiya – ikkita A va B mulohazalar dizyunksiyasi yoki mantiqiy yig'indisi «A or B» ko'rinishda yoziladi. Bu mulohaza rost bo'lishi uchun A yoki B mulohazalardan biri rost bo'lishi yetarli. Odatda «or» amali «yoki» deb o'qiladi.

Mantiqiy amallarni bajarilish jadvali quyidagicha.

A	B	Not A	Not B	A and B	A or B
0	0	1	1	0	0
0	1	1	0	0	1
1	0	0	1	0	1
1	1	0	0	1	1

Taqqoslash va mantiqiy amallarni ishlash jarayoni tushunarli bo'lishi uchun, ularni interaktiv rejimda sinab ko'ramiz.

Misol. Taqqoslash amallaridan foydalanish

```
Type "help", "copyright", "credits" or "license()" for more information.
>>> a=12
>>> b=-7
>>> a>b
True
>>> z=a<b
>>> z
False
>>> not z
True
>>> z=a==b
>>> z
False
>>> a!=b
True
>>> a>=b
True
>>> a<=b
False
>>>
```

Python dasturlash tilida mantiqiy amallardan foydalanishda albatta quyidagilarga e'tibor bering:

- *O'zgaruvchiga boshlang'ich qiymatlarni berishda = belgisi oldidan va orqasidan bitta bo'sh joy(probel) quyding;*
- *O'zgaruvchiga boshlang'ich qiymatlarni berishda True va False kabi yozilish kerak ya'ni birinchi harfi katta harflarda.*

Misol. Mantiqiy amallardan foydalanish

```
Type "help", "copyright", "credits" or "license()" for more information.
>>> a = True
```

```
>>> b = False
```

```
>>> a,b
```

```
(True, False)
```

```
>>> z=a and b
```

```
>>> z
```

```
False
```

```
>>> z=a or b
```

```
>>> z
```

```
True
```

```
>>> not z
```

```
False
```

Nazorat savollari

- 1.Arifmetik amallarni Python tilida yozilishi?*
- 2.Ta'minlash operatori va uning vazifasi?*
- 3.Amal bilan beriladigan ta'minlash operatorini turlari va ularni Python tilida yozilishi?*
- 4.Taqqoslash amallari va ularni Python tilida yozilishi?*
- 5.Mantiqiy amal?*
- 6.Inkor amali va uning Python tilida yozilishi;*
- 7.Konyunksiya amali va uning Python tilida yozilishi?*
- 8.Dizyunksiya amali va uning Python tilida yozilishi?*

Mustaqil ishlash uchun topshiriqlar.

1.a=rost(1) va b=yolg'on(0) bo'lganda quyidagi mantiqiy ifodalarning qiymatini aniqlang.

1. $(a \& \& b) !a;$	11. $(a b) \& \& (a \& \& b);$
2. $(a b) \& \& b;$	12. $(!a b) \& \& (a \& \& !b);$
3. $(a b) \& \& !b;$	13. $(a b) \& \& (!a \& \& !b);$
4. $!(a b) \& \& b;$	14. $(!a !b) \& \& (a \& \& b);$
5. $(!a b) \& \& b;$	15. $!(a b) \& \& !(a \& \& b);$
6. $(!a !b) \& \& b;$	16. $!(!a !b) \& \& !(a \& \& b);$
7. $(!a !b) \& \& !b;$	17. $!(!(a b) \& \& !(a \& \& b));$
8. $!((a b) \& \& b);$	18. $!(a \& \& b) !(a \& \& b);$
9. $(a \& \& b) !b;$	19. $!(!a !b) \& \& !(a b);$
10. $(!a \& \& !b) !b;$	20. $!(!(a \& \& b) \& \& !(a b));$

2. Uchta A, B, C butun sonlar berilgan. Jumlani rostlikka tekshiring: “ A, B, C sonlardan ikkitasi musbat son”.

3. Musbat butun son berilgan. Jumlani rostlikka tekshiring: “Berilgan son ikki xonali juft son”.

4. Musbat butun son berilgan. Jumlani rostlikka tekshiring: “Berilgan son uch xonali toq son”.

5. Jumlani rostlikka tekshiring: “Berilgan uchta butun sonlarning hech bo‘lmaganda 2 tasi bir biriga teng”.

6. Jumlani rostlikka tekshiring: “Berilgan uchta butun sonlarning hech bo‘lmaganda bir jufti o‘zaro qarama-qarshi”.

7. Uch xonali son berilgan. Jumlani rostlikka tekshiring: “Ushbu sonning barcha raqamlari xar xil”.

1.5 PYTHON DASTURLASH TILI TARKIBIDAGI MATEMATIK FUNKSIYALAR VA IFODALAR

Reja:

1. Python tilida ifodalar;

2. Python dasturlash tilida matematik funksiyalar.

Tayanch so‘zlar: *ifoda, import, math, standart funksiya, min, max, sum.*

Barcha dasturlash tillari kabi *Python* dasturlash tilida ham matematik ifodalar ma’lum bir standartlar asosida yoziladi. Ifodalar tarkibidagi

matematik funksiyalar Python tilida standart funksiyalar yordamida yoziladi, agar ifoda tarkibidagi funksiya standart funksiya tarkibida bo'lmasa, alohida funksiya yaratib olish kerak.

Ifoda-sonlar, harflarni arifmetik amallar va qavslar bilan birlashtirilgan yozuvga aytiladi.

Python dasturlash tilidagi ifodalar tarkibidagi amallarni bajarilishi matematikadagi amallarni bajarilish tartibiga mos keladi. Python tilida arifmetik amallarni yozilishi yuqoridagi mavzuga asosan yoziladi. Ifodalar tarkibidagi nomalumlarni faqatgina lotin alifbosida yozilishi kerak. Ifoda tarkibida kasr sonning surati yoki maxrajida ikki va undan ortiq hadlar bo'lsa, python tilida ular albatta qavsga olinishi kerak.

Python dasturlash tilida matematik funksiyalar

Python dasturlash tili tarkibida mavjud bo'lgan matematik funksiyalar **standart funksiyalar** deb ataladi.

Ifodalar tarkibidagi funksiyalarni Python dasturlash tilida ifodalash uchun standart funksiyalardan foydalaniladi. Funksiyalarni python dasturlash tilida ifodalash uchun ularni argumentlarini albatta qavsga olib yozish kerak.

Python dasturlash tilida matematik funksiyalardan foydalanish uchun albatta python tili tarkibidagi matematik funksiyalar kutubxonasiga murojat qilish kerak. Matematik funksiyalar kutubxonasiga murojat qilish quyidagicha.

from math import*

Python dasturlash tili tarkibidagi matematik funksiyalar yozilishi quyidagi ro'yxat asosida amalga oshiriladi.

No	Python dasturlash tilida ifodalanishi	Matematik ifodalanishi
1	<i>trunc(x)</i>	<i>Sonning butun qismi</i> <i>trunc(5.8)=5, trunc(-5.8)=-5</i>
2	<i>sqrt(x)</i>	<i>x ning kvadrat ildiz</i>
3	<i>log(x), log2(x), log10(x)</i>	<i>ln x, log₂ x, log₁₀ x</i>

4	$\log(x,a)$	$\log_a x$
5	$\sin(x), \cos(x), \tan(x)$	$\text{Sin}x, \text{cos}x, \text{tg}x$ trigonometrik funksiyalar
6	$\text{asin}(x), \text{acos}(x), \text{atan}(x)$	$\text{Arcsin}x, \text{arccos}x, \text{arctg}x$ teskari trigonometrik funksiyalar
7	$\text{atan2}(x,y)$	
8	$\text{degrees}(x)$	Radiandan gradusga o'tkazish funksiyasi
9	$\text{radians}(x)$	Gradusdan radianga o'tkazish funksiyasi
10	$\sinh(x), \cosh(x), \tanh(x)$	Giperbolik $\text{Sin}x, \text{cos}x, \text{tg}x$ trigonometrik funksiyalar
11	$\text{asinh}(x), \text{acosh}(x), \text{atanh}(x)$	Giperbolik $\text{Arcsin}x, \text{arccos}x, \text{arctg}x$ teskari trigonometrik funksiyalar
12	$\text{hypot}(x,y)$	Katetlari x va y bo'lgan to'g'ri burchakli uchburchakning getatinuzasini topish
13	$\text{factorial}(x)$	X faktorialni hisoblash funksiyasi
14	gamma	x ning gamma funksiyasi
15	π	π soni $\pi=3.1415\dots$
16	e	Eksponentsial funksiya $e=2.71\dots$

Matematik funksiyalarning bajarilish jarayoni, matematikada qanday bo'lsa python dasturlash tilida ham xuddi shunday amalga oshiriladi.

Matematik funksiyalarni ishlash jarayoni tushunarli bo'lishi uchun, ularni interaktiv rejimda sinab ko'ramiz. Chunki interaktiv rejim bir vaqtning uzida natija qaytaradi.

Misol. Matematik funksiyalarni bajarilishi

Type "help", "copyright", "credits" or "license()" for more information.

```
>>> from math import*
>>> x=12.7
>>> trunc(x)
12
>>> trunc(12.2)
12
>>> trunc(-12.7)
-12
>>> trunc(-12.2)
-12
>>> sqrt(81)
9.0
>>>log(e),log2(8),log10(x)
(1.0, 3.0, 1.1038037209559568)
>>> log(81,3)
4.0
>>> cos(pi)
-1.0
>>> atan(180)
1.565240828394204
>>> radians(180), degrees(pi/3)
(3.141592653589793, 59.99999999999999)
```

Qo‘shimcha funksiyalar

Python dasturlash tilida standart kutubxona tarkibida ketma ketliklar ustida bir nechta maximum, minimum, summa kabi funksiyalar aniqlangan. Boshqa dasturlash tillarida bu funksiyalar alohida algoritmlar yordamida tuzib olinadi, python dasturlash tilida esa bu funksiyalar tayyor holda saqlanadi. Qo‘shimcha funksiyalar quyidagi jadval ko‘rinishida amalga oshiriladi.

№	Python dasturlash tilida ifodalanishi	Matematik ifodalanishi
1	$\max(a,b,\dots)$	Sonlar yoki kortej ichidan eng kattasini topish. $\max(2,-8)=2$
2	$\min(a,b,\dots)$	Sonlar yoki kortej ichidan eng kichigini topish. $\min(2,-8)=-8$
3	$\text{sum}(a,b,\dots)$	Sonlar yoki kortej yig'indisini topish. $\text{sum}(2,-8)=-6$
4	$\text{sorted}(a,b,\dots)$	Sonlarni tartiblash. $\text{sorted}(3,12,-9)=(-9,3,12)$

Qo'shimcha funksiyalarni ishlash jarayoni tushunarli bo'lishi uchun, ularni interaktiv rejimda sinab ko'ramiz. Chunki interaktiv rejim bir vaqtning uzida natija qaytaradi.

Misol. Qo'shimcha funksiyalarni bajarilishi

```
Type "help", "copyright", "credits" or "license()" for more information.
>>> max(12,5,-8,7)
12
>>> min(12,5,-8,7)
-8
>>> a=(1,2,3,0,-12,123)
>>> a
(1, 2, 3, 0, -12, 123)
>>> max(a), min(a), sum(a)
(123, -12, 117)
>>> sorted(a)
[-12, 0, 1, 2, 3, 123]
>>>
```

Misol. Quyidagi ifodalarni python tilida ifodalash.

Matematik ifodasi

$$y = (x + \sin x)^3 - \cos^2 x + \frac{1 + \log_a x}{\sqrt{t - x^2}}$$

Python tilida ifodalanishi

```
y=pow((x+sin(x)),3)+sqr(cos(x))+(1+log(x,a))/(sqrt(t-x**2));
```

Misol. Quyidagi ifodalarni Python tilida ifodalash.

Matematik ifodasi

$$y = |x - 2| + \sin x - \left| \frac{4}{\sqrt{t - x^2}} \right|$$

Python tilida ifodalanishi

```
y=abs(x-2)+sin(x)-abs(4/sqrt(t-x**2));
```

Nazorat savollari

- 1.Ifoda deb nimaga aytiladi?
- 2.Ifodalarni python tilida yozilish tartibi?
- 3.Standart funksiya deb nimaga aytiladi?
- 4.Standart funksiyalarni python tilida yozilish tartibi?
- 5.Matematik kutubxonaga murojat qilish tartibini tushuntirib bering?
- 6.Matematik funksiyalarning har birini yozib ma'nosini tushuntirib bering?
- 7.Qo'shimcha(min, max, sum) funksiyalarning har birini yozib ma'nosini tushuntirib bering?

Mustaqil ishlash uchun topshiriqlar

Quyidagi ifodalarni python dasturlash tilida yozing

$$y = \frac{1 + x^4}{\sin x + |x|} - \operatorname{tg} x$$

$$y = \frac{ax^2 - bx + c}{|x|} - \left| \frac{x - t}{\sqrt{x + a}} \right|$$

$$y = \frac{2}{|ax^2 - bx + c|} - \left| \frac{x - t}{\sqrt{x + a}} \right|$$

$$y = \frac{\sqrt{x+a}}{\sin x + |x|} - 1 + x^4$$

$$y = \frac{\sqrt{x+a}}{\sin x + |x|} - ax^2 - bx + c$$

$$y = ax^2 - bx + c - \left| \frac{\log_a x}{\sqrt{x+a}} \right|$$

$$y = \frac{\sqrt{x+a}}{\sin x + |x|} - \left| \frac{\log_a x}{\sqrt{x+a}} \right|$$

$$y = \pi - 3x - \frac{\sqrt{x+a}}{\sin x + |x|}$$

$$y = \pi - 3x - \left| \frac{\log_a x}{\sqrt{x+a}} \right|$$

$$y = \pi - ax^2 - bx + c - \frac{\sqrt{x+a}}{\sin x + |x|}$$

II-BOB. PYTHON TILIDA CHIZIQLI, TARMOQLANUVCHI VA TAKRORLANUVCHI JARAYONLARNI DASTURLASH

2.1 PYTHON DASTURLASH TILIDA CHIZIQLI JARAYONLARNI DASTURLASH

Reja:

1. Python tilida chiziqli dasturlar;
2. `type()` va `help()` funksiyalari.

Tayanch soʻzlar: *type, help, operator, chiziqli dastur, oʻzgaruvchi.*

Programmash tilida tuzilgan dasturlar odatda uchta jarayonga asoslanib tuziladi. Dasturlash tili operatorlari yechilayotgan masala algoritmini amalga oshirish uchun ishlatiladi. Operatorlar chiziqli va boshqaruv operatorlariga boʻlinadi. Aksariyat boshqa dasturlash tillarida operatorlar nuqtali vergul (;) belgisi bilan tugallanadi python dasturlash tilida esa ; nuqtali vergulni quyish shart emas. Dastur tuzish vaqtida buyruqlar ketma-ketligi uzluksiz bajarilib boshqa shartlar talab etilmasa, dastur chiziqli hisoblanadi.

Tarif: Chiziqli algoritmlarga asoslanib python dasturlash tilida tuzilgan dasturlar **chiziqli dasturlar** deyiladi.

Chiziqli dasturlar tarkibiy qismi boʻlgan operator va buyruqlarda hech qanday shart yoki takrorlanish bajarilmaydi. Chiziqli dasturlar tarkibidagi boʻyruqlar, albatta, bir marta bajariladi.

Misol: *Quyidagi funksiyani hisoblang $((a+x)>0)$.*

$$y = \frac{ax^3 - \sin x}{1 + \ln x} - \sqrt{a + x}$$

```
a=input('a=')
x=input('x=')
a=float(a)
x=float(x)
from math import*
y=(a*pow(x,3)-sin(x))/(1+log(x))-sqrt(a+x);
print('y=',y)
===== RESTART: C:/Users/User/Desktop/2.py
=====
```

```
a=5
x=8
y= 827.3927343284367
>>>
```

Yuqoridagi masalaga e'tibor bersak a va x o'zgaruvchilar qiymati berilganda y funksiyani natijasi hisoblandi, algoritm dastur tarkibidagi operatorlar ham bir marta bajarilyapti.

Misol: Asosining radiusi r va balandligi h bo'lgan slindr hajmi va to'la sirtini toping.

Bu masala yechimini aniqlash uchun slindr to'la sirti va hajm formulalarini aniqlash kerak. Berilgan r va h yordamida to'la sirti va hajmini aniqlash mumkin.

```
r=input('r=')
h=input('h=')
r=float(r)
h=float(h)
from math import*
v=pi*r**2*h
s=2*pi*r*h+2*pi*r**2
print('v=',v)
print('s=',s)
===== RESTART: C:/Users/User/Desktop/2.py
=====
r=1
h=2
v= 6.283185307179586
s= 18.84955592153876
>>>
```

Python dasturlash tilida chiziqli dasturlar tuzilganda uning tarkibida matematik funksiyalar ishtirok etsa, albatta, matematik funksiyalar paketini chaqirish kerak. Ifodalarni ketma-ket ijro etish strukturasi Python

tomonidan ta'minlanadi. Normal sharoitda python ifodalari dasturdagi bo'yruqlar yozilishiga ko'ra bajariladi.

type() va help() funksiyalari

Dastur bajarilish vaqtida pythonda o'zgaruvchilar turi e'lon qilinmasligi sababli ba'zi o'zgaruvchilar natijasi qanday turda ekanligini aniqlash kerak bo'ladi. Python dasturlash tilida o'zgaruvchilar turini aniqlash uchun type() funksiyasi aniqlangan, type() funksiyasining umumiy ko'rinish quyidagicha.

type(<o'zgaruvchi>)

type(<o'zgaruvchi>) – bu funksiya o'zgaruvchining qanday turda ekanligini aniqlab beradi.

Type funksiyasini ishlash jarayoni tushunarli bo'lishi uchun, ularni interaktiv rejimda sinab ko'ramiz. Chunki interaktiv rejim bir vaqtning uzida natija qaytaradi.

Misol. Type funksiyasini bajarilishi

```
Type "help", "copyright", "credits" or "license()" for more information.
>>> a=12
>>> b=21.536
>>> c='matematika'
>>> type(a)
<class 'int'>
>>> type(b)
<class 'float'>
>>> type(c)
<class 'str'>
>>>
```

Dastur tuzuvchiga dasturlash tili tarkibidagi yordam bo'limi juda katta yordam beradi. Python dasturlash tilida funksiyalar qanday vazifani amalga oshirishni aniqlash uchun help() funksiyasi aniqlangan, help() funksiyasining umumiy ko'rinish quyidagicha.

help(<funksiya>)

help(<funksiya>) – bu funksiya funksiyaning qanday qanday vazifa bajarishini aniqlab beradi.

Help funksiyasini ishlash jarayoni tushunarli bo‘lishi uchun, ularni interaktiv rejimda sinab ko‘ramiz.

Misol. Help funksiyasini bajarilishi

```
Type "help", "copyright", "credits" or "license()" for more information.
```

```
>>> help(max)
```

```
Help on built-in function max in module builtins:
```

```
max(...)
```

```
max(iterable, *[, default=obj, key=func]) -> value
```

```
max(arg1, arg2, *args, *[, key=func]) -> value
```

With a single iterable argument, return its biggest item. The default keyword-only argument specifies an object to return if the provided iterable is empty.

With two or more arguments, return the largest argument.

```
>>>
```

Nazorat savollari

1. Chiziqli dasturlar deb nimaga aytiladi?
2. type() va help() funksiyalarning vazifalari?

Mustaqil ishlash uchun topshiriqlar

1. Nolga teng bo‘lmagan ikkita son berilgan. Ularning yig‘indisi, ko‘paytmasi va har birining moduli aniqlansin.
2. Uzunlik L santimetrda berilgan. Undagi to‘liq metrilar sonini aniqlovchi dastur tuzilsin (1 metr = 100 sm).
3. Og‘irlik M kilogramda berilgan. Undagi to‘liq tonnalar sonini aniqlovchi dastur tuzilsin (1 tonna = 1000 kg).

4. Faylning hajmi baytlarda berilgan. Bo‘lib butunni olish operatsiyasidan foydalanib fayl hajmining to‘liq kilobaytlarda ifodalovchi dastur tuzilsin (1 Kilobayt = 1024 bayt).
5. A va B ($A > B$) musbat sonlar berilgan. A kesmada, B kesmani necha marta joylashtirish mumkinligini aniqlovchi dastur tuzilsin.
6. A va B ($A > B$) musbat sonlar berilgan. A kesmada, B kesmani necha marta joylashtirish mumkin. A kesmada B kesmaning joylashmagan qismini aniqlovchi dastur tuzilsin.
7. Ikki xonali son berilgan. Oldin uning o‘nliklar xonasidagi raqamni, So‘ng birlar xonasidagi raqamni chiqaruvchi dastur tuzilsin.
8. Ikki xonali son berilgan. Uning raqamlar yig‘indisi va ko‘paytmasini aniqlovchi dastur tuzilsin.
9. Ikki xonali son berilgan. Uning raqamlari o‘rnini almashtirishdan hosil bo‘lgan sonni aniqlovchi dastur tuzilsin.
10. Uch xonali son berilgan. Uning yuzlar xonasidagi raqamini aniqlovchi dastur tuzilsin.
11. 999 dan katta bo‘lgan son berilgan. Bir marta bo‘lib butunni va bo‘lib qoldiqni olish operatsiyasidan foydalanib berilgan sonni mingliklar xonasidagi sonni aniqlovchi programma tuzilsin.
12. Kun boshidan boshlab N sekund vaqt o‘tdi. Kun boshidan boshlab qancha minut to‘la o‘tganligini aniqlovchi programma tuzilsin.
13. Hafta kunlari quyidagi tartibda berilgan: 1 – dushanba, 2 – seshanba, ..., 6 – shanba, 7 – yakshanba (N 1-7gacha bo‘lgan hafta kunlari soni). 1-365 oraliqda yotuvchi K soni berilgan. Agar 1-yanvar N chi kunga to‘g‘ri kelsa, kiritilgan K – kun haftaning qaysi kuniga to‘g‘ri kelishini aniqlovchi programma tuzilsin.
14. A , B , C butun sonlar berilgan. Tomonlari $A \times B$ bo‘lgan to‘g‘ri to‘rtburchakka tomoni C bo‘lgan kvadrat eng ko‘p joylashtirilsin. To‘g‘ri to‘rt burchakka eng ko‘p joylashgan kvadratlar soni va joylashmay qolgan qismi yuzasini aniqlovchi programma tuzilsin.
15. Qaysidir yil berilgan. Berilgan yilning qaysi yuz yillikka kirishini aniqlovchi programma tuzilsin. (Masalan: 20 – yuz yillikning boshi 1901 yil).

2.2 PYTHON DASTURLASH TILIDA TARMOQLANUVCHI JARAYONLARNI DASTURLASH

Reja:

1. *Qisqa shartli operator va uning umumiy ko‘rinishi;*
2. *To‘liq shartli operator va uning umumiy ko‘rinishi;*
3. *elif operatori va uning umumiy ko‘rinishi.*

Tayanch so‘zlar. *If, else, elif, qisqa shartli, to‘liq shartli.*

Python dasturlash tilida chiziqli jarayonlar buyruqlar ketma-ketligi asosida bajariladi, tarmoqlanuvchi jarayonlarni dasturlashda esa, buyruqlar ma’lum bir shartlar asosida tarmoqlanish bo‘yicha bajariladi. Python dasturlash tilida tarmoqlanuvchi(shartli) jarayonlarni bir necha turlarga bo‘lingan holda amalga oshirish mumkin.

Qisqa shartli operator va uning umumiy ko‘rinishi

Tarmoqlanuvchi jarayonlarni amalga oshiruvchi operatorlarni qisqacha qilib shartli operatorlar deb yuritimiz. Shartli operatorlar ham qisqa, to‘liq va umumiy shartli operator ko‘rinishlarida, tarmoqlanishni amalga oshiradi.

***if* operatori**

Tarmoqlanuvchi jarayonlarni python dasturlash tilida amalga oshirish uchun, albatta, tarmoqlanuvchi algoritmlar asosida bajariladi. Algoritm bajarilish vaqtida ma’lum bir shartlar asosida algoritmning u yoki bu qismi bajarilishini ta’minlash maqsadida shartli operatorlardan foydalaniladi.

*Tarif: Algoritm tarkibidagi shart asosida algoritmning tarmoqlarga bo‘linishiga xizmat qiluvchi operatorlar **shartli operatorlar** deyiladi.*

Hayotdagi asosiy ko‘p masalalarni dasturlash vaqtida, albatta, tarmoqlanuvchi algoritmlar asosida bajariladi. Tarmoqlanuvchi algoritmlar ham ikki xil holatni o‘z ichiga oladi, ya’ni shart rost bo‘lganda ma’lum bir vazifani yolg‘on bo‘lganda hech qanday vazifani bajarmaydi va shart rost bo‘lganda ma’lum bir vazifani yolg‘on bo‘lganda boshqa vazifani bajaradi. Demak, yuqoridagi ikki holatni e’tiborga olib, shartli operatorlar ham qisqa va to‘liq ko‘rinishga ega.

Qisqa shartli operatorning umumiy ko‘rinishi quyidagicha

if <mantiqiy ifoda> :

operatorlar

if operatori tarkibida shartlar ikki va undan ortiq bo'lsa oldingi bobdagi mantiqiy ifodalar asosida birlashtiriladi. if operatori tarkibidagi shart faqat chin bo'lgandagina : dan keyingi operatorlar bajariladi.

Misol: Berilgan sonning juft yoki toq ekanligini aniqlang.

```
a=10
if a%2==0:
    print('juft')
if a%2!=0:
    print('toq')
===== RESTART: C:\Users\User\Desktop\2.py
=====
juft
>>>
```

Berilgan masala yechimida faqat bitta shart tekshirildi, bu yerda blok yoki begin end vazifalari hech qanday buyruqsiz amalga oshiriladi. Agar shartdan keyin bir nechta operator bajarilish uchun : belgisidan keyin ENTER tugmasi bosilib yozilaveradi yani bir soha bo'yicha.

Misol: Berilgan a va b sonda $a > b$ bo'lsa ikkita sonning yig'idi va ayirmasini, $a \leq b$ bo'lsa ikkita sonning ko'paytmasi va bo'linmasini toping.

```
a=10
b=5
if a<=b:
    y=a+b
    z=a-b
if a>b:
    y=a*b
    z=a/b
print('y=',y,' z=',z)
===== RESTART: C:\Users\User\Desktop\2.py
=====
y= 50 z= 2.0
>>>
```

To'liq shartli operator va uning umumiy ko'rinishi

Python dasturlash tilida tarmoqlanuvchi jarayonlarni to'liq shartli ko'rinishida ifodalash uchun *if else* operatoridan foydalaniladi. Algoritm tarkibidagi shartlar chin qiymat qabul qilganda ma'lum bir operatorlar, yolg'on qiymat qabul qilganda boshqa operatorlar bajarilishi ham mumkin.

Tarmoqlanuvchi jarayonlarni python dasturlash tilida to'liq shartli operatorning umumiy ko'rinishi quyidagicha

```
if <mantiqiy ifoda> :  
    operatorlar1  
else:  
    operatorlar2
```

if else operatori tarkibidagi shartlarning chin qiymat qabul qilganda **operatorlar1** bajariladi, aks holda **operatorlar2** bajariladi.

Misol: Berilgan son juft bo'lsa 2 ga ko'paytirib juft so'zini aks holda toq bo'lsa 3 ga ko'paytirib toq so'zini ekranga chop eting.

```
a=15  
if a%2==0:  
    print('juft')  
    print(2*a)  
else:  
    print('toq')  
    print(3*a)  
===== RESTART: C:\Users\User\Desktop\2.py  
=====  
toq  
45  
>>>
```

Misol. Pythonda quyidagi masalani dasturiy ta'minotini yarating.

$$f(x) = \begin{cases} 0.5 \cdot \cos(x), & \text{agar } x > 0; \\ x+8, & \text{agar } x \leq 0; \end{cases}$$

```
x=float(input('x='))
from math import*
if x>0:
    y=0.5*cos(x)

else:
    y=x+8
print('y=',y)
===== RESTART: C:\Users\User\Desktop\2.py
=====
x=-2.8
y= 5.2
>>>
```

elif operatori

Tarmoqlanuvchi jarayonlarni algoritmlash vaqtida, albatta, shartga e'tibor berish talab etiladi, chunki algoritm shart bo'yicha tarmoqlanib qismlarga ajralib ketadi. Masalan, dastur bajarilishining birorta qadamida qandaydir shartni tekshirish natijasiga ko'ra boshqaruvni dasturning u yoki bu bo'lagiga uzatish mumkin.

elif shart operatori Python dasturlash tilida aniqlangan bo'lib, u aks holda ichida shart kelgan holatlarda ishlatiladi. **Elif shart operatorining** umumiy ko'rinishi quyidagicha.

if <mantiqiy ifoda1> :
operatorlar1

elif <mantiqiy ifoda2> :

operatorlar2

else:

peratorlar3

Bu operator aks holda qismida alohida if operatorini yozmaslik uchun ishlatiladi, agar <mantiqiy ifoda1> false qiymatidan farqli ya'ni true bo'lsa, <operatorlar1> bajariladi, aks holda <mantiqiy ifoda2> tekshiriladi, agar chin bo'lsa <operatorlar2> bajariladi, aks holda <operatorlar3> bajariladi.

Misol. Berilgan musbat sonni kabisa yili ekanligini aniqlang. 4 ga karrali yillar va 100 ga karralilar ichida faqat 400 ga karralilari Kabisa yili hisoblanadi.

Masalan: 16,24,2016,2020, 2400 lar kabisa va 5,100,200, 2017 lar kabisa yili emas.

```
x=float(input('x='))
from math import*
if x%100==0:
    if x%400==0:
        print('kabisa yili')
    else:
        print('kabisa yili emas')
elif(x%4==0):
    print('kabisa yili')
else:
    print('kabisa yili emas')
===== RESTART: C:\Users\User\Desktop\2.py
=====
x=2020
kabisa yili
>>>
```

Tarmoqlanuvchi jarayonlarini algoritmlarida uchraydigan shartlar holatiga qarab, yuqorida keltirib o'tilgan uch xil ko'rinishdagi shartli operatorlarning qulay birortasidan foydalaniladi.

Nazariy savollar.

1 Tarmoqlanuvchi algoritmlar?

2 Tarmoqlanuvchi jarayonlarni dasturlashning necha xil usuli mavjud?

3 To'liq shartli operatorning ta'rifi, uning umumiy ko'rinishi va vazifasi?

4 Qisqa shartli operatorning ta'rifi, uning umumiy ko'rinishi va vazifasi?

5 Umumiy shartli operatorning ta'rifi, uning umumiy ko'rinishi va vazifasi?

Mustaqil ishlash uchun topshiriqlar

1. Uchta butun son berilgan. Shu sonlarning ikkitasi o'zaro teng, qolgan bittasini tartib raqami aniqlansin.

2. To'rtta butun son berilgan. Shu sonlarning uchta o'zaro teng, qolgan bittasini tartib raqami aniqlansin.

3. Sonlar o'qida uchta A , B , C nuqtalar berilgan. A nuqtaga eng yaqin nuqta va ular orasidagi masofa topilsin.

4. Ikkita A va B butun tipli o'zgaruvchilar berilgan. Agar ularning qiymatlari o'zaro teng bo'lmasa, har bir o'zgaruvchiga bu qiymatlar yig'indisi ta'minlansin, agar o'zaro teng bo'lsa, o'zgaruvchilarga 0 ta'minlansin. A va B ning yangi qiymatlari ekranga chiqarilsin.

5. Uchta son berilgan. Ularning o'rtasidagi (ya'ni kattasi va kichigi orasida joylashgan) sonni aniqlovchi dastur tuzilsin.

6. Uchta son berilgan. Shu sonlarning yig'indisi eng katta bo'ladigan ikkitasini ekranga chiqaradigan dastur tuzilsin.

7. Kordinatalar tekisligida butun son berilgan. Agar nuqta koordinata boshida yotsa, 0 chiqarsin. Agar nuqta OX yoki OY o'qlarida joylashsa mos holda 1 va 2 chiqarilsin. Agar nuqta koordinata o'qida joylashmasa 3 chiqarilsin.

8. OX va OY koordinata o'qlarida yotmaydigan nuqta berilgan. Nuqta joylashgan koordinata choragi aniqlansin.

9. Koordinata o'qlariga parallel ravishda to'g'ri to'rtburchakning uchta uchi berilgan, to'rtinchi uchi koordinatasini aniqlansin.

10. X haqiqiy soni berilgan. Quyidagi funksiya hisoblansin.

$$f(x) = \begin{cases} 2 \cdot \sin(x), & \text{agar } x > 0; \\ x - 6, & \text{agar } x \leq 0; \end{cases}$$

11. X haqiqiy soni berilgan. Quyidagi funksiya hisoblansin.

$$f(x) = \begin{cases} 2 \cdot x, & \text{agar } x < -2 \text{ yoki } x > 2; \\ -3 \cdot x, & \text{aks holda;} \end{cases}$$

12. X haqiqiy soni berilgan. Quyidagi funksiya hisoblansin.

$$f(x) = \begin{cases} -x, & \text{agar } x \leq 0; \\ x^2, & \text{agar } 0 < x < 2; \\ 4, & \text{agar } x \geq 2; \end{cases}$$

13. X haqiqiy soni berilgan. Quyidagi funksiya hisoblansin.

$$f(x) = \begin{cases} 0, & \text{agar } x < 0; \\ 1, & \text{agar } x \in [0,1), [2,3), \dots; \\ -1, & \text{agar } x \in [1,2), [3,4), \dots; \end{cases}$$

14. Yil berilgan (musbat butun son). Berilgan yilda nechta kun borligini aniqlovchi dastur tuzilsin. Kabisa yilida 366 kun bor, kabisa bo'lmagan yilda 365 kun bor. Kabisa yil deb 4 ga karralilariga aytiladi. Lekin 100 ga karrali yillar ichida faqat 400 ga karrali bo'lganlari kabisa yil hisoblanadi. Masalan 300, 1300 va 1900 kabisa yili emas. 1200 va 2000 kabisa yili.

15. Butun son berilgan. Berilgan sonni "musbat toq son", "manfiy juft son", "son nolga teng" va h.k ekranga yozadigan dastur tuzilsin.

16. 1-999 oralig'idagi sonlar berilgan. Berilgan sonni "ikki xonali juft son", "uch xonali toq son" va h.k ekranga yozadigan dastur tuzilsin.

2.3 PYTHON DASTURLASH TILIDA TAKRORLANUVCHI JARAYONLAR VA PARAMETR BO'YICHA DASTURLASH

Reja:

1. Takrorlanuvchi jarayonlarni dasturlash;
2. `for(sikl)` operatori va uning umumiy ko'rinishi;

3. Ichma ich sikllarni tashkil qilish

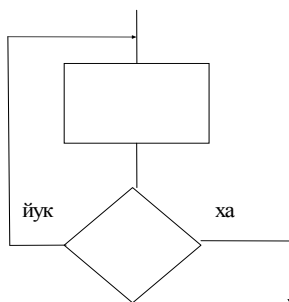
Tayanch soʻzlar. *For, sikl, takrorlanish, parametrli takrorlanish, ichma ich.*

Dastur tuzish jarayonida baʼzi bir masalalarni algoritmlari tarkibidagi buyruqlar ikki va undan ortiq marta bajarilishiga toʻgʻri keladi. Agar algoritm tarkibidagi bir necha marta takrorlanishi kerak boʻlgan buyruqlarni takrorlanuvchi jarayonlar asosida dasturlash tillarida tasvirlanmasa, bu buyruqlarni barchasini bajarish murakkablashadi. Elektron hisoblash mashinalarini insoniyatdan farqi shundaki, insoniyatda bir nechta buyruqlarni bajarish davomida toliqish holatlari boʻlishi mumkin, elektron hisoblash mashinalariga takrorlanishni qanchaligini maʼlum bir buyruqlar asosida berilsa, ular barchasini charchamasdan bajaradi. Baʼzi bir takrorlanuvchi jarayonlarni, takrorlanish formulasini chiqarib oddiy hisoblash mumkin, lekin ixtiyoriy ketma ketliklar yigʻindisini hisoblash oddiy usullar bilan hal etilmaydi, bunday holatlarda takrorlanuvchi jarayonlardan foydalaniladi.

*Tarif: Algoritmning qandaydir qismidagi buyruqlar ikki va undan ortiq bajarilishiga **takrorlanuvchi jarayonlar** deyiladi.*

Yuqoridagi taʼrifga etibor qaratsak, demak algoritmning qandaydir qismi ikki va undan ortiq bajarilishi mumkin boʻlgan holatlar ham mavjud. Bunda dasturchiga shunday vazifa qoʻyiladiki takrorlanish holatini bir yaxlit buyruq asosida kompyuterga qulay usulda berish kerak.

Takrorlanuvchi jarayonlarni quyidagi blok sxema koʻrinishda ixtiyoriy dasturlash tilida tasvirlash mumkin.



Yuqoridagi blok sxema shaklida shart toki chin boʻlgunga qadar takrorlanish bajarilaveradi, aks holda takrorlanish toʻxtatiladi.

Python dasturlash tillarida takrorlanuvchi jarayonlarni quyidagi usullar yordamida tasvirlash mumkin.

- Parametr bo'yicha takrorlash(for);
- Shart bo'yicha takrorlash(while, do while).

for(sikl) operatori

Takrorlanuvchi jarayonlarni takrorlanish soni aniq bo'lgan holatlardagina parameter bo'yicha takrorlash usulidan foydalaniladi. Takrorlanuvchi jarayonlarni parametr bo'yicha Python dasturlash tilida tasvirlash uchun, albatta, takrorlanish soniga e'tibor berish kerak. Parametr bo'yicha takrorlanuvchi jarayonlarga, masalan, birdan n gacha sonlarning kvadratlarini yig'indisini topish kabi misollar kiradi. Bunda takrorlanish soni aniq, ya'ni birdan dan n gacha deb berilyapti.

Parametr bo'yicha takrorlanuvchi jarayonlar takrorlanish oshishi yoki kamayishiga qarab ikki turga bo'linadi:

- noldan boshlab qadam 1 ga teng bo'lgan takrorlanish(1-tur);
- a dan boshlab qadam 1 ga teng bo'lgan takrorlanish(2-tur);
- a dan boshlab b gacha qadam x ga teng bo'lgan takrorlanish(3-tur).

Takrorlanuvchi jarayonlarni dasturlash vaqtida takrorlanish qadami birga oshib borilsa, birinchi turdan foydalaniladi. 1-tur takrorlanish qadami noldan boshlanib birga oshib boruvchi parameter bo'yicha sikl operatorining umumiy ko'rinishi quyidagicha.

for <o'zgaruvchi1> in range(<o'zgaruvchi2>) :
operatorlar

1-tur for operatori takrorlanish sonining boshlang'ich va oxirgi qiymatlari aniq bo'lgandagina ishlatiladi. For operatorining ishlash prinsipi takrorlanish $\langle \mathbf{o'zgaruvchi1} \rangle = \mathbf{0}$ dan boshlanib toki $\langle \mathbf{o'zgaruvchi1} \rangle = \langle \mathbf{o'zgaruvchi2} - \mathbf{1} \rangle$ ga teng bo'lguncha davom etadi, bunda har bir qadamda $\langle \mathbf{o'zgaruvchi1} \rangle$ ni qiymati birga oshib boradi. Bu yerda range() funksiyasi takrorlanish qadami va oxirini ta'minlab berishga xizmat qiladi.

Misol. 1 dan n gacha sonlarning kublari yig'indisini hisoblash dasturini tuzing.

```

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

===== RESTART: C:\Users\User\Desktop\1.py
=====
n=100
s= 25502500
>>>

```

Misol: Quyidagi yig‘indini hisoblash dasturini tuzing.

$$s = \frac{\cos 2}{1} + \frac{\cos^2 2}{2} + \frac{\cos^3 2}{6} + \dots + \frac{\cos^n 2}{n!}$$

Bu masalani hisoblash jarayonida sikl operatori tarkibida yigindini suratini hisoblash uchun alohida funksiya, maxrajini hisoblash uchun alohida funksiya va yig‘indi uchun alohida funksiya yaratib olsak maqsadga muvofiq bo‘ladi.

```

n=input('n=')
n=int(n)
p=1
s=0
t=1
from math import*
for i in range(n):
    p*=(i+1)
    t*=cos(2)
    s+=t/p
print('s=',s)

```

```

===== RESTART: C:\Users\User\Desktop\1.py
=====
n=15
s= -0.34041658757742116
>>>

```

Takrorlanuvchi jarayonlarni dasturlash vaqtida takrorlanish qadami a dan boshlanib qadam 1 bilan b gacha bajarilish ham mumkin, bunda, ikkinchi turdan foydalaniladi. 2-tur takrorlanish qadami a dan boshlanib qadam 1 ga oshib boruvchi parametr bo'yicha sikl operatorining umumiy ko'rinishi quyidagicha.

for <o'zgaruvchi1> **in range**(<o'zgaruvchi2>, <o'zgaruvchi3>) :
operatorlar

Bunda for operatorining boshlang'ich qiymati <o'zgaruvchi2> dan boshlanadi. For operatorining ishlash prinsipi <o'zgaruvchi1> = <o'zgaruvchi2> dan takrorlanish boshlanib toki <o'zgaruvchi1>=<o'zgaruvchi3 - 1> gacha davom etadi, <o'zgaruvchi1>ni qiymati birga oshib borishi ta'minlaydi.

Misol. a dan b gacha sonlar orasida toqlarining kvadratlar, juftlarini kublar yig'indisini toping.

```

a=input('a=')
a=int(a)
b=input('b=')
b=int(b)
p=0 # toqlar yig'indisi
s=0 # juftlar yig'indisi

from math import*
for i in range(a,b+1):
    if i%2==0:
        s+=i**3

```

```
else:
```

```
    p+=i**2
```

```
print('Juftlar kublar yigindisi=',s)
```

```
print('Toqlar kvadratlar yigindisi=',p)
```

```
===== RESTART: C:\Users\User\Desktop\1.py
```

```
=====
```

```
a=2
```

```
b=5
```

```
Juftlar kublar yigindisi= 72
```

```
Toqlar kvadratlar yigindisi= 34
```

```
>>>
```

Takrorlanishni uchinchi turi a dan boshlab b gacha qadam x ga teng bo'lgan takrorlanish jarayonlarini dasturlashni qarab o'tamiz. Bunda takrorlanish ixtiyoriy intervalda ixtiyoriy qadam bilan amalga oshirilish mumkin. C++, Pascal tillarida -1 qadam alohida yozilar edi, python tilida esa manfiy va musbat qadamlar uchinchi tur bo'yicha hal etiladi.

Takrorlanish a dan boshlab b gacha qadam x ga teng bo'lgan parametr bo'yicha sikl operatorining umumiy ko'rinishi quyidagicha.

for <o'zgaruvchi1> **in range**(<o'zgaruvchi2>,

<o'zgaruvchi3>,<o'zgaruvchi4>):

operatorlar

Bunda for operatorining boshlang'ich qiymati <o'zgaruvchi2> dan boshlanadi. Yuqoridagi operatorning ishlash prinsipi <o'zgaruvchi1> = <o'zgaruvchi2> dan takrorlanish boshlanib toki <o'zgaruvchi1>=<o'zgaruvchi3 - 1> gacha davom etadi, <o'zgaruvchi1>ni qiymati har takrorlanishda <o'zgaruvchi4>ga oshib borishi ta'minlanadi.

Misol. a dan b gacha juft sonlarni to'rtinchi darajalar yig'indisini hisoblash dasturini tuzing.

```
a=input('a=')
a=int(a)
b=input('b=')
b=int(b)
if a%2!=0:
    a+=1
s=0
from math import*
for i in range(a,b+1,2):
    s+=i**4

print('s=',s)

===== RESTART: C:\Users\User\Desktop\1.py
=====
a=1
b=5
s= 272
>>>
```

Yuqoridagi misolda a sonini toq ekanligi aniqlanadi, for tarkibida `range()` funksiyasi $b-1$ gacha bajarilishi sababli, biz $b+1$ ni yozamiz. For operatori takrorlanishni a dan b gacha 2 qadam bilan amalga oshiradi.

Uchinchi tur bo'yicha for sikl operatori yordamida katta sondan kichik songacha manfiy qadam bilan ham takrorlanishni amalga oshirish mumkin.

Misol. a dan b ($a > b$) gacha sonlarni ikkiga ko'paytirib ekranga chiqarish dasturini tuzing.

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

```
b=input('b=')
b=int(b)
for i in range(a,b-1,-1):
    print(2*i)

===== RESTART: C:\Users\User\Desktop\1.py
=====

a=1
b=-10
2
0
-2
-4
-6
-8
-10
-12
-14
-16
-18
-20
>>>
```

***Misol.** Sonning natural bo‘luvchilar soni va natural bo‘luvchilar yigindisini hisoblash dasturi tuzilsin.*

Dastur algoritmi berilgan sonni o‘zigacha natural sonlarga bo‘lib chiqiladi qaysi songa qoldiqsiz bo‘linsa usha son olinadi va oxirida natija olinadi.

```

a=input('a=')
a=int(a)
p=0
s=0
for i in range(a):
    if a%(i+1)==0:
        p+=1
        s+=i+1
print('bo"luvchilar soni=',p)
print('bo"luvchilar yig"indisi=',s)
===== RESTART: C:\Users\User\Desktop\1.py
=====
a=6
bo"luvchilar soni= 4
bo"luvchilar yig"indisi= 12
>>>

```

Yuqoridagi ta’rif va misollarga asoslanib, parametrli takrorlanishni (e’tibor bering faqat parametrli takrorlanishni) har qanday turini for operatori orqali amalga oshirish mumkin.

Ichma ich sikllarni tashkil qilish

Har qanday dasturlash tillarida parametrli sikl operatori yoritilgan vaqtda, albatta ichma ich sikllarni tashkil qilish keltirib o‘tiladi. Python dasturlash tilida ham ichma ich sikllarni tashkil qilish imkoniyati mavjud. Ichma ich sikllarni for sikl operatori orqali hosil qilinadi. Ichma ich sikllar – bu takrorlanish ichida yana takrorlanish hosil qilinadi, takrorlanishning har bir qadamida yana n marta takrorlanish bajariladi. Ichma ich sikllarni tashkil etish jarayonini misollar yordamida tushuntirib o‘tamiz.

Misol. $\sum_{i=1}^n \prod_{j=1}^n (\sin(j) + i^2)$ ni hisoblash dasturini tuzing.


```

n=input('n=')
n=int(n)
s=0
from math import*
for i in range(1,n+1):
    p=1
    for j in range(1,n+1):
        p*=(sin(j)+i**2)
    s+=p
print('s=',s)
===== RESTART: C:\Users\User\Desktop\1.py
=====
n=4
s= 77735.97341482465
>>>

```

Misol. $y = -x + \frac{x^2}{1*2} - \frac{x^3}{1*2*3} + \dots + (-1)^n \frac{x^n}{n!}$ ni hisoblash dasturini tuzing.

Bu misolni ikki usulda hisoblash dasturini tuzish mumkin, birinchisi ichma ich sikl asosida, ikkinchisi esa bitta sikl orqali, biz ikkala holatni ham keltirib o‘tamiz.

Ichma ich sikl orqali dasturini tuzish birinchi o‘rganuvchilar uchun bo‘ladi, lekin bu eng yomon usul hisoblanadi va u quyidagicha:

```

x=input('x=')
x=int(x)
n=input('n=')
n=int(n)
s=0
p=1 # suratdagi darajalarni aniqlaydi
t=-1 # ishorani aniqlaydi
q=1 #maxrajdagi darajalarni aniqlaydi

```

```

from math import*
for i in range(1,n+1):
    p=1
    q=1
    for j in range(1,i+1):
        p*=x
        q*=j
    s+=t*p/q
    t*=-1
print('s=',s)
===== RESTART: C:\Users\User\Desktop\1.py
=====
x=5
n=6
s= 8.368055555555557
>>>

```

Bitta sikl orqali bajarish jarayonini qarab o‘tamiz, bunda surat va maxrajdagi ko‘paytuvchilarni sikl ichiga olib yozamiz. $(-1)^n$ darajani hisoblashni eng maqbul usuli bu bitta o‘zgaruvchi orqali hisoblashdir, bunda sikl ichiga shu o‘zgaruvchiga -1 ni ko‘paytirish orqali bajaramiz.

```

x=input('x=')
x=int(x)
n=input('n=')
n=int(n)
s=0
p=1 # suratdagi darajalarni aniqlaydi

```

```

t=-1 # ishorani aniqlaydi
q=1 #maxrajdagi darajalarni aniqlaydi
from math import*
for i in range(1,n+1):
    p*=x
    q*=i
    s+=t*p/q
    t*=-1
print('s=',s)
===== RESTART: C:\Users\User\Desktop\1.py
=====
x=5
n=6
s= 8.368055555555557
>>>

```

Takrorlanuvchi jarayonlarni parametr bo'yicha tashkil qilishda for operatori ustuvorlikni ta'minlaydi va u bir nechta shakllarga bo'lingan holda barcha holatlarga javob beradi.

Nazariy savollar

- 1 Takrorlanuvchi jarayonlar deb nimaga aytiladi?
- 2 Takrorlanuvchi jarayonlarni blok sxema ko'rinishida tasvirlanishi?
- 3 Takrorlanuvchi jarayonlarni dasturlash tilida tasvirlanish usullari?
- 4 For sikl operatori va uning vazifasi?
- 5 For sikl operatorining turlari?
- 6 For sikl operatorining umumiy ko'rinishi?
- 7 1 dan n gacha siklni tashkil qilishni tushuntirib bering?
- 8 a dan b gacha siklni tashkil qilishni tushuntirib bering?
- 9 a dan b gacha x qadamli siklni tashkil qilishni tushuntirib bering?
- 10 Ichma ich sikllarni tashkil qilishni tushuntirib bering?

Mustaqil ishlash uchun savollar

1. A va B butun sonlari berilgan ($A < B$). A dan B gacha bo'lgan barcha butun sonlar (A va B ham kiradi) kvadratlarning yig'indisini chiqaruvchi dastur tuzilsin.

2. N ($N > 0$) butun son berilgan. Quyidagi yig'indini hisoblovchi dastur tuzilsin:

$$S = 1 + 1/2 + 1/3 + \dots + 1/N.$$

3. N ($N > 0$) butun son berilgan. Quyidagi yig'indini hisoblovchi dastur tuzilsin:

$$S = N^2 + (N+1)^2 + (N+2)^2 + \dots + (2N)^2.$$

4. N ($N > 0$) butun son berilgan. Quyidagi ko'paytmani hisoblovchi dastur tuzilsin:

$$P = 1.1 * 1.2 * 1.3 * \dots (N \text{ ta ko'paytuvchi}).$$

5. N ($N > 0$) butun son berilgan. Quyidagi ifodani hisoblovchi dastur tuzilsin: $S = 1.1 - 1.2 + 1.3 - \dots$

(N ta qo'shiluvchi ishoralar almashib keladi. Shartli operatoridan foydalanmang).

6. N ($N > 0$) butun son berilgan. Shu sonning kvadratini quyidagi mula asosida hisoblovchi dastur tuzilsin:

$$N^2 = 1 + 3 + 5 + \dots + (2N-1).$$

har bir qo'shiluvchidan keyin natijani ekranga chiqarib boring. Natijada ekranda 1 dan N gacha bo'lgan sonlarning kvadratlari chiqariladi.

7. A xaqiqiy va N ($N > 0$) butun sonlari berilgan. A ning N - darajasini aniqlovchi dastur tuzilsin: $A^N = A * A * \dots * A$

8. A xaqiqiy va N ($N > 0$) butun sonlari berilgan. Bitta ssikldan foydalanib A ning 1 dan N gacha bo'lgan barcha darajasini chiqaruvchi dastur tuzilsin.

9. A xaqiqiy va N ($N > 0$) butun sonlari berilgan. Bitta ssikldan foydalanib quyidagi A ning 1 dan N gacha bo'lgan barcha darajalarini chiqaruvchi va yig'indini hisoblash dasturi tuzilsin:

$$S = 1 + A + A^2 + \dots + A^N.$$

10. X xaqiqiy va N ($N > 0$) butun sonlari berilgan. Bitta ssikldan foydalanib quyidagi X ning 1 dan N gacha bo'lgan barcha darajalarini chiqaruvchi va yig'indini hisoblovchi dastur tuzilsin:

$$S=1-X+X^2-X^3+\dots+(-1)^N X^N.$$

Shartli operator dan foydalanmang.

11. N ($N>0$) butun soni berilgan. Birdan N gacha bo'lgan natural sonlari ko'paytmasini chiqaruvchi dastur tuzilsin: $N!=1*2*\dots*N$.

1 dan N gacha bo'lgan natural sonlari ko'paytmasi N faktorial deyiladi.

12. N ($N>0$) butun sonlari berilgan. Bitta ssikldan foydalanib quyidagi yig'indini hisoblash dasturi tuzilsin:

$$S=1!+2!+3!+\dots+N!$$

($N!$ ifoda - N faktorial - 1 dan N gacha bo'lgan butun sonlari ko'paytmasini bildiradi: $N!=1*2*\dots*N$).

13. N ($N>0$) butun sonlari berilgan. Bitta ssikldan foydalanib quyidagi yig'indini hisoblash dasturi tuzilsin:

$$S=1+1/1!+1/2!+1/3!+\dots+1/N!$$

($N!$ ifoda - N faktorial - 1 dan N gacha bo'lgan butun sonlari ko'paytmasini bildiradi: $N!=1*2*\dots*N$). Olingan natija $e=\exp(1)$ o'zgarmasning taqribiy qiymatiga teng bo'ladi.

14. X xaqiqiy va N ($N>0$) butun sonlari berilgan. Quyidagi ifoda qiymatini hisoblash dasturi tuzilsin:

$$1+X/1!+X^2/2!+X^3/3!+\dots+X^N/N! \quad (N!=1*2*\dots*N).$$

Olingan natija $e^X=\exp(X)$ ga taqribiy qiymatiga teng bo'ladi.

15. X xaqiqiy va N ($N>0$) butun sonlari berilgan. Quyidagi ifoda qiymatini hisoblash dasturi tuzilsin:

$$X-X^3/3!+X^5/5!-\dots+(-1)^N X^{2N+1}/((2N+1)!)$$

($N!=1*2*\dots*N$). Olingan natija $\sin(X)$ ga taqribiy qiymatiga teng bo'ladi.

16. X xaqiqiy va N ($N>0$) butun sonlari berilgan. Quyidagi ifoda qiymatini hisoblash dasturi tuzilsin:

$$1-X^2/2!+X^4/4!-\dots+(-1)^N X^{2N}/(2N)!$$

($N!=1*2*\dots*N$). Olingan natija $\cos(X)$ ga taqribiy qiymatiga teng bo'ladi.

17. X ($|X|<1$) xaqiqiy va N ($N>0$) butun sonlari berilgan. Quyidagi ifoda qiymatini hisoblash dasturi tuzilsin:

$$X-X^2/2+X^3/3-\dots+(-1)^{N-1} X^N/N.$$

Olingan natija $\ln(1+X)$ ga taqribiy qiymatiga teng bo'ladi.

18. N butun soni va sonlar o'qida 2 ta A , B nuqta berilgan. (A va B haqiqiy son). $[A,B]$ kesmani teng N ta kesmaga bo'ling. $[A,B]$ kesmada

ajratilgan barcha nuqtalar uchun $F(X) = 1 - \sin(X)$ funksiya qiymatini hisoblang. 27. X ($|X| < 1$) haqiqiy va N ($N > 0$) butun sonlari berilgan.

Quyidagi ifoda qiymatini hisoblash dasturi tuzilsin:

$$X + 1 * X^3 / (2 * 3) + 1 * 3 * X^5 / (2 * 4 * 5) + \dots + 1 * 3 * \dots * (2N - 1) X^{2n+1} / (2 * 4 * \dots * (2N) * (2N + 1)).$$

19. N butun soni va sonlar o'qida 2 ta A , B nuqta berilgan. (A va B haqiqiy son). $[A, B]$ kesmani teng N ta kesmaga bo'ling. $[A, B]$ kesmada ajratilgan barcha nuqtalarni chiqaring.

2.4 SIKL QADAMLARINI TASHLAB O'TISH VA SIKLLARNI MUDDATIDAN OLDIN TUGATISH

Reja:

1. Break operatori va uning umumiy ko'rinishi;
2. Continue operatori va uning umumiy ko'rinishi;

Tayanch so'zlar. Break operatori, continue operatori.

Python dasturlash tilida takrorlanuvchi jarayonlar qadamlarini tashlab o'tish va takrorlanuvchi jarayonlarni muddatidan oldin tugatish imkoniyati ham mavjud. Python dasturlash tilida bunday imkoniyatlarni break va continue operatorlari amalga oshiradi. Break va continue operatorlarini ishlash prinsiplari Python va C++ tillarida bir xildir.

Break operatori va uning umumiy ko'rinishi

Dasturlash tillarida algoritm bajarilayotgan vaqtda ma'lum bir sabablarga ko'ra tusatdan algoritm tarkibidagi takrorlanish o'z ish faoliyatini to'xtatish kerak bo'lib qoladi. Bunday holatlarda **break** operatoridan foydalaniladi.

Break operatori ko'p holatlarda takrorlanish jarayonlarida ishlatiladi. Break operatori vazifasi o'zi turgan takrorlanish ish faoliyatini to'xtatishdan iborat, agar break operatori dastur bosh tanasida joylashgan bo'lsa dastur xatolik beradi break faqat takrorlanish tanasida bo'ladi.

Python dasturlash tilida asosan takrorlanish jarayonida takrorlanishlar soni uning tarkibidagi ifodaga bog'liq bo'lib qoladi

shunday vaziyatlarda takrorlanishni to‘xtatish uchun break operatoridan foydalanish maqsadga muvofiq.

Misol: 1 dan n gacha sonlar tarkibidan, birinchi x ga karrali songacha bo‘lgan sonlarni ikkiga ko‘paytirib ekranga chiqaring.

Dastur tuzishda birinchi x ga karrali son chiqqanda dastur tusatdan to‘xtash kerak bo‘ladi, bu jarayonni break amalga oshiradi.

```
n=input('n=')
n=int(n)
x=input('x=')
x=int(x)
for i in range(1,n+1):
    if i%x!=0:
        print(i*2)
    else:
        break;
===== RESTART: C:\Users\User\Desktop\1.py
=====
n=10
x=5
2
4
6
8
>>>
```

Yuqoridagi dastur bajarilishi davrida break operatorigacha bo‘lgan operatorlar bajariladi qolganlari esa bajarilmasdan dastur takrorlanishdan

chiqib ketadi. Chunki break operatori takrorlanish ichida joylashgan, shuning uchun dastur natijasi $1*2=2$ dan boshlab $4*2=8$ gacha bajariladi.

Misol: n dan m gacha sonlar tarkibidan, birinchi x ga karrali songacha bo'lgan sonlar yig'indisi va ko'paytmasini ekranga chiqaring.

```
n=input('n=')
n=int(n)
m=input('m=')
m=int(m)
x=input('x=')
x=int(x)
s=0
p=1
for i in range(n,m+1):
    if i%x!=0:
        s+=i
        p*=i
    else:
        break;
print('s=',s)
print('p=',p)
===== RESTART: C:\Users\User\Desktop\1.py
=====
n=5
m=10
x=4
s= 18
p= 210
>>>
```

Yuqoridagi dastur bajarilish davrida takrorlanish operatori faqat $i=8$ gacha bajariladi, lekin 8 hisobga olinmaydi chunki 8 to'rtga karrali bu vaqtda break ishlaydi, chunki break operatori bajarilgandan so'ng takrorlanishdan chiqib ketadi.

Continue operatori

Dasturlash tillarida ba'zi bir holatlarda dastur tarkibidagi buyruqlar faqatgina ma'lum bir qadamlarda tashlab keyingisi bajarilish kerak bo'lgan holatlar ham mavjud. Bunday holatlarda break operatoridan foydalana olmaymiz, chunki break o'zi turgan takrorlanishdan chiqib ketadi. Dastur tanasida bitta buyruq bajarilmasdan keyingisiga o'tish uchun **continue** operatoridan foydalaniladi.

Continue operatori ham break operatoriga o'xshaydi, lekin bunda faqat bitta bo'yruqni cheklab o'tadi. Asosan takrorlanish jarayonlarida ma'lum bir holatlarda takrorlanishning ba'zi bir holatlari hisoblanmaslik kerak bo'lganda continue operatoridan foydalanish maqsadga muvofiq bo'ladi.

Takrorlanish jarayonida takrorlanishning biror bir qadamini tashlab ikkinchisiga o'tish uchun continue bo'yrugidan foydalaniladi.

***Misol:** 1 dan n gacha sonlar ichida 3 ga karrali bo'lmaganlarini ikkiga ko'paytirib ekranga chiqaring.*

```
n=input('n=')
n=int(n)

for i in range(1,n+1):
    if i%3!=0:
        print(2*i)
    else:
        continue

===== RESTART: C:\Users\User\Desktop\1.py
=====

n=10
2
4
```

```
8
10
14
16
20
>>>
```

Yuqoridag dasturda for operatori 3 ga karrali bo‘lgan sonlarni hisobga olmasdan keyingisiga o‘tib ketadi. Bu esa continue operatori imkoniyatini amalga oshiradi.

Nazariy savollar

- 1 Takrorlanishni muddatidan oldin tugatish deganda nimani tushunasiz?*
- 2 break operatori va uning ishlash jarayoni?*
- 3 Takrorlanish qadamlarini tashlab o‘tish deganda nimani tushunasiz?*
- 4 continue operatori va uning ishlash jarayoni?*

Mustaqil ishlash uchun topshiriqlar

- 1.[a,b] oraliqdagi toq sonlar ichidan 5 ga karrali bo‘lmaganlarini yig‘indisi va ko‘paytmasini toping.*
- 2.[a,b] oraliqdagi juft sonlar ichidan x ga karrali bo‘lmaganlarini bo‘luvchilar yig‘indisi va ko‘paytmasini toping.*
- 3.[a,b] oraliqdagi sonlar ichidan x ga karrali bo‘lganlarini bo‘luvchilar yig‘indisi va ko‘paytmasini toping.*

2.5 PYTHON DASTURLASH TILIDA SHARTLI TAKRORLANUVCHI JARAYONLAR DASTURLASH

Reja:

- 1. Shartli sikl operatori;*
- 2. while operatori va uning umumiy ko‘rinishi.*

Tayanch so‘zlar. *Shartli sikl, shart, while, shartli takrorlanish operatori.*

Shartli sikl operatori

Bazi bir masalalarni yechish algoritmlari tarkibida takrorlanishlar qandaydir shartlarga asosan bajariladi. Har bir takrorlanish jarayoni bajarilish qadamida shart tekshirilib o'tib borilaveradi, qachonki shart yolg'on bo'lgandagina takrorlanish jarayoni to'xtatiladi. Masalan yig'indisi S ga teng bo'lgan natural sonlar sonini topish yoki umumiy hadi n dan kichik bo'lgan cheksiz kamayuvchi geometrik progressiyani hadlar sonini topish kabi masalalarda shartli takrorlanuvchi operatorlardan foydalaniladi. Agar algoritm tarkibidagi bir necha marta takrorlanishi kerak bo'lgan buyruqlarni takrorlanuvchi jarayonlar asosida dasturlash tillarida tasvirlanmasa, bu buyruqlarni barchasini bajarish murakkablashadi.

Shartli takrorlanuvchi algoritmlarni shartli takrorlanuvchi jarayonlar ham deb ataymiz.

Tarif: Agar takrorlanishlar soni ma'lum bir shartlar asosida aniqlansa, bunday jarayonlar shartli takrorlanuvchi jarayonlar deyiladi.

Shartsiz o'tish operatori va tarmoqlanuvchi operatorlar yordamida ham shartli takrorlanuvchi jarayonlarni dasturlash imkoniyati mavjud. Lekin bunday holatlarda bitta amalni bajarish uchun bir nechta operatorlarni ishlatish kerak bo'ladi. Shartli takrorlanuvchi operatorlar bajarilish holatlariga qarab turlarga ajratiladi. Shart asosida takrorlanuvchi jarayonlarni python dasturlash tilida shartli sikl operator yordamida amalga oshiriladi.

While operatori

Shart oldi takrorlanuvchi jarayonlar bajarilish holati har bir takrorlanish oldidan shart tekshirilib keyin takrorlanish tanasidagi operatorlar bajariladi. Agar takrorlanish holati boshidan shart yolg'on qiymat qabul qilsa, takrorlanish bir marta ham bajarilmaydi.

Shart oldi takrorlanuvchi operatorlarning python dasturlash tilida ifodalash uchun while operatori yordamida tasvirlanadi.

Takrorlash strukturasi bir ifoda yoki operatorlarni ma'lum bir shart to'g'ri (true) bo'lishi davomida qaytarish imkonini beradi. Qaytarilayotgan ifoda shartga ta'sir ko'rsatishi kerak. Ma'lum bir vaqt o'tgandan keyin shart false ga o'zgartilishi kerak. Bo'lmasa while (davomida) ish jarayoni tugatilmaydi va cheksiz bajarilib qoladi, bu esa mumkin emas. While faqat

o'zidan keyin kelgan ifodaga ta'sir qiladi. Agar biz bir guruh amallarni qaytarmoqchi bo'lsak, : dan keyin enter bilan operatorlarni yozishimiz kerak. Shart takrorlanishning boshida tekshirilganligi sababli, agar shart noto'g'ri bo'lib chiqsa, takrorlanish bajarilmasligi ham mumkin.

Ta'rif: Agar shartli takrorlanuvchi jarayonlar tarkibidagi shart takrorlanishdan oldin tekshirilsa, shart oldi takrorlanuvchi jarayonlar deyiladi.

Takrorlanuvchi operator tarkibiga beriladigan shart tahlil qilinib yozilish kerak, chunki shart hech qachon yolg'on qiymat qabul qilmasa, dastur cheksiz ishlashga to'g'ri keladi. Takrorlanish hech qachon cheksiz bo'lishi mumkin emas, aks holda algoritmnining diskretlik hossasi buziladi.

Shart oldi takrorlanish operatori yani while operatorining umumiy ko'rinishi quyidagicha.

while <shart>: operatorlar

Agar shart chin qiymat qabul qilib tursa operatorlar bajarilaveradi, qachonki shart yolg'on bo'lgandagina takrorlanish o'z ish faoliyatini to'xtatadi.

While operatori tarkibidagi shart yolg'on qiymat qabul qilganda operatorlar bajarilmasdan qoladi, agar shart chin qiymat qabul qilganda operatorlar bajariladi. Ba'zi hollarda shart takrorlanish boshidan yolg'on qiymat qabul qiladi, bunda takrorlanish bir marta ham bajarilmaydi. Shart chin qiymat qabul qilib, lekin takrorlanish tanasida shart tarkibi o'zgartirilmasa, takrorlanish cheksiz bo'lib qoladi.

Misol: Python so'zi ekranga n mart chiqarilsin.

Bu masalani for sikl operatori yordamida ham ifodalash mumkin, lekin while operatorining mohiyatini o'rganish uchun oddiy masala yordamida qaraymiz.

```
n=input('n=')
n=int(n)
i=1
```

```
while i<=n:
    print(i)
    i+=1

===== RESTART: C:\Users\User\Desktop\1.py
=====

n=3
Python
Python
Python
>>>
```

While operatorini dasturlash tarkibida ishlatish vaqtida doimo takrorlanish tarkibidagi shart bilan tekshiriladigan bitta o‘zgaruvchi olish kerak. Bu masalada shart bilan tekshirish uchun i o‘zgaruvchisi tanlandi. Takrorlanish sonini n orqali i bilan solishtirish natijasida aniqlanadi.

While operatorining ishlash jarayoni yuqoridagi misolda quyidagicha.

Boshlang‘ich holatda $i=1$ shart $i \leq n (1 \leq 3)$ chin

Qadam1: Python so‘zi ekranda chiqariladi $i=2$;

shart $i \leq n (2 \leq 3)$

Qadam2: Python so‘zi ekranda chiqariladi $i=3$;

shart $i \leq n (3 \leq 3)$

Qadam3: Python so‘zi ekranda chiqariladi $i=4$;

shart $i \leq n (4 \leq 3)$ yolg‘on takrorlanish to‘xtatiladi.

Misol. n berilganda $k! \leq n$ shartni qanoatlantiruvchi eng katta k ni aniqlang.

Bu masalani Python dasturlash tilida ifodalash uchun k faktorialni takrorlanish tanasiga joylashtirish kerak, takrorlanish shartini esa $k! \leq n$ ko‘rinishida tasvirlanadi.

```
n=input('n=')
```

```

n=int(n)
p=1
k=1
while(p<=n):
    k+=1
    p=p*k

print(k-1)
===== RESTART: C:\Users\User\Desktop\1.py
=====
n=24
4
>>>

```

Bu masalani Python dasturlash tilidagi ko‘rinishiga e’tibor bersak oxirida `print(k-1)` operatori yozilgan, buni mohiyati shart chin qiymatida bitta qadam ortiq bajariladi, shuning uchun `k-1` holat bo‘yicha chiqariladi. Bu jarayonni `n= 7` qiymat berib, qo‘lda test qilib ko‘rsangiz tushunish oson bo‘ladi.

While operatori tarkibidagi shart ba’zi hollarda o‘zgarmas qiymat ko‘rinishda ham beriladi, bunda shart 0 bilan solishtiriladi, agar qiymat ortib borsa, dastur cheksiz takrorlanish mumkin, bunday holatlarda o‘zgarmas qiymatni kamaytirish kerak.

```

n=input('n=')
n=int(n)

while n:
    n+=1
    print(n)
===== RESTART: C:\Users\User\Desktop\1.py
=====
n=-10

```

```
-9
-8
-7
-6
-5
-4
-3
-2
-1
0
>>>
```

Yuqoridagi dastur tarkibidagi shart n faqat 0 bilan solishtiriladi 0 dan farqli bo'lsa, takrorlanish bajarilaveradi, demak takrorlanish -10 dan boshlab toki 0 gacha bajariladi. Bunday holatlarda takrorlanish cheksiz bo'lib qolish ham mumkin, shart 0 bilan solishtirishni e'tiborga olgan holda, shart yozilish kerak. Takrorlanish cheksiz bo'lgan holatini quyidagi dastur orqali tekshiramiz.

```
n=2
while n:
    n+=1
    print(n)
===== RESTART: C:\Users\User\Desktop\1.py
=====
3
4
5
6
7
...
>>>
```

Bunda while tarkibidagi takrorlanish parametri musbat sondan boshlanib, plus qadam bilan bajarilmoqda, natijada takrorlanish cheksiz

bo‘ladi. Shartli takrorlanuvchi jarayonlarni barcha turlarini, python dasturlash tilida while operatori orqali to‘liq amalga oshirish mumkin.

Nazariy savollar

- 1 Shart asosida takrorlanuvchi jarayonlar deb nimaga aytiladi?
- 2 Shart asosida takrorlanuvchi jarayonlarni turlari?
- 3 Shart oldin takrorlanuvchi jarayon deb nimaga aytiladi?
- 4 While operatorining umumiy ko‘rinishi va vazifasi?
- 5 While operatorining turlari va ishlash jarayoni?

Mustaqil ishlash uchun topshiriqlar

1. N natural soni berilgan ($N > 0$). Kvadrati N dan katta bo‘ladigan eng kichik butun K sonini ($K^2 > N$) aniqlovchi programma tuzilsin. Ildizdan chiqaruvchi funksiyadan foydalanmang.

2. N natural soni berilgan ($N > 0$). Kvadrati N dan katta bo‘lmagan eng katta butun K sonini ($K^2 \leq N$) aniqlovchi programma tuzilsin. Ildizdan chiqaruvchi funksiyadan foydalanmang.

3. N natural soni berilgan ($N > 1$). $3^K > N$ shartni qanoatlantiruvchi eng kichik butun K sonini aniqlovchi programma tuzilsin.

4. N natural soni berilgan ($N > 1$). $3^K \leq N$ shartni qanoatlantiruvchi eng katta butun K sonini aniqlovchi programma tuzilsin.

5. N natural soni berilgan ($N > 1$). $(1 + 2 + \dots + K) \geq N$ shart bajariladigan eng kichik K sonini aniqlovchi programma tuzilsin. 1 dan K gacha bo‘lgan yig‘indi ham ekranga chiqarilsin.

6. N natural soni berilgan ($N > 1$). $(1 + 2 + 3 + \dots + K) \leq N$ shart bajariladigan eng katta K sonini aniqlovchi programma tuzilsin. 1 dan K gacha bo‘lgan yig‘indi ham ekranga chiqarilsin.

7. A soni berilgan ($A > 1$). $(1 + 1/2 + 1/3 + \dots + 1/K) \geq A$ shart bajariladigan eng kichik K sonini aniqlovchi programma tuzilsin.

8. A soni berilgan ($A > 1$). $(1 + 1/2 + 1/3 + \dots + 1/K) \leq A$ shart bajariladigan eng katta K sonini aniqlovchi programma tuzilsin.

9. Bankka boshlang‘ich S so‘m qo‘yildi. Har oyda bor bo‘lgan summa P foizga oshadi ($0 < P < 25$). Necha oydan keyin boshlang‘ich qiymat 2 martadan ko‘p bo‘lishini hisoblovchi programma tuzilsin. Necha

oy K – butun son. Bankda hosil bo‘lgan summa haqiqiy son ekranga chiqarilsin.

10. Sportsmen birinchi kuni 10 km yugurib boshladi. Keyingi kunlari bir oldingi kunga nisbatan P foiz ko‘p yugurdi ($0 < P < 50$). Sportsmenning necha kundan keyin jami yugurgan masofasi 200 km dan oshadi? Jami kunlar soni va masofani (butun son) chiqaruvchi programma tuzilsin.

11. N va M butun musbat sonlari berilgan ($N > M$). N sonini M soniga bo‘lib butun va qoldiq qismlarini bo‘lish va qoldiqni olish amallarini ishlatmasdan topuvchi dastur tuzilsin.

12. N butun soni berilgan ($N > 0$). Bo‘lib butun va qoldiq qismlarini aniqlash orqali, berilgan son raqamlarini teskari tartibda chiqaruvchi dastur tuzilsin.

13. N butun soni berilgan ($N > 0$). Bo‘lib butun va qoldiq qismlarini aniqlash orqali, berilgan son raqamlarini yig‘indisi va raqamlari sonini chiqaruvchi dastur tuzilsin.

14. N butun soni berilgan ($N > 0$). Bo‘lib butun va qoldiq qismlarini aniqlash orqali, berilgan son raqamlarining orasida 2 raqami bor – yo‘qligini aniqlovchi dastur tuzilsin.

15. N butun soni berilgan ($N > 0$). Bo‘lib butun va qoldiq qismlarini aniqlash orqali, berilgan son raqamlarining orasida toq raqamlar bor – yo‘qligini aniqlovchi dastur tuzilsin.

16. N butun soni berilgan ($N > 0$). N sonini tub yoki tub emasligini aniqlovchi dastur tuzilsin.

17. A va B butun musbat sonlari berilgan. Berilgan sonlarning eng kata umumiy bo‘luvchisini aniqlovchi dastur tuzilsin.

18. N butun soni berilgan ($N > 1$). N sonini Fibonachchi sonlari orasida bor – yo‘qligini aniqlovchi dastur tuzilsin. (Fibonachchi sonlari while24 masalada berilgan).

19. N butun soni berilgan ($N > 1$). N sonidan kata bo‘lgan birinchi Fibonachchi sonini aniqlovchi dastur tuzilsin. Fibonachchi sonlari quyidagi qonuniyatlar asosida topiladi.

$$F_1 = 1; F_2 = 1; F_K = F_{K-1} + F_{K-2}; K = 3, 4, \dots$$

20. Fibonachchi soni bo‘lgan N butun soni berilgan ($N > 1$). (Fibonachchi sonlari 19 masalada berilgan). N sonidan bir oldingi va bir keying Fibonachchi sonlarini chiqaruvchi dastur tuzilsin.

21. *Fibonachchi soni bo'lgan N butun soni berilgan ($N > 1$). (Fibonachchi sonlari 19 masalada berilgan). N soni Fibonachchi ketma – ketligining nechanchi xadi ekanini chiqaruvchi dastur tuzilsin.*

22. *e haqiqiy musbat soni berilgan. Ketma-ketlik xadlari quyidagicha aniqlanadi:*

$$a_1=2; a_k=2+1/a_{k-1}; k = 2, 3, \dots$$

$|a_k - a_{k-1}| < e$ shartni qanoatlantiruvchi eng kichik k sonini aniqlovchi dastur tuzilsin. a_k va a_{k-1} ham ekranga chiqarilsin.

23. *e haqiqiy musbat soni berilgan. Ketma-ketlik xadlari quyidagicha aniqlanadi:*

$$a_1=1; a_2=2; a_k = (a_{k-2} + 2*a_{k-1})/3; k = 3, 4, \dots$$

$|a_k - a_{k-1}| < e$ shartni qanoatlantiruvchi eng kichik k sonini aniqlovchi dastur tuzilsin. a_k va a_{k-1} ham ekranga chiqarilsin.

24. *A, B, C musbat butun sonlari berilgan. $A \times B$ to'rtburchak ichida tomoni C bo'lgan kvadratdan nechitasi sig'ishini aniqlovchi dastur tuzilsin. Ko'paytirish va bo'lish amallarini ishlatmang.*

III-BOB. PYTHON DASTURLASH TILIDA FUNKSIYALAR

Mazkur bobda python dasturlash tili tarkibida funksiyalar, funksiyalarni e'lon qilish va funksiyalardan foydalanish usul va algoritmlari qarab o'tilgan. Funksiyalar bo'yicha nazariy tushunchalar, nazariy tushunchalarni o'zlashtirish uchun amaliy masalalar ishlab chiqilgan.

3.1. PYTHON DASTURLASH TILIDA FUNKSIYALARNI YARATISH VA ULARDAN FOYDALANISH

Reja:

1. *Qism dasturlar;*
2. *Funksiya tanasini faollashtirish;*
3. *Global va lokal o'zgaruvchilar;*
4. *Funksiyaga argument berishni soddalashtirish.*

Tayanch so'zlar. *Qism dasturlar, funksiya, global, bir qiymat, argument, def.*

Python dasturlash tili tarkibida dastur tuzish vaqtida ma'lum bir jarayonlar bir necha marta bajarilishi yoki bir necha marta murojat qilinishi mumkin. Dasturlash tillari tarkibida bir necha marta bajarilish kerak bo'lgan jarayonlarni bir marta tasvirlab, keyin shu dasturga murojat qilish imkoniyati mavjud. Dasturlash tillari tarkibida bir necha marta bajarilishi mumkin bo'lgan holatlarni qism dastur sifatida e'lon qilish va kerakli joyga shu qism dasturga murojat qilish mumkin.

*Ta'rif: Dasturlash tilida yaratilgan dastur tarkibida ma'lum bir vazifani bajaruvchi kichik dasturlar **qism dasturlar** deyiladi.*

Qism dasturlarning mohiyati berilgan masala tarkibi ma'lum bir vazifani bajarish kerak bo'ladi, lekin masala tarkibida ham kichik bir vazifani bajarish kerak bo'ladi shunday vazifalarni qism dastur yordamida hal etish mumkin. Bunday masalalar asosan matematik masalalar tarkibida ko'p bo'lishi mumkin, yoki boshqa sohalarda ham uchrab turadi. Masalan biror bir tashkilotning ma'lumotlar bazasi berilganda uning tarkibidagi xodimlarning oylik maoshini hisoblash jarayonini qism dastur yordamida hisoblash maqsadga muvofiq. Agar tashkilot ma'lumotlar bazasi tarkibidagi xodimlarning oylik maoshini qism dastur yordamida

hisoblanmasa, har bir xodim uchun oylik maoshini hisoblash jarayonini keltirish kerak, qism dasturdan foydalansa har bir xodim uchun qism dasturga murojat qilib qo'yiladi. Dastur tuzish vaqtida qism dasturlardan qachon foydalanamiz, agar siz hal etadigan masala yoki muommo tarkibida ma'lum bir jarayonlar ikki va undan ortiq sodir bo'lsa, o'sha jarayonni qism dastur sifatida e'lon qilish kerak va kerakli joyda qism dasturga murojat qilish kerak. Masalan, quyidagi masalaga e'tibor bering.

$$y = \frac{a^{1+2+\dots+n}}{(1+2+\dots+n)^x}$$

Bu masala tarkibiga e'tibor qaratsak, birdan n gacha bo'lgan sonlar yig'indisi bir necha marta bajarilyapti, bu masalani hal etish uchun tuziladigan dastur tarkibida birdan n gacha bo'lgan sonlar yig'indisini hisoblash jarayonini qism dastur qilib e'lon qilish kerak. Agar qism dastur sifatida e'lon qilinsa, masalani hal etishda qism dasturga ikki marta murojat qilish asosida berilgan masalani hal etish mumkin.

Yuqorida keltirilgan masalalarga o'xshash masalalarni hal etish usulini samarasini oshirish uchun qism dasturlardan foydalaniladi. Qism dasturlar python dasturlash tilida qisqacha qilib funksiyalar deb ataladi. Demak, funksiyalar ma'lum bir vazifani bajaruvchi dastur tarkibidagi qism dasturlar ekan.

Ta'rif: Dasturlash tilida tuzilgan dastur tarkibidagi ma'lum bir vazifalarni bajaruvchi qism dasturlar funksiyalar deyiladi.

Dastur ishlash jarayoni samarasi asosan ikki tur bo'yicha oshiriladi, ya'ni xotira hajmi va ishlash tezligi bo'yicha. Dasturning ishlash tezligini oshirish uchun, albatta, dastur tarkibi sodda va ixcham bo'lishi ta'lab etiladi. Dastur tarkibi sodda va ixcham bo'lishi uchun imkoniyat boricha qism dasturlardan foydalanish kerak bo'ladi. Funksiyalar python dasturlash tilida ma'lum bir vazifani bajaruvchi dastur tarkibidagi kichik dasturlar, demak, funksiyani hosil qilish uchun datur tarkibida ma'lum bir vazifalar bir necha marta bajarilish kerak bo'ladi. Python dasturlash tilida har qanday funksiya hech bo'lmaganda bitta natija qaytaradi.

Dastur tarkibidagi funksiyaga uning nomi bilan murojat qilinadi, dastur bajarilish vaqtida funksiya nomi uchrasa, komplyator shu funksiyaning tanasiga murojat qilib natijani funksiya nomiga qaytaradi va

dastur keyingi qadamlarni bajaradi. Python dasturlash tilida funksiyalar asosan ikki guruhga ajratiladi, ya'ni standart va standart bo'lmagan funksiyalar. Standart funksiyalar python dasturlash tili tarkibida biror bir kutubxonaga tarkibiga joylashtirilgan bo'ladi. Standart funksiyalar haqida mazkur qo'llanmaning I-bobida batafsil yoritilgan. Standart bo'lmagan funksiyalar dastur tarkibida yaratiladi va unga nomi bilan murojat qilinadi. Python dasturlash tilida dasturlashning asosiy qismlaridan biri funksiyalardir. Funksiyalarning foydasi shundaki, katta hajmli masala bir necha kichik bo'laklarga bo'linib, har biriga alohida funksiya yozilganda, masala yechish algoritmi ancha soddalashadi. Bunda dasturchi yozgan funksiyalar pythonning standart kutubxonasi va boshqa firmalar yozgan kutubxonalar ichidagi funksiyalar bilan birlashtiriladi. Bu esa ishni bir muncha osonlashtiradi. Ko'p holda dasturda takroran bajariladigan amallarni funksiya sifatida yozish va kerakli joyda ushbu funktsiyani chaqirish mumkin. Funksiyani programma tanasida ishlatish uchun u chaqiriladi, yani uning ismi yoziladi va unga kerakli argumentlar beriladi. () qavslar ushbu funksiya chaqirig'ini ifodalaydi. Masalan: foo(); k = square(1); Demak, agar funksiya argumentlar olsa, ular () qavs ichida yoziladi. Argumentsiz funksiyadan keyin esa () qavslarning o'zi qo'yiladi.

Funksiya tanasini faollashtirish

Python dasturlash tilida funksiyalardan foydalanish uchun, albatta, funksiyalarni dastur tarkibida alohida yozilish kerak. Funksiyalar python tilining ixtiyoriy joyida yozilishi mumkin. Funksiyalar dasturchi ishini juda yengillashtiradi. Funksiyalar yordamida programma modullashadi, qismlarga bo'linadi. Bu esa keyinchalik dasturni rivojlantirishni osonlashtiradi. Dastur yozilish davrida hatolarni topishni yengillashtiradi. Funksiya nomi funksiya bajaradigan vazifadan kelib chiqqan holda qo'yilishi maqsadga muvofiq, chunki dastur tuzuvchi mutaxassislar uchun umumiy holda tushunish oson bo'lishi kerak. Masalan, ketma-ketliklarni yig'indisini hisoblash funksiyasiga **sum()**, to'rtinchi darajali ildizni hisoblash uchun **sqrt4()**, sonning ekubini hisoblash uchun **ekub()**, sonning ekukini hisoblash uchun **ekuk()** va faktorialni hisoblash uchun **fakt()** deb nomlash maqsadga muvofiq bo'ladi. Funksiya tarkibiga kiritilishi kerak bo'lgan o'zgaruvchi argumentlar, albatta, qavs ichida vergul bilan ajratilib yozilishi kerak.

Funksiya tanasini tasvirlash jarayoni ikki qismdan iborat bo‘ladi, ya’ni funksiya sarlavhasi va funksiya tanasidan iborat bo‘ladi. Funksiya tanasini tasvirlash jarayonida funksiya sarlavhasidan keyin nuqtali vergul qo‘yilmaydi va *def* xizmatchi so‘z orqali boshlanib, natija **return** xizmatchi so‘zidan keyin probel bilan yoziladi.

Funksiya tanasini dastur tarkibiga yozishda funksiyaga murojaat qilishdan bir qadam oldin ixtiyoriy joyda yozish mumkin, murojaatdan keyin yozilsa dastur xatolik qaytaradi. Funksiya tanasi tarkibi yozilishida xuddi boshqa dastur tuzilishi kabi unda ishlatiladigan o‘zgaruvchilar e’lon qilinadi, buyruqlar nuqtali vergul yordamida ajratilish shart emas. Funksiyalarni python dasturlash tilida yozilish jarayonining umumiy ko‘rinishi quyidagicha bo‘ladi:

def <funksiya nomi>(<argumentlar>):

funksiya tanasi

return natija

Funksiyalar tansini tasvirlashda funksiya qaytaradigan qiymat yoki ifoda **return** so‘zidan keyin probel bilan yozilishi kerak.

Misol. Python dasturlash tilida ikki sonning yig‘indisini hisoblash uchun yig() fuksiya yarating va unga murojat qilishni tasvirlang.

```
def yig(a,b):
    y=a+b
    return y
a=input('a=')
b=input('b=')
a=int(a)
b=int(b)
z=yig(a,b)
```

```
print(z)
===== RESTART: C:\Users\User\Desktop\1.py
=====
a=10
b=2
12
>>>
```

Yuqoridagi masalani hal etish uchun $yig(a,b)$ funksiyasi yaratildi, funksiya tanasini dastur boshida tasvirlandi. Dastur bajarilish vaqtida kompyator dastur tarkibida $yig(a,b)$ funksiyasini uchratganda bajarilish qadami $yig(a,b)$ funksiya tanasiga o'tib natijani hisoblab qaytib keladi va bajarilish qadami buyruqlar ketma-ketligi bo'yicha bajariladi. Yuqoridagi masalani ikkinchi ko'rinishda ham bajarish mumkin, ya'ni funksiya tanasi dasturni ixtiyoriy joyida keltirilishi mumkin.

```
a=input('a=')
b=input('b=')
def yig(a,b):
    y=a+b
    return y
a=int(a)
b=int(b)
z=yig(a,b)
print(z)
===== RESTART: C:\Users\User\Desktop\1.py
=====
a=-12
b=4
```

```
-8
>>>
```

Ikki sonning yi'gindisini hisoblash uchun keltirilgan dasturning ikkinchi ko'rinishi faqat $yig(a,b)$ funksiyasi dastur ichida yozilgan. Return xizmatchi so'zidan keyin funksiya qaytaradigan qiymat natijasini ifodalovchi ifodani ham yozish mumkin.

Misol: Sonning natural bo'luvchilar yigindisini aniqlash uchun $bul_yig(x)$ funksiyasini yarating.

```
def bul_yig(x):
    s=0
    for i in range(1,x+1):
        if(x%i==0):
            s=s+i
    return s

a=input('a=')
a=int(a)
z=bul_yig(a)
print(z)

===== RESTART: C:\Users\User\Desktop\1.py
=====

a=6
12
>>>
```

Yuqoridagi masalalarga e'tibor qaratsak masala tarkibidagi funksiyalardan dastur tarkibida ixtiyoriy joyida ixtiyoriy marta foydalanish mumkin.

Global va lokal o'zgaruvchilar

Python dasturlash tilida tuziladigan dasturlar tarkibida bir nechta o'zgaruvchilardan foydalaniladi. Python dasturlash tilida funksiyalar mavzusidan keyin o'zgaruvchilar ikki turga ajratiladi, ya'ni global va lokal o'zgaruvchilar.

*Ta'rif: Dastur tarkibining ixtiyoriy joyida foydalanish mumkin bo'lgan o'zgaruvchilar **global o'zgaruvchilar** deyiladi.*

Global o'zgaruvchilar dasturning ixtiyoriy qismida o'z qiymatini saqlaydi, hattoki, dasturning ixtiyoriy joyida o'z qiymatini ushlab qoladi. Global o'zgaruvchilar qism dasturning tashqarisida faollashtiriladi.

Python dasturlash tilidagi funksiyalar tarkibidagi o'zgaruvchilar global hisoblanmaydi. Global bo'lmagan o'zgaruvchilar faqatgina o'z qism funksiya tarkibiga tegishli bo'ladi.

*Ta'rif: Python dasturlash tilidagi funksiyalar tarkibidagi o'zgaruvchilar **lokal o'zgaruvchilar** deyiladi.*

Dastur tarkibidagi qism funksiyalar tarkibidagi barcha o'zgaruvchilar lokal o'zgaruvchilar hisoblanadi, funksiya tarkibidagi o'zgaruvchilar faqatgina funksiyaning tarkibi ichida o'rinli bo'ladi. Global va lokal o'zgaruvchilarni aniqlash uchun quyidagi orqali aniqlaymiz.

Misol. *pow_uzb()* funksiyasini yarating, bunda *x* o'zgaruvchini qiymatini lokal ekanligini aniqlang.

```
def pow_uzb(a):
    x=2 # lokal
    y=a**x
    return y
b=input('b=')
b=int(b)
z=pow_uzb(b)
print(z,x)

===== RESTART: C:\Users\User\Desktop\1.py
```

```
=====
```

```
b=6
```

```
Traceback (most recent call last):
```

```
File "C:\Users\User\Desktop\1.py", line 8, in <module>
```

```
    print(z,x)
```

```
NameError: name 'x' is not defined
```

```
>>>
```

Yuqoridagi dasturga e'tibor bersak x ning funksiya tarkibidagi qiymati faol, lekin `print()` funksiyasi ichidagi qiymati faol emas, yani dastur xatolik beryapti. Demak bu yerda x lokal o'zgaruvchidir, lokal o'zgaruvchilar qism dastur tashqarisida faol bo'lmaydi. Endi xuddi shu dasturni x o'zgaruvchisin global sifatida ishlatib ko'ramiz.

```
def pow_uzb(a):
```

```
    x=2
```

```
    y=a**x
```

```
    return y
```

```
b=input('b=')
```

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

```
z=pow_uzb(b)
```

```
x=-4 # global
```

```
print(z,x)
```

```
===== RESTART: C:\Users\User\Desktop\1.py =====
```

```
b=5
```

```
25 -4
```

```
>>>
```

Yuqoridagi dastur tarkibiga e'tibor qaratsak, x o'zgaruvchiga global sifatida -4 soni berilgan edi dastur natija qaytardi. Endi esa x global o'zgaruvchi sifatida ishlatilishi mumkin. Global o'zgaruvchilarni dasturning ixtiyoriy joyida foydalanish imkoniyati mavjud, lokal o'zgaruvchilarni esa faqatgina o'z funksiya bloki ichida foydalanish mumkin. Global va lokal o'zgaruvchilar haqida aytilganda global va lokal o'zgaruvchilarni **yashash davri** haqida tushuncha beriladi. Global o'zgaruvchilarni yashash davri dastur ishga tushirilgandan toki tugaguncha hisoblanadi. Lokal o'zgaruvchilarni yashash davri esa faqatgina o'zgaruvchi keltirilgan funksiya ishga tushirilgan vaqti hisoblanadi. Agar dasturdagi global o'zgaruvchilar funksiya tarkibida alohida boshlang'ich qiymati berilsa, uning global qiymati faqat funksiya ichida unutiladi va funksiya tarkibidagi qiymat hisobga olinadi. Global va lokal o'zgaruvchilarni yashash davrini quyidagi masala orqali aniqlaymiz:

```
a=10
b=20
def yig(a,b):
    a=2
    b=3
    return a+b
z=yig(a,b)
print('yig=',z)
print('a=',a)
print('b=',b)
===== RESTART: C:\Users\User\Desktop\1.py
=====
yig= 5
a= 10
b= 20
```

```
>>>
```

Dastur tarkibida $a=20$, $b=10$ global sifatida aniqlangan edi, lekin $yig(a,b)$ funksiyasini tarkibida $a=2$, $b=3$ bo'lganligi uchun a va b larning boshlang'ich qiymatlari unutiladi va natija 9 ga teng bo'ladi. Global o'zgaruvchilarning qiymatlari qism dastur tarkibidan chiqqandan yana faol bo'ladi, chunki dastur oxirida $print('a=',a)$ va $print('b=',b)$ natijasida $a=10$ va $b=20$ natijani chiqaradi.

Boshqa dasturlash tillarida global o'zgaruvchi qism dastur tarkibida boshqa qiymat olsa eski qiymat unutiladi (C++ tilida), lekin python tilida global o'zgaruvchi qiymatini o'zgartirmaydi, buni isboti yuqoridagi misolda o'z aksini topgan.

Funksiyaga argument berishni soddalashtirish

Python dasturlash tili tarkibida qism dastur funksiyaning argumentlarini soddalashtirilgan holatlarda ishlatish imkoniyati mavjud. Bunda asosan jimlik qoidasi bo'yicha funksiya tarkibidagi o'zgaruvchi qiymati olinadi, aks holda foydalanuvchi tomonidan berilgan qiymat qabul qilinadi. Bu jarayonni tushunib olish uchun quyidagi dasturlarga e'tibor bering.

Misol. $daraja(a,x)$ funksiyasini yarating bu funksiya $daraja(a)$ sifatida ham natija qaytarsin.

```
def daraja(a,x=2):  
    y=a**x  
    return y  
a=3  
b=3  
z=daraja(a,b)  
print('z=',z)  
k=daraja(a)  
print('k=',k)
```

```
t=daraja(1)
print('t=',t)
===== RESTART: C:\Users\User\Desktop\1.py
=====
z= 27
k= 9
t= 1
>>>
```

Yuqoridagi dasturda $daraja(a,x=2)$ funksiya ikkita argument bilan shakllantirilgan lekin dasturning asosiy tanasida $z=daraja(a,b)$, $k=daraja(a)$ va $t=daraja(1)$ ko‘rinishlarida murojaat qilinmoqda. Bunda funksiya argumenti soddalashtirilgan holatda bitta qiymat ham qabul qilishi mumkin, bunday holatlarda funksiya jimlik qoidasi bo‘yicha o‘zining tarkibidagi qiymatni qabul qiladi. z,k va t o‘zgaruvchilarni qiymatlari har xil bo‘lmoqda, z ning argumenti b deb berilmoqda, k va t ning argument qiymatlari berilmayapti.

Nazariy savollar

- 1 *Qism dastur deganda nimani tushunasiz?*
- 2 *Funksiyalarni shakllantirish usullarini tushuntirib bering?*
- 3 *Funksiyalarni shakllantirishni umumiy ko‘rinishi?*
- 4 *Funksiya tanasini tasvirlash?*
- 5 *Global va lokal o‘zgaruvchilar deganda nimani tushunasiz?*
- 6 *Global va lokal o‘zgaruvchilarni yashash davrini tushuntirib bering?*
- 7 *Funksiyaga argument berishni soddalashtirish jarayonini tushuntirib bering?*

Mustaqil ishlash uchun topshiriqlar

1. *To‘g‘ri to‘rtburchakning yuzini va perimetrini uning qarama – qarshi uchlari koordinatasi orqali hisoblovchi funksiya hosil qiling. (x1,*

y_1, x_2, y_2) to'g'ri to'rtburchakning qarama – qarshi uchlari RectPS funksiya orqali 2 ta to'rtburchak yuzi va perimetrini hisoblang.

2. Natural sonning raqamlari soni va raqamlari yig'indisini hisoblovchi funksiya hosil qiling.

3. Butun musbat sonining raqamlarini teskari tartibda chiqaruvchi funksiya hosil qiling. Bu funksiya orqali a, b, c sonlarining raqamlari teskari tartibda chiqaruvchi dastur tuzilsin.

4. Kiritilgan K butun musbat sonining o'ng tarafiga (oxiriga) R raqamini ($1 \leq R \leq 9$) qo'shuvchi funksiya hosil qiling.

5. Kiritilgan K butun musbat sonining chap tarafiga (boshiga) R raqamini ($1 \leq R \leq 9$) qo'shuvchi funksiya hosil qiling.

6. Ikkita sonning qiymatini almashtiruvchi Swap nomli funksiya hosil qiling. Swap funksiya orqali A, B, C, D sonlaridan $(A, B), (D, C)$ juftliklarining qiymatlarini almashtiruvchi dastur tuzilsin.

7. X va Y sonlaridan kichigini X ga va kattasini Y ga yozuvchi $\text{Minmax}(X, Y)$ funksiyasini hosil qiling. Minmax funksiyasini 4 marta chaqirish orqali a, b, c, d butun sonlaridan kattasini va kichigini aniqlovchi dastur tuzilsin.

8. A, B, C sonlarini o'sish tartibida joylashtiruvchi $\text{SortInc3}(A, B, C)$ funksiyasini hosil qiling. Ya'ni A, B, C sonlari qiymatlarini shunday almashtiringki, natijada A ning qiymati eng kichik va C ning qiymati eng katta bo'lsin. Bu funksiya orqali (A_1, B_1, C_1) va (A_2, B_2, C_2) sonlarini tartiblang.

9. A, B, C sonlarini kamayish tartibida joylashtiruvchi $\text{SortDec3}(A, B, C)$ funksiyasini hosil qiling. Ya'ni A, B, C sonlari qiymatlarini shunday almashtiringki, natijada A ning qiymati eng katta va C ning qiymati eng kichik bo'lsin. Bu funksiya orqali (A_1, B_1, C_1) va (A_2, B_2, C_2) sonlarini tartiblang.

10. o'ngga siklik siljishni amalga oshiruvchi $\text{ShiftRight3}(A, B, C)$ funksiyasini hosil qiling. Ya'ni A ning qiymati B ga, B ning qiymati C ga, C ning qiymati A ga o'tib qolsin. Bu funksiya orqali (A_1, B_1, C_1) va (A_2, B_2, C_2) sonlarini siljiting.

3.2. PYTHON DASTURLASH TILIDA KO‘P QIYMAT QAYTARUVCHI FUNKSIYALAR VA ULARDAN FOYDALANISH

Reja:

1. *Ko‘p qiymat qaytaruvchi funksiyalar;*
2. *Ko‘p qiymat qaytaruvchi funksiyalarni shakllantirish;*

Tayanch so‘zlar. *Ko‘p qiymat, return, funksiya tanasi.*

Python dasturlash tilida funksiyalar dastur tarkibidagi kichik dasturlar hisoblanadi, ular dastur bajarilish natijasida bitta qiymat qaytaradi. Masalan, sonning faktorialini, sonlarning ekubini, sonlarning ekukini va hakoza shunga o‘xshash natijalarni qaytaradi. Lekin dastur tarkibida ikki va undan ortiq natija qaytaradigan kichik muommolar ham mavjud. Masalan, kvadrat funksiyaning ildizlarini aniqlash, unda dastur ko‘pi bilan ikkita qiymat qaytarish kerak, massivlarni o‘sish yoki kamayish tartibida tartiblash va hakoza shunga o‘xshash masalalar ko‘p o‘chraydi. Bu turdagi masalalarni yechish uchun oddiy funksiyalardan foydalanish maqsadga muvofiq bo‘lmaydi. Python dasturlash tilida ikki va undan ortiq qiymat qaytaradigan funksiyalarni qisqacha qilib ko‘p qiymat qaytaruvchi funksiyalar deb nomlaymiz. Yuklangan funksiyalar chaqirilganda, qaysi funksiyaning chaqirish kirish parametrlarining soniga, ularning tipiga va navbatiga bog‘liqdir. Yani ism yuklanishida funksiyaning imzosi rol o‘ynadi. Agar kirish parametrlari va ismlari ayni funksiyalarning farqi faqat ularning qaytish qiymatlarida bo‘lsa, bu yuklanish bo‘lmaydi, kompilyator buni xato deb e‘lon qiladi. Funksiya yuklanishi asosan ayni ishni yoki amalni farqli usul bilan farqli ma‘lumot tiplari ustida bajarish uchun qo‘llaniladi. Masalan bir fazoviy jismning hajmini hisoblash kerak bo‘lsin. Har bir jismning hajmi farqli formula yordamida, yani farqli usulda topiladi, bir jismda radius tushunchasi bor bo‘lsa, boshqasida asos yoki tomon tushunchasi bor bo‘ladi, bu esa farqli ma‘lumot tiplariga kiradi. Lekin amal ayni hajmni hisoblash. Demak, biz funksiya yuklanishi mexanizmini qo‘llasak bo‘ladi. Bir hil amalni bajaruvchi funksiyalarni ayni nom bilan atashimiz esa, dasturni o‘qib tushunishni osonlashtiradi. Kompilyator biz bergan funksiya imzosidan (imzoga funksiya ismi va kirish parametrlari kiradi, funksiyaning qaytish qiymati esa imzoga kirmaydi) yagona ism tuzadi, dastur ijrosi davruda esa

funksiya chaqirig‘idagi argumentlarga qarab, kerakli funksiyani chaqiradi. Demak, funksiyani chaqirish uning nomiga bog‘liq ekan. Ko‘p qiymat qaytaruvchi funksiyalar esa, albatta, uning imzosida prosedura nomi kirish va chiqish parametrlari, albatta, keltirilishi kerak, chunki ko‘p qiymat qaytaruvchi funksiyalar tarkibida bir nechta qaytariladigan qiymatlar, albatta, biror bir parametrlarga bog‘langan bo‘ladi.

Ko‘p qiymat qaytaruvchi funksiyalarni shakllantirish

Ko‘p qiymat qaytaruvchi funksiyalar, oddiy funksiyalardan farqi shundaki, u faqat bitta qiymat qaytarmaydi, balki bir nechta qiymat qaytarishga mo‘ljallangandir. Yagona nom bilan saqlangan ko‘p qiymat qaytaruvchi funksiyalar yordamida ikki sonning yig‘indisini, ko‘paytmasini, nisbatini va ayirmasini hisoblovchi funksiya yaratish mumkin.

Ta’rif: Python dasturlash tilining dastur tarkibida ikki va undan ortiq qiymat qaytaradigan qism dasturlar ko‘p qiymat qaytaruvchi funksiyalar deyiladi.

Ko‘p qiymat qaytaruvchi funksiyalarni shakllantirishda, albatta, uning kiritish qiymatlar parametrlari keltirilishi kerak. Ko‘p qiymat qaytaruvchi funksiyalarni shakllantirish usullari to‘liq funksiyalarni e’lon qilish usullari bilan bir xil bo‘ladi, bu yerda ham funksiya murojaat qilinishidan oldin shakllantirilgan bo‘lishi kerak.

Ko‘p qiymat qaytaruvchi funksiyalar python dasturlash tilida shakllantirishining umumiy ko‘rinishi quyidagicha bo‘ladi.

def <funksiya nomi>(<argumentlar>):

funksiya tanasi

return <o‘zgaruvchilar>

Ko‘p qiymat qaytaruvchi funksiyalarni shakllantirishda qavs ichida **argumentlar** sifatida kiritish parametrlari tasvirlanadi, keyin return xizmatchi so‘zidan keyin **o‘zgaruvchilar** qiymat qaytaruvchi parametrlar sifatida tasvirlanadi. Bunda return so‘zidan keyingi o‘zgaruvchilar, mos ravishda funksiya tanasidagi o‘zgaruvchilar qiymatlarini qabul qiladi.

Ko‘p qiymat qaytaruvchi funksiyalarni tasvirlash uchun ikki sonning yig‘indisi va ko‘paytnasini hisoblovchi $kop(x,y)$ nomi bilan yaratilgan funksiya dasturiga e‘tibor bering.

Misol. *Ikki sonning yig‘indisi va ko‘paytnasini hisoblovchi $kop(x,y)$ funksiya yarating va bu funksiya natijasidan foydalanish dasturini tuzing.*

```
def kop(x,y):
    t=x+y
    z=x*y
    return t,z
a=input('a=')
b=input('b=')
a=int(a)
b=int(b)
kop(a,b)
n,m=kop(a,b)
print('a+b=',n)
print('a*b=',m)
===== RESTART: C:\Users\User\Desktop\1.py
=====
a=15
b=10
a+b= 25
a*b= 150
>>>
```

$kop(x,y)$ funksiyasi a va b sonlarining yig‘indisini n o‘zgaruvchiga, ko‘paytnasini esa m o‘zgaruvchiga saqlaydi. Funksiya qiymat qaytaruvchi parametrlar, albatta, return so‘zidan keyin yoziladi, natijada qism dastur tarkibidagi bir nechta o‘zgaruvchi qiymatlarini dasturning asosiy tanasiga olib chiqish imkonini yaratiladi.

Misol. *Chiziqli funksiyaning yechimi cheksiz, yagona va mavjud emaslik holatlarini aniqlash dasturini tuzing.*

```

def chiziq_fun(a,b):
    if a==b:
        print('cheksiz yechimga ega')
    elif (a==0 and b!=0):
        print('yechimga ega emas')
    elif (a!=0 and b!=0):
        print('yagona yechimga ega')

a=input('a=')
b=input('b=')
a=int(a)
b=int(b)
chiziq_fun(a,b)
===== RESTART: C:\Users\User\Desktop\1.py
=====
a=12
b=18
yagona yechimga ega
>>>

```

Yuqoridagi dasturga e'tibor qaratsak, return so'zi ishlatilmayapti, funksiya natijasi biror o'zgaruvchiga olib chiqish shart bo'lmasa return so'zi ham kerak emas.

Nazariy savollar

- 1 Ko'p qiymat qaytaruvchi funksiyalar deganda nimani tushunasiz?
- 2 Ko'p qiymat qaytaruvchi funksiyalarni shakllantirishni tushuntirib bering?
- 3 Ko'p qiymat qaytaruvchi funksiyalarni umumiy ko'rinishi?
- 4 Ko'p qiymat qaytaruvchi funksiyalarga qanday usullarda murojat qilinadi.

Mustaqil ishlash uchun topshiriqlar

1. Chapga siklik siljishni amalga oshiruvchi $\text{ShiftLeft3}(A, B, C)$ funksiyasini hosil qiling. Ya'ni C ning qiymati B ga, B ning qiymati A ga, A ning qiymati C ga o'tib qolsin. Bu funksiya orqali $(A1, B1, C1)$ va $(A2, B2, C2)$ sonlarini siljiting.

2. Haqiqiy sonning ishorasini aniqlovchi ishora nomli funksiya hosil qiling. Funksiya argumenti noldan kichik bo'lsa -1 ; noldan katta bo'lsa 1 ; nolga teng bo'lsa 0 qiymat qaytarsin. Haqiqiy a va b sonlari uchun $\text{ishora}(a) + \text{ishora}(b)$ ifodasi hisoblansin.

3. Kvadrat tenglamaning ildizlar sonini aniqlovchi funksiya hosil qiling. $A \cdot x^2 + B \cdot x + C = 0$ ko'rinishidagi tenglama kvadrat tenglama deyiladi. (A noldan farqli son)

4. Doiraning yuzini hisoblovchi funksiya hosil qiling. Bu funksiya yordamida 3 ta doira yuzini hisoblang. Doiraning yuzi $S = \pi R^2$ orqali hisoblanadi. $\pi = 3.1415$ ni o'zgarmas deb qabul qiling.

5. Markazi bir nuqtada bo'lgan, $R1$ va $R2$ radiusga ega 2 ta aylananing ustma-ust tushmaydigan (kesishmaydigan) qismining yuzasini topuvchi RingS nomli funksiya hosil qiling. Doiraning yuzini hisoblash formulasidan foydalaning, $S = \pi R^2$. $\pi = 3.1415$ ni o'zgarmas deb qabul qiling.

6. To'g'ri burchakli uchburchakning katetlari A va B berilganda, uning perimetrini hisoblovchi TriangleP nomli funksiya hosil qiling.

7. A va B sonlari orasidagi sonlar yig'indisini hisoblovchi $\text{SumRangle}(A, B)$ nomli funksiya hosil qiling. Agar $A > B$ bo'lsa, funksiya 0 qiymat qaytaradi. Bu funksiya orqali A dan B gacha va B dan C gacha bo'lgan sonlar yig'indisini hisoblang. A, B, C butun sonlar.

8. Arifmetik amallarni bajaruvchi $\text{Calc}(A, B, Op)$ funksiyasini hosil qiling. A va B haqiqiy sonlar. Op o'zgaruvchisi orqali bajariladigan arifmetik amal aniqlanadi. 1-ayirish, 2-ko'paytirish, 3-bo'lish, boshqalari qo'shish. Shu funksiya orqali A va B sonlari uchun $N1, N2, N3, N4$ amallari bajarilsin. ($N1-N4$ butun sonlar)

9. X va Y butun sonlari berilgan (X va Y noldan farqli). (X, Y) nuqta qaysi chorakda ekanini aniqlovchi Quarter nomli funksiya hosil qiling. Bu funksiya orqali 4 ta nuqtaning choragini aniqlang.

10. Butun sonning juft – toqligini aniqlovchi $Even(K)$ funksiyasini hosil qiling. Funksiya K juft son bo'lsa – true, aks xolda false qiymat qaytarsin. Bu funksiya orqali 3 ta sonning juft yoki toqligi aniqlansin.

11. $IsSquare(K)$ mantiqiy funksiyasini hosil qiling. ($K > 0$). Agar K biror butun sonning kvadrati bo'lsa – true, aks xolda false qiymat qaytarsin. Shu funksiya orqali 3 ta sonni tekshiring.

12. $IsPower5(K)$ mantiqiy funksiyasini hosil qiling. ($K > 0$). Agar K soni 5 ning biror darajasi bo'lsa – true, aks xolda false qiymat qaytarsin. Shu funksiya orqali 5 ta sondan nechtasi 5 ning darajasi ekanini aniqlovchi dastur tuzilsin.

13. $IsPowerN(K)$ mantiqiy funksiyasini hosil qiling. ($K > 0$). Agar K soni N ning biror darajasi bo'lsa – true, aks xolda false qiymat qaytarsin. Shu funksiya orqali 5 ta sondan nechtasi N ning darajasi ekanini aniqlovchi dastur tuzilsin.

14. $IsPrime(N)$ mantiqiy funksiyasini hosil qiling. ($N > 0$). Agar N soni tub bo'lsa – true, aks xolda false qiymat qaytarsin. Shu funksiya orqali kiritilgan k ta sondan nechtasi tub ekanini aniqlovchi dastur tuzilsin.

15. Butun qiymat qaytaruvchi $DigitCount(K)$ funksiyasini hosil qiling. ($K > 0$). Funksiya K ning raqamlari sonini qaytarsin. Shu funksiya orqali 5 ta sonning raqamlari soni aniqlansin.

16. Butun qiymat qaytaruvchi $DigitN(K, N)$ funksiyasini hosil qiling. ($K > 0$). Funksiya K sonining N – raqamini qaytarsin. Agar K soni raqamlari N dan kichik bo'lsa, minus bir qaytarsin. Shu funksiya orqali $K1, K2, K3$ sonlarining N – raqami aniqlansin.

3.3. REKURSIV FUNKSIYALAR

Reja:

1. Rekursiv funksiyalar;
2. Rekursiv funksiyalarga oid dasturlar.

Tayanch so'zlar. Rekursiya, murojaat, rekursiv funksiya.

Boshqa dasturlash tillari kabi Python dasturlash tilida ham rekursiv funksiyalar va ulardan foydalanish imkoniyati yaratilgan. Ba'zi hollarda ketma ket jarayonlarni rekursiv holda amalga oshirish dastur ish faoliyati

samaradorligini oshirishga xizmat qiladi. rekursiv funksiyalarning ishlash jarayoni bu funksiyalar o'z o'ziga murojaatni amalga oshiradi.

Ta'rif: O'z o'ziga murojaatni amalga oshiradigan funksiyalar rekursiv funksiyalar deb nomlanadi.

Rekursiv funksiyalarga berilgan ta'rif shuni ko'rsatadiki jarayonlarni amalga oshirishda yaratilgan qism dasturlar albatta o'z o'ziga murojaat orqali bajariladi.

Misol. Faktorialni hisoblash jarayonini rekursiv holda bajarishni amalga oshirish dasturini tuzing.

```
def fakt(n):
    if n==0:
        return 1
    else:
        return n*fakt(n-1)
a=input('a=')

a=int(a)
b=fakt(a)
print(b)
===== RESTART: C:\Users\User\Desktop\1.py
=====
a=50
3041409320171337804361260816606476884437764156896051200000
0000000
>>>
```

Yuqoridagi dasturga e'tibor qaratsak, $fakt(a)$ funksiyasi tarkibida yana $fakt(a)$ funksiyasiga murojaat qilinmoqda bu toki $a=0$ bo'lguncha bajariladi.

Misol. $\sqrt{n+\sqrt{(n-1)+\dots+\sqrt{1}}}$ ni rekursiv funksiya yordamida dasturini tuzing.

```
from math import*
def ildiz_yig(n):
```

```

if n==0:
    return 1
else:
    return sqrt(n+sqrt(n-1))
a=input('a=')

a=int(a)
b=ildiz_yig(a)
print(b)
===== RESTART: C:\Users\User\Desktop\1.py
=====
a=50
7.54983443527075
>>>

```

$\sqrt{n+\sqrt{(n-1)+\dots+\sqrt{1}}}$ ni rekursiv funksiya yordamida hal etish uchun *ildiz_yig(n)* nomli rekursiv funksiya yaratildi. Python tilini imkoniyati kattaligi yuqoridagi birinchi misolda yaqqol nomoyon bo‘lmoqda, natijalar o‘n yoki yigirma xonali bo‘lsa ham xotirada muommo bo‘lmaydi.

Nazariy savollar

- 1 *Rekursiv funksiyalar deganda nimani tushunasiz?*
- 2 *Rekursiv funksiyalarni shakllantirishni tushuntirib bering?*
- 2 *Rekursiv funksiyalarni ishlash jarayonini tushuntirib bering?*

IV-BOB. PYTHON DASTURLASH TILIDA MURAKKAB TURLAR RO‘YXAT, KORTEJ, LUG‘AT, TO‘PLAM VA MASSIVLAR

Python dasturlash tillarida oddiy turdan tashqari boshqa turdagi o‘zgaruvchilar ham ishlatiladi. Ba’zi hollarda bir o‘zgaruvchining bir nechta xil ko‘rinishlari mavjud bo‘ladi, matematika kabi ma’lum bir vazifani bajaruvchi o‘zgarmas turga mansub bo‘lgan funksiyalardan foydalanishga to‘g‘ri keladi. Kompyuter xotirasiga ma’lumotlarni oddiy bo‘lmagan usullar yordamida joylashishiga ma’lumotlarning murakkab turlari yordam beradi. Murakkab turlarga ro‘yxat, slovar, massivlar, satrlar va to‘plamlarni misol tariqasida olish mumkin.

Murakkab turlardan foydalanish, dasturlashda bir nechta o‘zgaruvchilardan foydalanishni yuqotadi. Massiv, satr va to‘plam ko‘rinishdagi ma’lumot turlari bir o‘zgaruvchi yordamida bir nechta qiymatlarni kompyuter xotirasiga joylashtirish imkoniyatini beradi. Murakkab turlardan dasturlash tilida foydalanishda bir qancha qulayliklar mavjud, bunda o‘zgaruvchini ixtiyoriy elementiga uning indeksleri orqali murojat qilish imkoniyati mavjud.

4.1 PYTHON DASTURLASH TILIDA RO‘YXATLAR VA KORTEJLAR

Reja:

- 1. Ro‘yxatlar va ularning umumiy ko‘rinishi;*
- 2. Ro‘yxatlarga oid dasturlar;*
- 3. Kortejlar va ularning umumiy ko‘rinishi;*
- 4. Kortejlarga oid dasturlar;*

Tayanch so‘zlar. *Ro‘yxat, kortej, dinamik xotira, sorted, append.*

Python dasturlash tilida kompyuter xotirasiga bir o‘zgaruvchi yordamida bir nechta qiymatlarda foydalanishga to‘g‘ri keladi. Bir o‘zgaruvchi bilan bir nechta qiymat ustida amallar bajarish boshqa dasturlash tillaridan farqli ravishda, berilgan ma’lumotlar bir turga mansub bo‘lishi shart emas.

Ro‘yxatlar

Bu qismda dasturdagi ma’lumot strukturalari bilan tanishishni

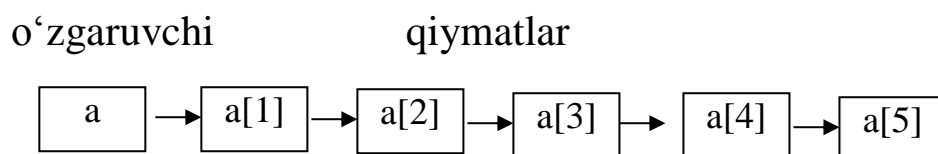
boshlaymiz. Dasturda ikki asosiy tur ma'lumot strukturalari mavjuddir. Birinchisi statik, ikkinchisi dinamikdir. Statik deganimizda xotirada egallagan joyi o'zgaras, dastur boshida beriladigan strukturalarni nazarda tutamiz. Dinamik ma'lumot tiplari dastur davomida o'z hajmini, egallagan xotirasini o'zgartirishi mumkin. Python dasturlash tilining imkoniyatlar kengligining yana bir xususiyati, o'zgaruvchilarning dinamikligidir. Python dasturlash tilida har xil turdagi bir nechta ma'lumotlarni boshqarish va qayta ishlash imkoniyati mavjud. Bu imkoniyatni python dasturlash tilida ro'yxatlar(list) amalga oshiradi. Alohida bir o'zgaruvchini ko'rsatish uchun ro'yxat nomi va kerakli o'zgaruvchi indeksini yoziladi.

Ta'rif: Har xil turga mansub bo'lgan yagona nom bilan saqlanuvchi tartiblangan ma'lumotlar majmuasi ro'yxat(list) deyiladi.

Ro'yxatlar yagona o'zgaruvchi bilan kompyuter xotirasiga saqlanadi, uning elementlari ma'lum bir indekslar bilan tartiblab joylashtiriladi. Python dasturlash tilida ro'yxatlarni boshqa dasturlash tillaridagi bir o'lchovli massivlarga o'xshatish mumkin, lekin pythonda ma'lumotlar bir turga mansub bo'lmasligi ham mumkin.

Python dasturlash tilida ro'yxatlardan foydalanishda bozordagi mahsulotlarning narxini olish mumkin. Mahsulot narxlarini ro'yxat sifatida qaralganda narx1, narx2, narx3, ..., narxn ko'rinishda bir nechta mahsulot narxlarini kompyuter xotirasiga saqlab undan foydalanish mumkin.

Odatda ro'yxatlar zarurat, katta hajmdagi tartiblangan, lekin chekli elementlarga oid masalalarni hal etishda yuzaga keladi. Dastur ishlatilishi davomida ro'yxatlar aniq nomga ega bo'lishi va uning elementlari ma'lum bir turda bo'lishi kerak. Python dasturlash tilida ro'yxatlar kompyuter xotirasiga quyidagi shaklda saqlanadi.



Yuqoridagi holat bo'yicha ro'yxatlar kompyuter xotirasiga saqlanadi, bunda ro'yxatning ixtiyoriy elementiga murojat qilish uchun uning indeks nomeri bo'yicha murojat qilinadi.

Ro‘yxatlarni boshlang‘ich qiymatlari bergan holatda faollashtirish quyidagicha amalga oshiriladi.

<ro‘yxat o‘zgaruvchisi>=[qiymat1, qiymat2, ...]

Ro‘yxatni Python dasturlash tilida faol qilish uchun, albatta, elementlar soni berilish shart emas, ro‘yxatning elementlar soni uning tarkibidagi qiymatlariga qarab aniqlanadi.

Python dasturlash tilida ro‘yxatlarni faollashtirish va ulardan foydalanish.

```
>>> a=[1,12,'b',10.3,5,5.2]
>>> a
[1, 12, 'b', 10.3, 5, 5.2]
>>> a[1]
12
>>> a[2]
'b'
>>> a[1]+a[0]
13
```

Yuqoridagi dastur kodiga e‘tibor bersak, bunda ro‘yxat elementlari uchun quyidagilar o‘rinli bo‘ladi:

- Ro‘yxat elementlari ixtiyoriy turda;
- Ro‘yxat elementlari vergul bilan ajratiladi;
- Ro‘yxat elementlari soni oldindan berish shart emas;
- Ro‘yxat elementlari 0- tartibdan boshlanadi;
- Ro‘yxat elementlariga murojaat indekslar orqali amalga oshiriladi.

Ro‘yxat elementlarini tashkil qilish va ro‘yxat elementlari ustida amallar bajarishni quyidagi masala orqali qaraymiz.

Ro‘yxat elementlariga o‘zgarirish kiritish jarayonini quyidagi dastur orqali amalga oshiriladi.

```
>>> a=[10,2.3,'h',9]
>>> a
[10, 2.3, 'h', 9]
>>> a[3]='salom'
>>> a
[10, 2.3, 'h', 'salom']
>>>
```

Python dasturlash tilida ro'yxat elementlari uchun quyidagi ko'rinishdagi amal funksiyalar aniqlangan.

len (L) - L ro'yxatidagi elementlar sonini aniqlaydi

max (L) - L ro'yxatdagi maksimal elementni aniqlaydi

min (L) - L ro'yxatidagi minimal elementni aniqlaydi

sum (L) - L ro'yxatidagi qiymatlarning yig'indisini aniqlaydi

sorted (L) - L ro'yxat elementlarini saralaydi

del(a[i]) – L ro'yxatning a[i] elementi o'chiriladi

Python dasturlash tilida ro'yxat elementlari uchun aniqlangan funksiyalardan foydalanishni quyidagi dastur orqali qarab chiqamiz.

```
>>> a=[7,8,3,2,5,6,1,4,10,9]
>>> a
[7, 8, 3, 2, 5, 6, 1, 4, 10, 9]
>>> len(a)
10
>>> max(a)
10
>>> min(a)
1
>>> sum(a)
55
>>> sorted(a)
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
>>> del(a[1])
>>> a
```

```
[7, 3, 2, 5, 6, 1, 4, 10, 9]
```

```
>>>
```

```
>>> del(a[:])
```

```
>>> a
```

```
[]
```

```
>>>
```

Ro'yxat elementlari tarkibidan, boshqa ro'yxat elementlari mavjudligini tekshirish imkoniyati ham mavjud. Python dasturlash tilida elementlari tarkibidan, boshqa ro'yxat elementlari mavjudligini va ro'yxatlar ustida amallarni bajarishni quyidagi dastur orqali qarab chiqamiz.

```
>>> a
```

```
[7, 8, 3, 2, 5, 6, 1, 4, 10, 9]
```

```
>>> b=[0,-2,-1]
```

```
>>> b
```

```
[0, -2, -1]
```

```
>>> b in a
```

```
False
```

```
>>> 7 in a
```

```
True
```

```
>>> a+b
```

```
[7, 8, 3, 2, 5, 6, 1, 4, 10, 9, 0, -2, -1]
```

```
>>> b*2
```

```
[0, -2, -1, 0, -2, -1]
```

```
>>>
```

Python dasturlash tilida ro'yxat elementlari uchun quyidagi ko'rinishdagi protseduralar aniqlangan.

L.append(x) – L ro'yxatning oxiriga x elementni qo'shish

L.extend(T) - L ro'yxatning oxiriga T ro'yxatni qo'shish

L.insert(i,x) - L ro'yxatning i- o'rniga x elementga qo'shish

L.pop(i) - L ro'yxatning i-o'rnidagi elementni uchirish

L.remove(x) - L ro'yxatning x elementni uchirish

L.count(x) - L ro'yxat ichida x elementlar sonini aniqlash

L.reverse() - L ro'yxat elementlarini kamayish tartibida saralash

L.sort() - L ro'yxat elementlarini o'ssh tartibida saralash

Yuqoridagi protseduralarni ishlash jarayonini aniqlash uchun quyidagi dasturga e'tibor bering.

```
>>> b=[0,-2,-1,5]
>>> b
[0, -2, -1, 5]
>>> b.append('a')
>>> b
[0, -2, -1, 5, 'a']
>>> c=[7,9]
>>> b.extend(c)
>>> b
[0, -2, -1, 5, 'a', 7, 9]
>>> b.insert(2,'f')
>>> b
[0, -2, 'f', -1, 5, 'a', 7, 9]
>>> b.pop(0)
0
>>> b
[-2, 'f', -1, 5, 'a', 7, 9]
>>> b.remove(7)
>>> b
[-2, 'f', -1, 5, 'a', 9]
>>> b.count('12')
0
>>> b
[-2, 'f', -1, 5, 'a', 9]
>>> b.count('f')
1
```

```

>>> b.reverse()
>>> b
[9, 'a', 5, -1, 'f', -2]
>>> a=[1,3,2]
>>> a.sort()
>>> a
[1, 2, 3]
>>>

```

Misol. 10 ta butun sonli elementdan tashkil topgan ro'yxat hosil qilib, elementlarini ikkiga ko'paytirib ekranga chiqaring.

```

>>> A = list( range(0 , 10 , 2))
>>> A
[0, 2, 4, 6, 8]
>>> 2*A
[0, 2, 4, 6, 8, 0, 2, 4, 6, 8]
>>>

```

Misol. n ta butun sonli elementdan tashkil topgan ro'yxat hosil qilib, ro'yxatning juft elementlarini ikkiga ko'paytirib toq elementlarini 3ga ko'paytirib ekranga chiqaring.

```

n=int(input('n='))
a=[int(input()) for i in range(n)]
for i in range(n):
    if a[i]%2==0:
        a[i]=a[i]*2
    else:
        a[i]=a[i]*3
print(a)
===== RESTART: C:/Users/User/Desktop/1.py
=====
n=5

```

```
2
9
7
1
8
[4, 27, 21, 3, 16]
>>>
```

Kortejlar

Python dasturlash tilida kortej degan tushuncha mavjud, bunda kortej ro'yxat elementlari sifatida qabul qilinadi, lekin uni elementlarini o'zgartirish mumkin emas. Kortejlar ro'yxatga qaraganda kamroq xotiradan foydalaniladi. Kortejni belgilashda kvadrat qavslar o'rniga aylana qavslar ishlatiladi. Kortejlar o'zgarishlarga ruxsat bermaydi, siz unga yangi element qo'sha olmaysiz, mavjud elementlarni o'chira yoki o'zgartira olmaysiz.

Kortejlardan foydalanishni quyidagi dastur orqali qarab chiqamiz.

```
>>> a=(1,2,3)
>>> a
(1, 2, 3)
>>> b=(5,6,7)
>>> a+b
(1, 2, 3, 5, 6, 7)
>>>
```

Ro'yxatlarni **tuple()** funksiyasi orqali kortejlarga aylantirish imkoniyati mavjud. **List()** funksiyasi orqali esa kortejlarni ro'yxatlarga aylantirish imkoniyati mavjud.

```
>>> a=(1,2,3)
>>> a
(1, 2, 3)
>>> b=list(a)
```

```
>>> b
[1, 2, 3]
>>> tuple(b)
(1, 2, 3)
>>>
```

Bir nechta elementlarni yagona o'zgaruvchi bilan boshqarish va qayta ishlash uchun ro'yxat va kortej orqali to'liq amalga oshirish mumkin.

Nazariy savollar

- 1 Murakkab tur deganda nimani tushunasiz?
- 2 Ro'yxat va uning vazifasi?
- 3 Ro'yxat elementlari ustida bajariladigan funksiyalar va ularning vazifalari?
- 4 Ro'yxat elementlari ustida bajariladigan protseduralar va ularning vazifalari?
- 5 Kortej va uning vazifasi?
- 6 Kortej elementlari ustida bajariladigan funksiyalar va ularning vazifalari?
- 7 List() va tuple() funksiyalarining vazifasi?

Mustaqil ishlash uchun topshiriqlar

1. Natural n hamda haqiqiy sonli $A(1:n)$ ro'yxat(kortej) berilgan. Uning o'rta arifmetik qiymatini toping.
2. Natural n hamda haqiqiy sonli $A(1:n)$ ro'yxat(kortej) berilgan bo'lsin. Uning eng kattasi elementini toping.
3. Natural n hamda haqiqiy sonli $A(1:n)$ ro'yxat(kortej) berilgan. Uning eng kichik elementi necha marta uchraydi ?
4. N butun soni va haqiqiy sonli $B(1:N)$ ro'yxat(kortej) berilgan. Uning tub elementlari orasida eng kattasini aniqlang.
5. Butun $a_1, a_2,$ va a_3 sonlari berilgan. Butun sonli $B(1:3)$ (kortej) elementlarini $b_i = a_i - 3a_i$ formula yordamida aniqlang. Bu jadvalning barcha

elementlari ko'paytmasini hisoblang.

6. *Natural m hamda haqiqiy sonli $A(1:m)$ ro'yxat berilgan bo'lsin. Shu ro'yxatning har bir eng katta va eng kichigi elementlar orasidagi sonlarni toping.*

4.2 PYTHON DASTURLASH TILIDA LUG'AT VA TO'PLAMLARDAN FOYDALANISH

Reja:

1. *Lug'atlar va ularning umumiy ko'rinishi;*
2. *Lug'atlarga oid dasturlar;*
3. *To'plamlar va ularning umumiy ko'rinishi;*
4. *To'plamga oid dasturlar;*

Tayanch so'zlar: *lug'at, to'plam, del, pop, set, add.*

Dastur yaratish vaqtida ma'lum bir qiymatlar bir biriga mos holda tenglik jarayonlari ikki va undan ortiq bo'lgan holatlarda dastur tuzuvchilarga bir muncha qiyinchiliklar tug'diradi. Agar ma'lum bir qiymatlar bir biriga mos holda tenglik jarayonlari ikkitadan ortiq holatlarni e'tiborga olish kerak bo'lgan jarayonlarni python dasturlash tilida hal etish imkoniyat mavjud.

Lug'atlar

Agar ma'lum bir qiymatlar bir biriga mos holda tenglik holatlarda, albatta, python dasturlash tilida **lug'at** turli operatoridan foydalanish maqsadga muvofiqdir. C++ dasturlash tilida bu jarayon tanlash ya'ni switch operatori orqali amalga oshirilardi.

Ta'rif: Agar ma'lum bir qiymatlar bir biriga mos holda tenglik jarayonlarini aks ettirsa bunday turlar lug'at turi deb ataladi.

Pythonda esa lug'atlar yuqorida ta'kidlangan masalaga javob beradi, python dasturlash tilida lug'atlarni faollashtirishning umumiy ko'rinishi quyidagicha bo'ladi.

<lug'at o'zgaruvchisi>={x:x1, y:y1, ...}

Lugʻatlarni shakllantirishda ikkita mos tenglik : belgisi bilan birlashtiriladi, bu qiymatlar vergul bilan ajratiladi. Lugʻat turi foydalanuvchiga maʼlum bir qiymatlar qanday boshqa qiymatlarga tengligini aniqlab berishga xizmat qiladi, lugʻat turi oʻz nomi bilan ham aytib turibdiki, maʼlum bir soʻzni boshqa soʻzga tengligini aniqlaydi

Misol. Hafta kunlariga mos lugʻat yarating va ixtiyoriy hafta kunini aniqlash dasturini yarating.

```
a={1:'dushanba',2:'seshanba',3:'chorshanba',4:'payshanba',5:'juma',
6:'shanba',7:'yakshanba'}
n=int(input('Hafta kuni raqamini kiriting='))
print(a[n])
===== RESTART: C:/Users/User/Desktop/1.py
=====
Hafta kuni raqamini kiriting=5
juma
>>>
```

Lugʻatlar uchun **del()**, **pop()** funksiyalari aniqlangan, lugʻatlarni tushunib olish uchun quyidagi dastur kodiga eʼtibor bering.

```
>>>a={ 1:'dushanba',2:'seshanba',3:'chorshanba',4:'payshanba',5:'juma',6:'
shanba',7:'yakshanba'}
>>> a
{1: 'dushanba', 2: 'seshanba', 3: 'chorshanba', 4: 'payshanba', 5: 'juma', 6:
'shanba', 7: 'yakshanba'}
>>> a[3]
'chorshanba'
>>> a.keys()
dict_keys([1, 2, 3, 4, 5, 6, 7])
>>> a.pop(1)
'dushanba'
>>> a
{2: 'seshanba', 3: 'chorshanba', 4: 'payshanba', 5: 'juma', 6: 'shanba', 7:
```

```
'yakshanba'}
>>>
>>>a={1:'dushanba',2:'seshanba',3:'chorshanba',4:'payshanba',5:'juma',6:'
shanba',7:'yakshanba'}
>>> a
{1: 'dushanba', 2: 'seshanba', 3: 'chorshanba', 4: 'payshanba', 5: 'juma', 6:
'shanba', 7: 'yakshanba'}
>>> 'Hafta' in a
False
>>>
```

Misol. O‘zbek va ingliz tillarida ranglar bo‘yicha lug‘at hosil qiling.

```
a={'oq':'white','qora':'black','yashil':'green'}
n=input('rangni o"zbek tilida yozing=')
print(a[n])
===== RESTART: C:/Users/User/Desktop/1.py
=====
rangni o"zbek tilida yozing=yashil
green
>>>
```

Yuqoridagi dasturlar shuni ko‘rsatadiki, python dasturlash tilida lug‘atlar mos tenglik bo‘yicha barcha masalalarni hal etish imkoniyati mavjud.

To‘plamlar

Matematika kursidan biz to‘plamlar tushunchasi bilan oldindan tanishmiz. To‘plam deganda, bir necha elementlarning majmuasi tushuniladi. Bu elementlar bir xil toifali, lekin tartiblanmagan bo‘ladi. Masalan, butun sonlar to‘plami, shakllar to‘plami, radiodetallar to‘plami va hokazo.

Python dasturlash tilida to‘plam ma’lumotlari matematika to‘plamlari bilan bir xil ravishda ishlaydi, ammo bunda ma’lumotlar ixtiyoriy turda bo‘lishi mumkin.

Tarif: Python dasturlash tilida ixtiyoriy turdagi cheklangan sondagi ma’lumotlarning betartib majmuasiga to‘plam deyiladi.

Python dasturlash tilida har bir shakllantirilgan to‘plamga nom beriladi. Python dasturlash tilida to‘plamlarni faollashtirishning umumiy ko‘rinishi quyidagicha bo‘ladi.

<to‘plam o‘zgaruvchisi>={x1, x2, ...}

To‘plamga kirgan ma’lumotlar to‘plam elementlari deb yuritiladi. Elementlar turi ixtiyoriy belgi bo‘lishi mumkin. Python dasturlash tilida to‘plamlarni faollashtirishni quyidagi dastur orqali ko‘rib chiqamiz.

```
>>> a={1,2,3}
>>> a
{1, 2, 3}
>>> b={1,2.3,'k',-8}
>>> b
{-8, 1, 2.3, 'k'}
>>>
```

Demak python dasturlash tilida to‘plam elementlari {} qavslar bilan shakllantiriladi va elementlar vergul bilan ajratiladi.

To‘plamlar tarkibida **set()** funksiyasi aniqlangan, bu funksiya ro‘yxatlar va elementlarni to‘plamga aylantirishga xizmat qiladi. Set() funksiyasini ishlash jarayoni quyidagi dasturda keltirilgan.

```
>>> a=[2,3,'a']    # list
>>> a
[2, 3, 'a']
>>> set(a)        # toplam
{2, 3, 'a'}
```

```
>>> b=set(range(10))
>>> b
{0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
>>>
```

To‘plamlar ustida python dasturlash tilida quyidagi amallar aniqlangan.

s1.add(x) – s1 toplamga x elementni qo‘shish

s1.intersection(s2) – s1 va s2 to‘plam elementlari kesishmasini aniqlash

s1.union(s2) – s1 va s2 to‘plam elementlari birlashmasini aniqlash

s1-s2 bu s1 to‘plam elementlaridan s2 to‘plam elementlarini aniqlash

To‘plam elementlari ustida aniqlangan amallarning ishlash jarayoni quyidagi dasturda keltirilgan.

```
>>> a={1,2,3,'a',5}
>>> a
{1, 2, 3, 5, 'a'}
>>> a.add('c')
>>> a
{1, 2, 3, 5, 'a', 'c'}
>>> a={1,2,3,5}
>>> b={4,5}
>>> a.union(b)
{1, 2, 3, 4, 5}
>>> b.union(a)
{1, 2, 3, 4, 5}
>>> a.intersection(b)
{5}
>>> b.intersection(a)
{5}
>>> a-b
{1, 2, 3}
>>> b-a
```

Nazariy savollar

- 1 Python tilida lug‘at deganda nimani tushinasiz?*
- 2 Lug‘at qanday vazifalarni amalga oshiradi?*
- 3 Lug‘atni shakllantirishning umumiy ko‘rinishi?*
- 4 Lug‘at ustida qanday amallar mavjud?*
- 5 Python tilida to‘plam deganda nimani tushinasiz?*
- 6 To‘plam qanday vazifalarni amalga oshiradi?*
- 7 To‘plamni shakllantirishning umumiy ko‘rinishi?*
- 8 To‘plam ustida qanday amallar mavjud?*

Mustaqil ishlash uchun topshiriqlar

- 1. Ingiliz o‘zbek tilidagi lug‘at yarating.*
- 2. Har bir son va uning bo‘luvchilar soni bo‘yicha lug‘at yarating.*
- 3. Har bir son va uning bo‘luvchilar yig‘indisi bo‘yicha lug‘at yarating.*
- 4. Ikkita to‘plam yarating va u to‘plamlarning birlashmasini toping.*
- 5. Ikkita to‘plam yarating va u to‘plamlarning kesishmasini toping.*
- 6. Ikkita to‘plam yarating va u to‘plamlarning bir birining qism to‘plami ekanligini aniqlang.*

4.3 PYTHON DASTURLASH TILIDA MASSIVLAR VA ULARDAN FOYDALANISH

Reja:

- 1. Bir o‘lchovli massivlar;*
- 2. Bir o‘lchovli massivlarga oid dasturlar;*
- 3. Massiv elementlari ustida aniqlangan amallar;*
- 3. Ikki o‘lchovli massivlar;*
- 4. Ikki o‘lchovli massivlarga oid dasturlar;*
- 5. Random funksiyasi?*

Tayanch soʻzlar: *massiv, bir oʻlchovli massiv, ikki oʻlchovli massiv, random, numpy.*

Python dasturlash tilida kompyuter xotirasiga bir oʻzgaruvchi yordamida bir nechta qiymatlarda foydalanish boʻyicha bir nechta turlarni qarab oʻtdik. Bir oʻzgaruvchi bilan bir nechta qiymatlarni roʻyxat, kortej, toʻplam va satr turlar bilan amalga oshirish mumkin. Python dasturlash tilida bir oʻzgaruvchi yordamida bir nechta qiymatlardan foydalanish uchun massiv degan turdan ham foydalanish imkoniyati mavjud.

Bu qismda dasturdagi maʼlumot strukturalari bilan tanishishni boshlaymiz. Dasturda ikki asosiy tur maʼlumot strukturalari mavjuddir. Birinchisi statik, ikkinchisi dinamikdir. Statik deganimizda xotirada egallagan joyi oʻzgarmas, dastur boshida beriladigan strukturalarni nazarda tutamiz. Dinamik maʼlumot tiplari dastur davomida oʻz hajmini, egallagan xotirasini oʻzgartirishi mumkin. Massivlar hotirada ketma-ket joylashgan, maʼlumotlar guruhidir. Alohida bir oʻzgaruvchini koʻrsatish uchun massiv nomi va kerakli oʻzgaruvchi indeksini yoziladi.

Taʼrif: Bir turga mansub boʻlgan yagona nom bilan saqlanuvchi tartiblangan maʼlumotlar majmuasi massiv deyiladi.

Massivlar yagona oʻzgaruvchi bilan kompyuter xotirasiga saqlanadi, uning elementlari maʼlum bir indekslar bilan tartiblab joylashtiriladi. Bu qonuniyat roʻyxatlar uchun ham oʻrinli boʻladi.

Massivlar yagona nom bilan bir nechta qiymatni oʻzida mujassamlashtiradi, bularga matematikadagi vektorlarni misol keltirish mumkin. Vektor ham yagona nom bilan saqlanib uning tarkibida bir nechta qiymatni oʻzida mujassamlashtiradi. Vektorning ham elementlari bir turga mansub va tartiblangan boʻladi.

Massivlar holatiga koʻra ikki turga boʻlinadi.

- Bir oʻlchovli massivlar;
- Ikki oʻlchovli massivlar;

Bir oʻlchovli massivlar maʼlumotlarni bir satri koʻrinishda saqlansa, ikki oʻlchovli massivlar esa maʼlumotlarni satrlar satri koʻrinishida saqlaydi.

Python dasturlash tilida massivlarni qayta ishlash uchun **numpy** kutubxonasini faollashtirish kerak. Python 3.7 versiyasining IDLE rejimi tarkibida numpy kutubxonasi mavjud emas. Python dasturlash tili tarkibiga

numpy kutubxonasi yuklashning bir nechta turlari mavjud, shulardan bittasi **Anoconda** dasturini kompyuterga oʻrnatib bu dasturni **Spyder(python 3.7)** tizimini yuklash kerak. **Spyder(python 3.7)** tizimi tarkibida python dasturlash tilining barcha kutubxonalar mavjud. Dastur tuzishdan oldin albatta **numpy numpy** kutubxonasi faollashtirish kerak.

Bir oʻlchovli massivlar

Odatda massivlar zarurat, katta hajmdagi tartiblangan, lekin chekli elementlarga oid masalalarni hal etishda yuzaga keladi. Massivning ixtiyoriy elementiga murojat qilish uchun uning indeks nomeri boʻyicha murojat qilinadi. Bir oʻlchovli massivlarni python dasturlash tilida faollashtirish mumkin. Bir oʻlchovli massivlarni boshlangʻich qiymatlari bergan holda python dasturlash tilida quyidagicha faollashtiriladi.

<massiv oʻzgaruvchisi> = array([x1, x2, ...])

Massivni python dasturlash tilida faollashtirish uchun, elementlar soni yoki massiv elementlarining boshlangʻich qiymatlari berilishi shart emas.

Python dasturlash tilida bir oʻlchovli massivni faollashtirish quyidagicha amalga oshiriladi.

```
from numpy import*  
a=array([1,2,8])  
print(type(a))
```

Natija

```
<class 'numpy.ndarray'>
```

Massivni umumiy koʻrinishida birinchi massiv oʻzgaruvchisi va massiv elementlari yoziladi. Massiv elementlari, ixtiyoriy turdagi sondan iborat boʻlishi mumkin. Lekin massiv elementlari tarkibida bitta haqiqiy son boʻlsa boshqa elementlari ham haqiqiy deb qaraladi.

Massiv elementlari soni biror bir ifoda yoki yagona oʻzgaruvchi boʻlishi mumkin, bitta oʻzgaruvchi orqali massivning umumiy indekslarini

ifodalash mumkin. Massiv elementlarini ustida amallar bajarishni quyidagi masala orqali qaraymiz. Massiv elementlarini tartib nomeri doimo 0 dan boshlanadi.

Misol: 10 ta elementdan tashkil topgan massiv elementlarining juft elementlarini ikkiga toq elementlarini to'rtga ko'paytirib ekranga chiqaring.

```
from numpy import*
a=array([1,2,3,4,5,6,7,8,9,10])
for i in range(10):
    if a[i]%2==0:
        a[i]=2*a[i]
    else:
        a[i]=4*a[i]
print(a)
```

Natija

```
[ 4  4 12  8 20 12 28 16 36 20]
```

Python dasturlash tilida massiv elementlari boshlang'ich qiymatlarini bermasdan faollashtirishh quyidagicha.

<massiv o'zgaruvchisi> = array([])

Bunda massiv elementlari kvadrat qavs ichida yozilmaydi ya'ni bo'sh holda qoldiriladi, bunda massiv elementsiz shakllanadi.

```
from numpy import*
a=array([])
print(a)
```

Natija

```
[]
```


Massiv elementlar soni uning tarkibidagi qiymatlar orqali aniqlanadi va massivning elementlari doimo nolinci tartibdan boshlanadi.

Massiv elementlarini funksiyalar orqali va klaviatura yordamida hosil qilish

Massiv elementlarini python dasturlash tilida ma'lum bir ketma ketlik yoki funksiya orqali hosil qilish imkoniyati mavjud. Bunda massiv elementlari berilgan funksiyaning qiymatlarini qabul qiladi. Range() funksiyasi orqali massiv elementlarini shakllantirishni quyidagi dastur orqali hal etamiz.

```
from numpy import*
L=range(10)
a=array(L)
print(a)
```

Natija

```
[0 1 2 3 4 5 6 7 8 9]
```

Haqiqiy sonlarni arange() funksiyasi orqali quyidagicha massiv elementlariga ta'minlash mumkin.

```
from numpy import*
from math import*
L=arange(0, 2*pi, pi/6)
S=array(L)
for i in range(len(L)):
    print(S[i])
```

Natija

```
0.0
```

```
0.5235987755982988
```

```
1.0471975511965976
```

```
1.5707963267948966
```

```
2.0943951023931953
```

```
2.617993877991494
3.141592653589793
3.665191429188092
4.1887902047863905
4.71238898038469
5.235987755982988
5.759586531581287
```

Massiv elementlarini doimo statik bo‘lavermaydi, masala sharoitiga qarab dinamik ham bo‘lishi mumkin. Massiv elementlari dinamik bo‘lganda, elementlarni klaviaturadan kiritishga to‘g‘ri keladi. Klaviaturadan kiritishni $a=array([int(input()) for i in range(n)])$ kabi amalga oshiramiz bu jarayonni quyidagi dastur orqali tushinib olamiz.

Misol: n natural son va n ta elementdan tashkil topgan massiv berilgan uning eng katta elementi va o‘rnini aniqlash dasturini tuzing.

Bu masalani yechimini aniqlash uchun birinchi elementni eng katta deb qarab, massivning barcha elementini eng katta deb qaralgan element bilan solishtiriladi, agar solishtirilayotgan sondan kattasi topilsa, u katta bilan almashtiriladi.

```
from numpy import*
n=int(input('n='))
a=array([int(input()) for i in range(n)])
max=a[0]      # eng katta element
l=0           # eng katta element o'rne
for i in range(n):
    if a[i]>max:
        max=a[i]
        l=i
print('max=', max)
print('l=', l+1)
```

Natija
n=5

```
8
11
-5
6
9
max= 11
l= 2
```

Yuqoridagi dasturda massiv elementlari klaviaturadan kiritilishi shakllantirilgan. Massivning eng katta elementi va uning o'ri aniqlanish jarayonida o'zgaruvchi l maxning o'ri, massiv 0-tartibdan boshlanganligi sababli $l+1$ ekranga chiqarildi.

Massiv elementlari ustida aniqlangan amallar

Massivning elementlari ustida, boshqa, ya'ni ro'yxat, to'plam elementlari kabi ba'zi bir amallar aniqlangan. Massivning har bir elementiga murojaat qilish natijasida amallar bajarish jarayoni python tilida soddalashtirilgan holatga keltirilgan, ya'ni amallar massiv nomiga yozilish natijasida uning barcha elementlariga ta'sir ettiradi.

Massiv elementlari ustida quyidagi amallar aniqlangan.

a.sum() – *a* massiv elementlarining yig'indisini hisoblash

a.mean() – *a* massiv elementlarining o'rta arifmetik qiymatini hisoblash

a.max() – *a* massiv elementlarining maksimumini hisoblash

a.min() – *a* massiv elementlarining minimumini hisoblash

Yuqoridagi amal funksiyalarni ishlash jarayonini quyidagi dasturda qarab o'tamiz.

```
from numpy import*
n=int(input('n='))
a=array([int(input()) for i in range(n)])
s=a.sum()
print('sum=',s)
```

```
t=a.mean()
print('o"rta arifmetik=',t)
max=a.max()
print('max=',max)
min=a.min()
print('min=',min)
```

Natija

```
n=3
1
3
2
sum= 6
o"rta arifmetik= 2.0
max= 3
min= 1
```

Massiv elementlari ustida arifmetik amallar ham aniqlangan, massiv elementlari ustida arifmetik amallarni quyidagi dastur orqali qarab o‘tamiz.

```
from numpy import*
a=array([1,4,5,6])
b=array([2,3,1,4])
print('a+b=',a+b)
print('a*b=',a*b)
print('a-b=',a-b)
print('a/b=',a/b)
```

Natija

```
a+b= [ 3  7  6 10]
a*b= [ 2 12  5 24]
a-b= [-1  1  4  2]
a/b= [0.5  1.33333333  5.  1.5 ]
```

Yuqoridagi dasturga e'tibor qaratsak, massiv elementlari ustida arifmetik amallar, massivning mos elementlari ustida bajarilmoqda.

Ikki o'lchovli massivlar

Python dasturlash tilida ba'zi hollarda bir nechta o'lchamlari va turi bir xil bo'lgan, bir o'lchovli massivlardan foydalanishga to'g'ri keladi. Bir nechta bir o'lchovli massivlarni birlashtirish natijasida ikki o'lchovli massivlarni hosil qilish mumkin. Ikki o'lchovli massivlarni tarkibida ma'lumotlar satrlarning satri ko'rinishida tasvirlanadi. Ikki o'lchovli massivlarning tarkibi ham bir o'lchovli massivlar kabi tartiblangan bo'ladi.

Ikki o'lchovli massivlarga matematikadagi matritsalar misol bo'lishi mumkin. Ikki o'lchovli massivlar tarkibidagi elementlar xuddi matritsani elementlari kabi tasvirlanadi.

Ta'rif: Bir turga mansub bo'lgan yagona nom bilan saqlangan matritsa ko'rinishdagi tartiblangan ma'lumotlar majmuasi ikki o'lchovli massivlar deyiladi.

Ikki o'lchovli massivning barcha elementlari aniq turga mansub bo'ladi va uning elementlari bir nechta satrlar ko'rinishda bo'ladi. Ikki o'lchovli massivlar quyidagi shaklda bo'ladi.

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ & \cdot & \cdot & \cdot \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix}$$

Yuqoridagi shakldan ko'rinib turibdiki ikki o'lchovli massiv bir o'lchovli massivlarning bir nechtasi yoki matritsa ko'rinishida tasvirlanar ekan. Ikki o'lchovli massivlarning kompyuter xotirasiga har bir satr uchun alohida tartib nomer ya'ni indeks bilan saqlanadi. Ikki o'lchovli massivlarning har bir elementiga o'zining indeksi bo'yicha murojat qilinadi.

Ikki o'lchovli massivlarni python dasturlash tilida faollashtirishning umumiy ko'rinishi quyidagicha.

<massiv o'zgaruvchisi> = array([x1, x2, ...],[x1, x2, ...], ...])

Ikki o‘lchovli massivlarni python dasturlash tilida faollashtirish uchun massiv o‘zgaruvchisi, massiv va uning elementlar kvadrat qavslar ketma ketligida yoziladi. Agar $a[2,3]$ massiv berilgan bo‘lsa, 2 – bu satr tartibi, 3 – esa ustun tartibi hisoblanadi. Ikki o‘lchovli massivlarni python dasturlash tilida faollashtirish quyidagicha.

```
a=array([[1,2],[4,5]])  
print(a)
```

Natija

```
[[1 2]  
 [4 5]]
```

Massiv elementlari sonini, komplyator massiv elementlarining boshlang‘ich qiymatlarga qarab aniqlaydi. Massiv elementlarini ikkiga ko‘paytirib ekranga chiqarish jarayoni quyidagicha.

Misol. $A(2,2)$ massiv berilgan uning elementlarini ikkiga ko‘paytirib ekranga chiqarish dasturini tuzing.

```
from numpy import*  
a=array([[1,2],[4,5]])  
print(a*2)
```

Natija

```
[[ 2  4]  
 [ 8 10]]
```

Ikki o‘lchovli massivlar yordamida python dasturlash tilida matematikani balki boshqa sohalarning ham bir necha masalalarini hal etish mumkin.

Misol. $A(n,n)$ matritsa berilgan uning juft elementlarini ikkiga, toq elementlarini to‘rtga ko‘paytirib ekranga chiqarish dasturini tuzing.

```
from numpy import*
```

```

n=2
a=array([[int(input()) for i in range(n)],[int(input()) for i in
range(n]])
for i in range(n):
    for j in range(n):
        if a[i,j]%2==0:
            a[i,j]=2*a[i,j]
        else:
            a[i,j]=4*a[i,j]
print('a=',a)

```

Natija

```

1
2
3
4
a= [[ 4  4]
     [12  8]]

```

Ikki o‘lchovli massiv elementlarining har birini tekshirish yoki ular ustida amallar bajarish uchun ichma ich sikllardan foydalaniladi.

Bir o‘lchovli massivlar uchun aniqlangan barcha amal va funksiyalar, ikki o‘lchovli massivlar uchun ham o‘rinli hisoblanadi. Quyidagi dastur yuqoridagi fikrni isbotlaydi.

```

from numpy import*
a=array([[4,5],[1,2]])
b=array([[1,2],[2,1]])
y=a+b
print(y)
print('sum a=',y.sum())

```

Natija

```

[[5 7]

```

```
[3 3]
```

```
sum a= 18
```

Random funksiyasi

Massiv elementlarini ixtiyoriy tasodifiy sonlar bilan to'ldirish uchun python dasturlash tilida imkoniyat yaratilgan. Agar massiv elementlarini tasodifiy sonlar bilan to'ldirish kerak bo'lsa, tasodifiy sonlar bilan ishlash funksiyasiga murojat qilish kerak. Python dasturlash tilida tasodifiy sonlarni hosil qilishni **random** funksiyasi va uning bir nechta kutubxonalari amalga oshiradi. $[a,b]$ oraliqda n ta sonni tasodifiy tanlash random funksiyasining umumiy ko'rinishi quyidagicha bo'ladi.

Random.randint(a,b,n)

Random funksiyasining vazifasi biror bir o'zgaruvchiga yoki massiv elementlariga tasodifiy sonni o'zlashtirish uchun xizmat qiladi.

Misol: $A(10)$ massiv elementlarini tasodifiy sonlar yordamida hosil qilib uning juft elementlarini ikkiga ko'paytirib ekranga chiqaring.

```
from numpy import*
a=random.randint(1,20,10)
print('a massivning dastlabki holati=',a)
for i in range(10):
    if a[i]%2==0:
        a[i]=2*a[i]
print('a ning natija holati=',a)
```

Dastur natijasi

```
a massivning dastlabki holati= [19 13 12 15 9 9 11 2 14 12]
```

```
a ning natija holati= [19 13 24 15 9 9 11 4 28 24]
```

Demak, masala yechimiga e'tibor qaratsak, $[1,20]$ oraliqdagi massivning tasodifiy sonlardan iborat 10 ta elementlari ichidan faqat

to'rttasi juft son ekan. Random funksiyasining **random.uniform(a,b,n)** va **random.normal(a,b,n)** funksiyalari ham mavjud.

Mavzularni o'quvchi tez o'zlashtirish maqsadida nisbatan oson dasturlar tanlanib olindi. Qiyinlik darajasi yuqori bo'lgan dasturlarni yaratish uchun sintaksikani o'rni yo'q, unda faqat algoritmikani o'rni bor.

Nazariy savollar

- 1 Murakkab tur deganda nimani tushunasiz?
- 2 Massiv deb nimaga aytiladi?
- 3 Massiv nechta turga bo'linadi?
- 4 Bir o'lchovli massiv deb nimaga aytiladi?
- 5 Bir o'lchovli massivning umumiy ko'rinishi?
- 6 Bir o'lchovli massivning faollashtirish usullari?
- 7 Ikki o'lchovli massivning umumiy ko'rinishi?
- 8 Ikki o'lchovli massivning faollashtirish usullari?
- 9 Massiv elementlarining shakllantirishni(klaviaturadan) tushintirib bering?
- 10 Massivlar ustida aniqlangan amal va funksiyalar vazifalari?
- 11 Random funksiyasi va uning umumiy ko'rinishi?
- 12 Random funksiyasining vazifasi?

Mustaqil ishlash uchun topshiriqlar

1. n natural soni berilgan. Datslabki n ta Fibonachchi sonlaridan tashkil topgan massivni hosil qiling va elementlarini chiqaring.

$$F_0 = 1; F_1 = 1; F[k] = F[k-1] + F[k-2]; k=2, 3, 4, \dots$$

2. n natural soni va A, B butun sonlari berilgan ($n > 2$). $a[0] = A$; $a[1] = B$; boshqa elementlari o'zidan oldingi barcha elementlari yig'indisiga teng bo'lgan massivni hosil qiling va elementlarini chiqaring.

3. n ta elementdan tashkil topgan massiv berilgan. Uning elementlari teskari tartibda chiqaruvchi dastur tuzilsin.

4. n ta elementdan tashkil topgan massiv berilgan. Dastlab massiv elementlari orasidan juftlarini indeksleri o'sish tartibida chiqaruvchi, keyin massiv elementlari orasidan toqlarini indeksleri kamayish tartibida chiqaruvchi dastur tuzilsin.

Massiv elementlar: 4 5 7 8 6 9

Natija: 4 6 8 9 7 5

5. n ta elementdan tashkil topgan massiv va K butun soni berilgan ($1 \leq K < n$). Massiv elementlari orasidan indeksi K ga karralilarini chiqaruvchi dastur tuzilsin. $A_k, A_{2k}, A_{3k}, \dots$ Shart operatori ishlatilmasin.

6. n ta elementdan tashkil topgan massiv berilgan (n juft son). Massiv elementlari orasidan quyidagilarni chiqaruvchi dastur tuzilsin. $A[0], A[2], A[4], \dots$ Shart operatori ishlatilmasin.

7. n ta elementdan tashkil topgan massiv berilgan (n toq son). Massiv elementlari orasidan quyidagilarni chiqaruvchi dastur tuzilsin. $A[n-1], A[n-3], \dots, A[1]$. Shart operatori ishlatilmasin.

8. Natural n va m hamda haqiqiy sonli $A(1:n, 1:m)$ massiv berilgan. Uning o'rtacha arifmetik qiymati, max va minlarni toping.

9. Butun $a_1, a_2, va a_3$ sonlari berilgan. Butun sonli $B(1:3, 1:3)$ jadval elementlarini $b_{i,j} = a_i - 3a_j$ formula yordamida aniqlang. Bu jadvalning barcha elementlari ko'paytmasini hisoblang.

10. Butun a_1, a_2, a_3 va a_4 sonlari berilgan bo'lsin. Butun sonli $B(1:4, 1:4)$ jadvalning elementlari

$$b_{ij} = \frac{2a_i - 3a_j}{i + j}$$

formula yordamida aniqlanadi. Bu jadvalning eng katta va eng kichik elementlarining tartib raqamlarini aniqlang.

11. Natural n va m hamda haqiqiy sonli $A(1:n, 1:m)$ massiv berilgan bo'lsin. Shumassivning har bir satridagi eng katta elementlar ichida eng kichigini toping.

12. n butun soni va haqiqiy sonli $B(1:n, 1:n)$ massiv berilgan. Uning bosh va qarama-qarshi diagonallaridagi elementlar yig'indisini hisoblang.

13. Natural n va m hamda haqiqiy sonli $A(1:n, 1:m)$ massiv berilgan bo'lsin. Shu massivning har bir ustunidagi eng kichik elementlar ichida eng kattasini toping.

14. Natural n va m hamda haqiqiy sonli $A(1:n, 1:m)$ massiv berilgan bo'lsin. Shu massivning birinchi elementlari musbat bo'lgan ustunlaridagi elementlarning yig'indisini hisoblansin.

15. Natural n va m hamda haqiqiy sonli $A(1:n, 1:m)$ massiv berilgan bo'lsin. Shu massivning birinchi elementlari musbat bo'lgan satrlardagi elementlarning ko'paytmasini toping

16. Butun sonli $A(1:10, 1:20)$ massivda necha xil elementlar uchrashini aniqlang.

17. Butun sonli $A(1:10, 1:10)$ massivda bir xil elementlar mavjud yoki mavjud emasligini aniqlang.

18. n butun soni va haqiqiy sonli $B(1:n, 1:n)$ massiv berilgan. Hisoblang: $x_1|y_1 + \dots + x_n|y_n$. Bu yerda x_i - B ning i -satriidagi eng katta element, y_j -esa B ning j ustunidagi eng kichik element.

19. n natural soni berilgan bo'lsin. $A(1:n, 1:n)$ haqiqiy sonli jadval elementlarini

$$a_{ij} = \begin{cases} \sin(i + j) & , \text{agar } i < j \\ 1 & , \text{agar } i = j \\ \text{ctg} \left(i + \frac{j}{2i} + j \right) & , \text{agar } i > j \end{cases}$$

formula yordamida aniqlang.

20. Natural n va m hamda haqiqiy sonli $A(1:n, 1:m)$ massiv berilgan. Shu massivning eng katta va eng kichik elementlari joylashgan satrlari o'rinlarini almashtiring.

21. Natural n va m hamda haqiqiy sonli $A(1:n, 1:m)$ massiv berilgan. Shu massivning eng katta va eng kichik elementlari joylashgan ustunlari o'rinlarini almashtiring.

22. n tartibli B kvadrat matrisa berilgan bo'lsin. Unda hamma elementlari juft sonlardan iborat bo'lgan ustun mavjudmi ?

23. n tartibli B kvadrat matrisa berilgan bo'lsin. Unda hamma elementlari bir xil bo'lgan satrlar qancha ?

24. m tartibli A kvadrat matrisa berilgan. Unda ikki qo'shni elementlarning o'rta geometrik quymatiga teng bo'lgan elementlar joylashgan satrlarning tartib raqamlarini aniqlang.

28. n tartibli B kvadrat matrisa berilgan. Unda elementlari simmetrik usulda joylashgan ustunlar mavjudmi ?

29. n natural, x haqiqiy sonlar hamda $A(1:n,1:n)$ haqiqiy sonli jadval berilgan bo'lsin. $B(n)$ - bir o'lchovli jadval elementlarini aniqlang. Bu yerda $b_i=1$, agar A ning i -satrida x dan katta bo'lgan elementlar mavjud bo'lmasa, aks holda $b_i=0$.

30. m tartibli A kvadrat matrisa berilgan. Unda elementlari o'sish tartibida joylashgan satr mavjudmi ?

31. m tartibli A kvadrat matrisa berilgan. Unda elementlari kamayish tartibida joylashgan biror ustun mavjudmi?

32. Natural n va m sonlar hamda $A(1:n,1:m)$ haqiqiy sonlar jadvali berilgan. Shu jadvalning har bir satridagi elementlarni o'sish tartibida tartiblang.

33. Natural n va m sonlari hamda $A(1:n, 1:m)$ haqiqiy sonlar jadvali berilgan. Bu jadvalning diagonal elementlarini o'sish tartibida tartiblang.

V-BOB. PYTHON DASTURLASH TILIDA SATRLAR VA FAYLLAR

Mazkur bobda python dasturlash tili tarkibida satrli ma'lumotlar va ularni qayta ishlash algoritmlari yoritilgan. Satrlar bo'yicha nazariy ma'lumotlar va amaliy dasturiy ta'minotlar keltirib o'tilgan. Satr turi bu dasturlash texnologiyasining asosiy turlaridan hisoblanadi, bunda barcha ma'lumotlar string turi asosida qayta ishlanish mumkin. Dasturlash asoslarida satr turi bilan ishlash bu fanning asosiy maqsadi hisoblanadi. Tashkilot va muassasalarning axborotlarini qayta ishlash vaqtida uning tarkibida asosan satr turlari bilan ishlashga to'g'ri keladi. Bu bobda satrlar ustida satrning uzunligi, satrni nusxalash, satrni kerakli joyga o'rnatish kabi bir qator vazifalar keng qarab o'tilgan.

5.1 PYTHON DASTURLASH TILIDA SATRLAR VA ULARDAN FOYDALANISH

Reja:

1. *Satrlar;*
2. *Satrlar ustida aniqlangan amallar;*
3. *Satrlarga oid dasturlar.*

Tayanch so'zlar: *Satr, ord, chr, str, asciiz.*

Satrlar

Python dasturlash tilida satr turi va ulardan foydalanish usullari va tamoyillarini qarab o'tamiz. Satr turi bu belgilardan tashkil topgan ma'lumotlar majmuasi hisoblanadi. Satrlarning uzunligini aniqlash, satrlarni nusxalash, satrlarni kerakli joyga o'rnatish kabi amallarni bajarish uchun Python dasturlash tilida keng imkoniyatlar mavjud. Python dasturlash tilida satrlar ham boshqa turlar kabi oldin dastur tarkibida e'lon qilinishi ta'lab etilmaydi. Python dasturlash tilida ixtiyoriy tur ma'lumotlaridan foydalanish va ular ustida amallar bajarish mumkin, dasturlash muhitida boshqa turlar kabi satrli turlar ham mavjud. Python dasturlash tilida satrlarni belgilar massivi sifatida ishlatish imkoniyati mavjud.

Ta'rif: Alohida nom bilan saqlanuvchi bir nechta belgilardan tashkil topgan belgilar majmuasi satr deyiladi.

Satrlarni belgilar massivi sifatida qarab ular ustida amallar bajarish mumkin. Python dasturlash tilida satrlar nol('\0') terminatori bilan tugaydi. Nol terminatori bilan tugaydigan satrlar ASCIIZ –satrlari deyiladi.

Satrlardan foydalanish va ular ustida amallar bajarish uchun albatta oldin ularni shakllantirish kerak. Satr turiga mansub bo'lmagan o'zgaruvchilarni **str()** funksiyasi orqali satr turiga aylantiriladi. Satrlar, belgilar massivi bo'lganligi sababli uning elementlariga indeks bo'yicha murojaat qilinadi. Satrlarni shakllantirish uchun quyidagi dasturlarga etibor bering.

```
>>> s='fakultet'
>>> s
'fakultet'
>>> s[1]
'a'
>>> s[0]
'f'
>>>
```

Python dasturlash tilida satrlarning elementlari 0-tartibdan boshlanadi va satr tarkibida ixtiyoriy turdagi belgilarni qayta ishlash imkoniyati mavjud. Dasturlash texnologiyalarida shunday holatlar bo'ladiki, bunda sonli ma'lumotlar va satri ma'lumotlarni bir biriga almashtirish kerak bo'ladi. Bunday almashtirishlarni **int()**, **float()** va **str()** funksiyalari amalga oshiradi bu funksiyalarni qo'llanmaning I-boboda keltirib o'tganmiz. **int()**, **float()** va **str()** funksiyalarni ishlash jarayoni quyidagi dasturda keltirilgan.

```
>>> s='abs12fd'
>>> s[4]
'2'
>>> s[-1]
'd'
>>> int(s[4])
```

```
2
>>> int(s[4])+int(s[3])
3
>>> y=int(s[4])+int(s[3])
>>> str(y)
'3'
>>>
```

Satr elementlariga teskaridan ham murojaat qilish imkoniyati mavjud, bunda indeks manfiy tartibda beriladi. S[-1] – bu eng oxirgi element hisoblanadi.

Python dasturlash tilida har bir belgining maxsus kodi(key)ni va bu kodga mos belgini aniqlash imkoniyati mavjud.

ord() – bu funksiya ixtiyoriy belgini maxsus kodini aniqlaydi

chr() – bu funksiya maxsus kodga mos belgini aniqlaydi

ord() va chr() funksiyalarni ishlash jarayoni quyidagi dasturda keltirilgan.

```
>>> ord('a')
97
>>> chr(15)
'\x0f'
>>> chr(105)
'i'
>>>
```

Satrlar ustida aniqlangan amallar

Satrlar ma'lumotlar bo'yicha python dasturlash tilida bir qancha amallar aniqlangan. Bu amallarni ishlash jarayoni va ulardan foydalanishni qarab o'tamiz.

s1+s2	S1 va s2 satrlarni bir biriga ulaydi
s1*s2	S1 satr ma'lumotlarini ikki marta bir biriga ulaydi

s1[n:m]	S1 satr elementlarini n-dan boshlab m-gacha ajratib oladi
s1[n:m:x]	S1 satr elementlarini n-dan boshlab m-gacha x tasini ajratib oladi
s1 in s2	S2 satr tarkibida s1 satr borligini aniqlaydi
max(s1)	S1 satr tarkibida eng katta kodga ega elementni aniqlaydi
min(s1)	S1 satr tarkibida eng kichik kodga ega elementni aniqlaydi
len(s1)	S1 satr elementlar uzunligi ya'ni sonini aniqlaydi

Yuqoridagi funksiyalarni ishlash jarayoni quyidagi dasturda keltirilgan.

```

>>> s1='abs12'
>>> s1
'abs12'
>>> s2='a1'
>>> s1+s2
'abs12a1'
>>> s1*3
'abs12abs12abs12'
>>> s1[2:3]
's'
>>> s1[2:5]
's12'
>>> s2 in s1
False
>>> s1[2:6:2]
's2'
>>> 'ab' in s1
True
>>> max(s1)

```



```
's'
>>> min(s1)
'1'
>>> ord(max(s1))
115
>>> ord(min(s1))
49
>>> len(s1)
5
```

Python dasturlash tilida string turdagi satrlar ustida bir nechta qo‘shimcha amallar bajarish mumkin. Python dasturlash tilida string satrlari ustida quyidagi qo‘shimcha amallarni bajarish mumkin.

Amal	Mazmuni	Misol
=,+=	Qiymat berish amali	S="dastur"; s+="absd"; s1=s;
+	Satrlarni ulash	s=s+s1
==,!<,>,<=,>=	Satrlarni solishtirish amallari	s==s1 s>=s1 s==s1 and s<s1

Misol. *S* satr tarkibida *x* belgi sonini aniqlash dasturini tuzing.

```
s=input('s=')
x=input('x=')
n=len(s)
p=0
for i in range(n):
    if s[i]==x:
        p+=1
print(p)
===== RESTART: C:/Users/User/Desktop/1.py
=====
```

```
s=Maxmadiyoraka
```

```
x=a
```

```
4
```

```
>>>
```

Misol. *S satr tarkibida soʻzlar sonini aniqlash dasturini tuzing.*

```
s=input('s=')
```

```
n=len(s)
```

```
p=0
```

```
q=""
```

```
x=' '
```

```
i=0
```

```
while i<n:
```

```
    while i<n and s[i]!=x:
```

```
        q+=s[i]
```

```
        i=i+1
```

```
    if i!=0:
```

```
        if s[i-1]!=x:
```

```
            p=p+1
```

```
    i=i+1
```

```
    q=""
```

```
print(p)
```

```
===== RESTART: C:/Users/User/Desktop/1.py
```

```
=====
```

```
s= amaliy mat inf
```

```
3
```

```
>>>
```

Yuqoridagi dastur algoritmidagi har bir boʻsh joylar eʼtiborga olingan, agar faqat boʻsh joylar sonini eʼtiborga olinsa, soʻzlar orasida ikki va undan ortiq boʻsh joy kelganda dastur notoʻgʻri javob qaytaradi.

Misol. Berilgan satrda nechta raqam borligini aniqlash dasturini tuzing.

```
s=input('s=')
n=len(s)
p=0
a={'0','1','2','3','4','5','6','7','8','9'}
for i in range(n):
    if s[i] in a:
        p=p+1
print('Berilgan satrda=',p,' ta raqam bor')
===== RESTART: C:/Users/User/Desktop/1.py
=====
s=5as56ssd78
Berilgan satrda= 5 ta raqam bor
>>>
```

Yuqoridagi dasturda satr tarkibidagi raqamlar sonini aniqlash uchun a nomli to'plam yaratilib satrning har bir elementi a to'plamga tegishli ekanligini aniqlandi. Satr tarkibidan raqam izlanayotganligi sababli to'plam elementlari ya'ni raqamlar belgi sifatida qabul qilindi. Satrlar bo'yicha barcha turdagi dasturlar yuqorida keltirilgan nazariy va amaliy ma'lumotlarga asoslanib tuziladi.

Nazariy savollar

- 1 String turi deganda nimani tushunasiz?
- 2 String turidagi o'zgaruvchilarga boshlang'ich qiymatlar berish?
- 3 String turidagi o'zgaruvchilarni elementlariga murojat qilishni tushuntirib bering?
- 4 String turdagi satrlar ustida qanday amallar bajariladi va ularning yozilishi?
- 5 String turdagi satrlar ustida qanday funksiyalar bajariladi va ularning yozilishi?

Mustaqil ishlash uchun topshiriqlar

1. Nuqta bilan tugaydigan satr berilgan. Satrda nechta soʻz borligini hisoblab chiqing.
2. Ingliz matnidan iborat satr berilgan. b harfi bilan boshlanuvchi soʻz nechtaligini toping.
3. Satr berilgan. Unda r, k, t harflari qanchaligini hisoblab chiqing.
4. Satr berilgan. Unda *(yulduzcha), ;(nuqta vergul), :(ikki nuqta) belgilari qanchaligini hisoblab chiqing.
5. Matndan iborat satr berilgan. Eng qisqa va eng uzun soʻz uzunliklarini toping.
6. Orasida ikki nuqtasi boʻlgan belgili satr berilgan. Ungacha boʻlgan belgilar qanchaligini aniqlang.
7. Nuqta bilan tugaydigan, matndan tashkil topgan satr berilgan. Uchta harfdan iborat soʻzni ekranga chiqaring.
8. Berilgan satrdagi barcha na qisman satrni nad qisman satr bilan almashtirng.
9. Satr berilgan. Unda abc qisman satr necha marta uchrashini aniqlang.
10. Satr berilgan. Uning oxirgi soʻzidagi k harfi migʻdorini hisoblab chiqing.
11. Satr berilgan. Unda har xil belgilar necha marta uchrashini hisoblab chiqing. Ularni ekranga chiqaring.
12. Satr berilgan. Berilgan ikkita gapdagi oʻchraydigan bir xil soʻzni chop eting.
13. Orasida bitta ochilgan qavs va bitta yopilgan qavs mavjud boʻlgan belgili satr berilgan. Bu qavslar orasidagi barcha belgilarni ekranga chiqaring.
14. Lotin harflari va raqamlardan tashkil topgan satr mavjud. ...
15. Ikki nuqta bilan tugovchi va nuqta vergul bilan ajarilgan soʻzlar toʻplami berilgan. a harfi bilan tugovchi soʻzlar qanchaligini aniqlang.
16. Satr berilgan. Tarkibida kamida bitta k harfi boʻlgan soʻzni koʻrsating.
17. Satr berilgan. Boshlanishi va tugashi bir xil harfdan iborat soʻzni toping.
18. Satrdagi barcha ikki nuqtalarni nuqta vergul bilan almashtiring. Almashishlar migʻdorini hisoblang.

19. Satrdagi ikki nuqta belgilarini o'chiring va o'chirilgan belgilar mig'dorini hisoblang.
20. Satrdagi so'zlar orasidagi bO'sh joy o'rniga vergul va bO'sh joyni qo'ying.
21. Qavsga olingan belgili satr qismini o'chiring (qavslar bilan birgalikda).
22. Berilgan so'z satrda necha marta uchrashini aniqlang.
23. Satrda bitta nuqtali vergul mavjud. Nuqtali vergulgacha bo'lgan va undan keyingi belgilar mig'dorini hisoblang.
24. Satr berilgan. Birinchi $n/2$ belgilar orasida uchraydigan barcha ikki nuqtalarni nuqta bilan almashtiring va $n/2$ dan keyingi belgilar orasida uchraydigan barcha undov belgilarni nuqta almashtirib uni o'zgartiring.
25. Satr bitta so'zdan tashkil topgan. Uni chapdan o'ngga va o'ngdan chapga qarab o'qiganda bir xil bo'lishini (ya'ni u polindrom hisoblanishini) tekshiring.
26. Qo'lyozmadagi so'zlarning har biri teskarisiga yozilib shifrlangan. Xabarni oching (rasshifrovka qiling).
27. Berilgan satrda ochilgan qavslar soni yopilgan qavslar soni bilan bir hilligini tekshiring.
28. 200 belgidan oshib ketmaydigan ixtiyoriy ruscha matndan tashkil topgan satr berilgan. Bu matnda qaysi belgi necha marta uchrashini yozing. Javob grammatik jihatdan to'g'ri bo'lishi kerak, masalan «a – 25 marta», «k – 3 marta» va h.k.
29. Ingliz so'zlardan iborat berilgan massivni alfavit bo'yicha tartiblang.
30. Ikkita A va B satrlar berilgan. A satrdagi harflardan B satrni tuzish mumkinligini (harflarni o'rnini almashtirib ishlatish mumkin, lekin ularni bir martadan ortiq ishlatish mumkin emas) tekshiradigan dastur tuzing. Masalan, A:INTEGRAL, B:AGENT – tuzish mumkin; B:AGENT – tuzish mumkin emas.

5.2 PYTHON DASTURLASH TILIDA FAYLLAR VA ULARDAN FOYDALANISH

Reja:

1. Fayllarni faollashtirish;
2. Fayllar ustida amallar bajarish.

4. Fayldan ma'lumot o'qish

5. Fayl tarkibiga ma'lumotlarni yozish

6. Fayl tarkibidagi ma'lumotlarni o'chirish

Tayanch so'zlar: *fayl, open, read, write, del, mantiqiy nom, fizik nom.*

Python dasturlash tili tarkibida fayllar bilan ishlash uchun barcha turdagi imkoniyatlar e'tiborga olingan. Dasturlash tillarida fayllar dastur tarkibidagi o'zgaruvchilar qabul qiladigan qiymatlarni saqlash uchun ishlatiladi. Dasturlash asoslari fani rivojlanib borishi bilan dasturlash tillari tarkibida fayllar bilan ishlash ham rivojlanib bormoqda. Dasturlash asoslari tarkibida murakkab masalalarni hal etishda o'zgaruvchilarni qiymatlari soni ko'p bo'lsa, bunday holatlarda qo'lda kiritish qulay hisoblanmaydi. o'zgaruvchilarning qiymatlari soni ko'p bo'lgan holatlarda qiymatlar fayllarda saqlansa dasturning o'zi qiymatlarni tezkor holda fayllardan qabul qilib oladi. Kompyuterning ma'lum bir joyida qiymatlar fayllar asosida saqlanadi.

Ta'rif: Kompyuterda alohida nom bilan saqlanadigan ma'lum bir turga mansub bo'lgan ma'lumotlar majmuasi fayl deb nomlanadi.

Python dasturlash tili tarkibida fayllarga murojat qilish uchun fayl nomiga to'g'ridan to'g'ri murojat qilib bo'lmaydi, fayllarga murojat qilish uchun fayllarni dastur bilan bog'lash uchun alohida o'zgaruvchi qabul qilinadi.

Fayllarni faollashtirish

Python dasturlash tilida fayllar bilan ishlashda **.txt** kengaytmali fayllardan foydalanish maqsadga muvofiq bo'ladi. Dastur tarkibida fayllarga murojat qilish uchun albatta oldin **.txt** kengaytmali faylni yaratib alohida joyga saqlab quyish kerak bo'ladi. Dasturlash tilida ishlatiladigan fayllar nomi ikki turda bo'ladi.

1) Fizik nomi;

2) Mantiqiy nomi.

Faylning fizik nomi kompyuterda **.txt** kengaytmali nom bilan saqlangan nomi hisoblanadi. Faylning mantiqiy nomi esa dastur tarkibida faylning fizik nomi bilan bog'lashga xizmat qiladigan nomi hisoblanadi.

Python dasturlash tilida fayllarni faollashtirishdan oldin kutubxonaga murojaat qilish shart emas.

Python dasturlash tilida fayllarning asosiy nomiga to'g'ridan to'g'ri murojat qilib bo'lmaganligi sababli, fayllarga murojat qilish uchun faylning fizik nomiga mantiqiy nomini bog'lab murojat qilish kerak. Fayllar mantiqiy nomini fizik nomiga bog'lagandan so'ng dastur tarkibida faylga uning nomi bilan murojat qilinadi. Dastur tarkibining ixtiyoriy joyida faylga murojat qilish uchun albatta uning mantiqiy nomi bilan murojat qilinadi.

Python dasturlash tilida fayllarni faollashtirish va ularni mantiqiy nomi bilan bog'lash uchun ixtiyoriy o'zgaruvchini mantiqiy nom sifatida qabul qilinadi.

Python dasturlash tilida fayllar bir nechta holatlar bo'yicha faollashtiriladi.

1) Fayllarga ma'lumot o'qish uchun ochish bo'yicha faollashtirish;

2) Fayllardan ma'lumot yozish uchun ochish bo'yicha faollashtirish;

Python dasturlash tilida fayllarga ma'lumot yozishning ikki turi mavjud, birinchi turi bo'yicha ma'lumot yozilganda eski qiymatlar uchirilib ustiga yangi ma'lumot yoziladi. Ikkinchi tur bo'yicha faylga ma'lumot yozilsa, unda ma'lumot faylning oxiriga borib yozishni bajaradi. Ikki tur bo'yicha ham faylga ma'lumot o'zgaruvchi yordamida yoziladi.

Python dasturlash tilida fayllarga ma'lumot o'qish uchun ochish bo'yicha faollashtirishning umumiy ko'rinishi quyidagicha bo'ladi:

<fayl mantiqiy nomi>=open('fayl fizik nomi','r')

`f=open('test.txt','r')` orqali fayl faollashtirilganda, bu fayl ustida faqatgina ma'lumot o'qish amali bajariladi. Bunda *f*, faylning mantiqiy nomi, *test.txt* esa faylning fizik nomi, agar *test.txt* fayli dastur fayli bilan bir joyda saqlangan bo'lsa berilgan ko'rinishda yoziladi, aks holda *test.txt* fayli joylashgan adresi bilan yoziladi. `Open()` funksiyasi ichidagi *r*, fayldan ma'lumot o'qish bo'yruq'ini aks etadi. Natijada *f* mantiqiy nom fizik nom bilan bog'lanadi. Dastur natijasi biror bir fayldan ma'lumot o'qish uchun faollashtiriladi.

Python dasturlash tilida fayllarga ma'lumot yozish uchun faollashtirishning umumiy ko'rinishi quyidagicha bo'ladi:

<fayl mantiqiy nomi>=open('fayl fizik nomi','w')

Bu usulda faylga eski ma'lumotlar o'chirilib yangisi yoziladi. `f=open('test.txt','w')` orqali fayl faollashtirilganda, bu fayl ustida faqatgina ma'lumot yozish amali bajariladi. Bunda *f*, faylning mantiqiy nomi, *test.txt* esa faylning fizik nomi. `Open()` funksiyasi ichidagi *w*, faylga yozish bo'yrug'ini aks etadi. Natijada *f* mantiqiy nom fizik nom bilan bog'lanadi. Dastur natijasi biror bir faylga ma'lumot yozish uchun faollashtiriladi.

<fayl mantiqiy nomi>=open('fayl fizik nomi','a')

Bu usulda faylga ma'lumotlar faylning oxiridan yoziladi.

Agar faylni o'zidan ma'lumot o'qib shu faylni o'ziga ma'lumot yozish talab etilsa, albatta, birinchi fayl o'qish uchun ochilib ma'lumot o'qiladi va fayl yopilib keyin fayl yozish uchun ochilib ma'lumot yozilishi kerak.

Fayllarga murojat qilib bo'lgandan so'ng, albatta, fayllar yopilish kerak. Python dasturlash tilida fayllarni yopishning umumiy ko'rinishi quyidagicha bo'ladi:

<faylni mantiqiy nomi>.close()

Fayllarni yopish uchun, albatta, mantiqiy nomi dan so'ng `.close()` kalit so'zi yoziladi. Ikkita *f* va *g* o'zgaruvchilar fayllarning mantiqiy nomi sifatida qabul qilinib, ularni mos ravishda o'qish, yozish va yopish bo'yicha quyidagi ko'rinishda faollashtiriladi:

```
f=open('test.txt','r')
g=open('test.txt','w')
f.close()
g.close()
```


Python dasturlash tilida fayllarni o‘qish yoki yozish uchun alohida faollashtirish shart. Yuqorida python dasturlash tilida yozilgan dasturga e’tibor bersak, f fayl o‘qish uchun, g fayl yozish uchun faollashtiriladi va f , g fayllarni yopish jarayonlari qarab o‘tildi.

Python dasturlash tilida fayllarning mantiqiy nomi bog‘langandan so‘ng faylga uning mantiqiy nomi bilan murojat qilinadi. Faylning mantiqiy nomi fizik nomiga bog‘lanish vaqtida fizik nomi adresi bilan to‘liq yozilish kerak aks holda fayl bog‘lanmaydi. Agar yaratilgan dastur turgan papkada fayl yaratilgan bo‘lsa, adresi ko‘rsatilmasdan uning nomi ko‘rsatilishi kifoya.

Fayllar ustida amallar bajarish

Python dasturlash tili tarkibida fayllar ustida amallar bajarish uchun yuqoridagi holatlar bo‘yicha oldin faollashtirish va mantiqiy nomi bilan bog‘lashi shart. Fayllar ustida quyidagi amallar mavjud.

- 1)Fayllardan ma’lumot o‘qish
- 2)Fayllarga ma’lumot yozish
- 3)Fayllardan ma’lumot o‘chirish

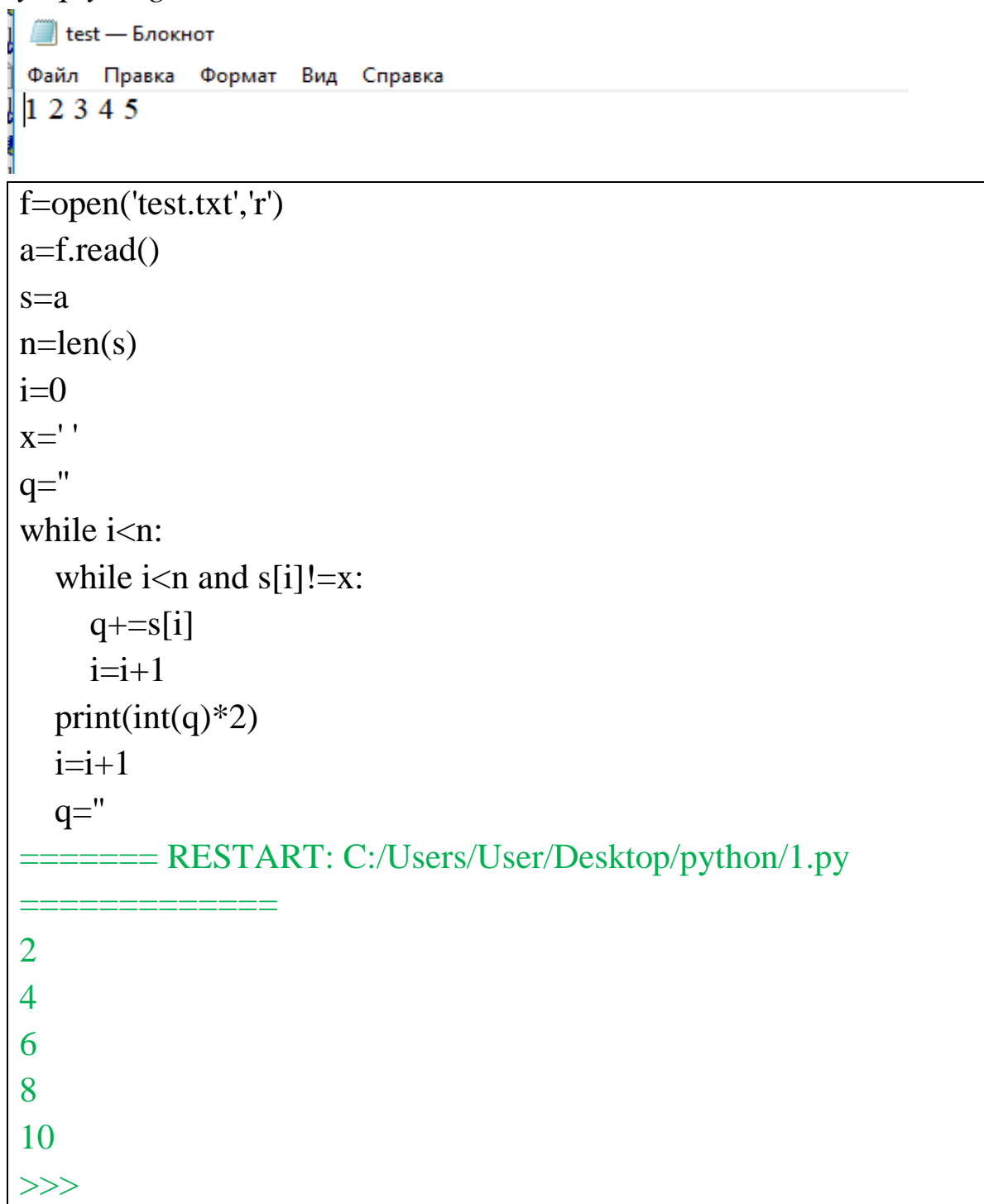
Fayldan ma’lumot o‘qish jarayonini python dasturlash tilida amalga oshirish uchun fayl o‘qish uchun faollashtirilish kerak. Fayldan ma’lumotlar faqat o‘zgaruvchilarga o‘qiladi. Fayldan ma’lumotlarni o‘qish jarayonida ma’lumotlar satr turiga mansub bo‘ladi.

Python dasturlash tilida fayldan ma’lumot o‘qishning umumiy ko‘rinishi quyidagicha bo‘ladi:

<o‘zgaruvchi>=<mantiqiy nom>.read()

Fayl o‘qish uchun ochilgandan so‘ng uning elementlarini biror bir o‘zgaruvchiga o‘qib olish imkoniyati mavjud. Fayldan olingan ma’lumotlar satr ko‘rinishda bo‘lganligi sababli boshqa o‘zgaruvchiga olinib son ko‘rinishiga o‘tkaziladi. Satrdan sonlarni ajratib olish uchun so‘z ajratgan algoritmdan foydalanamiz, lekin bundan soddaroq usuli bo‘lishi kerak.

Misol: 1 dan 5 gacha sonlar yozilgan test.txt faylidan uning elementlarini ko‘chirib olib ikkiga ko‘paytirib ekranga chiqaring. Test.txt fayl quyidagicha



The image shows a Notepad window titled "test — Блокнот" with a menu bar containing "Файл", "Правка", "Формат", "Вид", and "Справка". The text area contains the following Python code and its output:

```
f=open('test.txt','r')
a=f.read()
s=a
n=len(s)
i=0
x=' '
q=""
while i<n:
    while i<n and s[i]!=x:
        q+=s[i]
        i=i+1
    print(int(q)*2)
    i=i+1
    q=""
===== RESTART: C:/Users/User/Desktop/python/1.py
=====
2
4
6
8
10
>>>
```

Fayl elementlarini ro‘yxat elementlariga ham olish mumkin, quyidagi dastur orqali fayl elementlari ro‘yxat elementlariga olinadi.

```
f=open('test.txt','r')
a=f.read()
t=[]
```

```

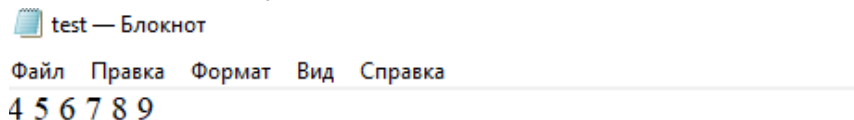
t=a
print(type(t))
print(t)
===== RESTART: C:/Users/User/Desktop/python/1.py =====
<class 'list'>
1 2 3 4 5
>>>

```

Python dasturlash tilida fayl elementlarini oxirigacha o‘qib olish imkoniyati mavjud, bu imkoniyatni **strip()** funksiyasi amalga oshiradi. Fayl elementlarini oxirigacha o‘qish jarayonini quyidagi dastur orqali amalga oshiramiz.

Misol. *f(test.txt)* faylning barcha elementlarini ekranga chiqarish dasturini tuzing.

test.txt fayl ko‘rinishi



```

f=open('test.txt','r')
a=f.read()
for i in a:
    print(i.strip())
===== RESTART: C:/Users/User/Desktop/python/1.py =====
=====
4
5
6
7
8

```


```
9
```

```
>>>
```

Python dasturlash tilida fayl elementlarini teskari tartibda o‘qish imkoniyati mavjud, bu imkoniyatni **reversed()** funksiyasi amalga oshiradi. Fayl elementlarini teskari tartibda o‘qish jarayonini quyidagi dastur orqali amalga oshiramiz.

Misol. *f(test.txt)* faylning barcha elementlarini teskari tartibda ekranga chiqarish dasturini tuzing.

test.txt fayl ko‘rinishi

 test — Блокнот

Файл Правка Формат Вид Справка

4 5 6 7 8 9

```
f=open('test.txt','r')
```

```
a=f.read()
```

```
for i in reversed(a):
```

```
    print(i.strip())
```

```
===== RESTART: C:/Users/User/Desktop/python/1.py
```

```
=====
```

```
9
```

```
8
```

```
7
```

```
6
```

```
5
```

```
4
```

```
>>>
```

Python dasturlash tilida fayldan ma’lumotlar yuqoridagi holatlar bo‘yicha o‘qiladi va qayta ishlanadi.

Fayl tarkibiga ma'lumotlarni yozish

Python dasturlash tilida ma'lumotlarni ikkita turda yozishni aytgan edik. Python dasturlash tilida birinchi tur bo'yicha faylga ma'lumot yozish uchun `<fayl mantiqiy nomi>=open('fayl fizik nomi','w')`, ikkinchi tur bo'yicha faylga ma'lumot yozish uchun `<fayl mantiqiy nomi>=open('fayl fizik nomi','a')` buyruqlari oldin yozilishi shart undan so'ng uning tarkibiga ma'lumot yozish mumkin. Python dasturlash tilida faylga ma'lumot yozishning umumiy ko'rinishi quyidagicha bo'ladi:

`<fayl ni mantiqiy nomi> .write(o'zgaruvchi)`

Fayl tarkibiga ma'lumot yozilganda uning oldingi ma'lumotlari o'chirilib, yangi ma'lumotlar yoziladi yoki fayl oxiridan yoziladi.

Misol: *a ro'yxat berilgan ro'yxat elementlarini f(test.txt) faylga yozish dasturini tuzing.*

```
f=open('test.txt', 'w')
a=[1,2,8]
f.write(str(a))
f.close()
===== RESTART: C:/Users/User/Desktop/python/1.py =====
>>>
Natija fayl
test — Блокнот
Файл  Правка  Формат  Вид  Справка
[1, 2, 8]
```

Dastur natijasiga e'tobor bersak, faylga ma'lumot faqat satr ko'rinishida yozilganligi sababli ro'yxat elementlari to'g'ridan to'g'ri faylga yozilmoqda.

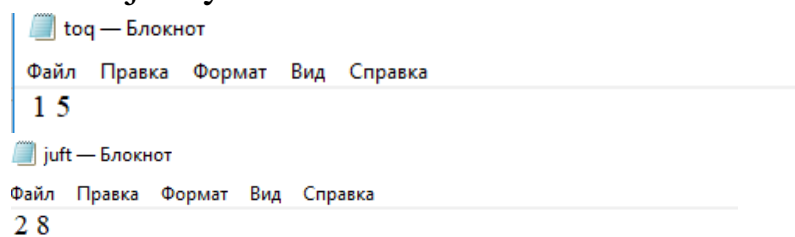
Misol: *a ro'yxat berilgan ro'yxat elementlari ichidan toqlarini f faylga, juftlarini g faylga yozing.*

```

f=open('toq.txt', 'w')
g=open('juft.txt', 'w')
a=[1,2,8,5]
n=len(a)
x=' '
for i in range(n):
    if a[i]%2==0:
        g.write(str(a[i]))
        g.write(x)
    else:
        f.write(str(a[i]))
        g.write(x)
f.close()
g.close()
===== RESTART: C:/Users/User/Desktop/python/1.py
=====
>>>

```

Natija fayllar



Yuqoridagi dasturda massiv elementlarining toqlarini f fayl orqali toq.txt faylga, juft elementlarini esa g o'zgaruvchi orqali juft.txt faylga joylashtirdi.

Ikkinchi tur bo'yicha ma'lumot yozish degani fayl oxiriga ma'lumot yozish tushuniladi. Faylning oxiriga ma'lumot yozish uchun <fayl mantiqiy nomi>=open('fayl fizik nomi','a') buyrug'i keltiriladi.

Misol: To'rtta 1 2 3 4 elementlari mavjud bo'lgan f(test.txt) fayl oxiriga 10 dan 19 gacha bo'lgan sonlarni yozing.

```

f=open('test.txt', 'a')
x=' '

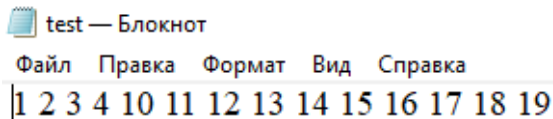
```

```

for i in range(10,20):
    f.write(x)
    f.write(str(i))
f.close()
===== RESTART: C:/Users/User/Desktop/python/1.py
=====
>>>

```

Natija fayl



Yuqoridagi dasturda test.txt faylida 1 2 3 4 sonlar mavjud edi, 10 dan 19 gacha bo'lgan sonlar 1 2 3 4 sonlaridan keyin test.txt fayliga joylashtirildi.

Fayl tarkibidagi ma'lumotlarni o'chirish

Fayl tarkibidagi ma'lumotlarni o'chirish uchun Python dasturlash tili tarkibida imkoniyat mavjud. Ma'lumot yozilgan faylni tozalash yoki uning tarkibidagi elementlarni o'chirish uchun quyidagi kodni yozish kerak.

```
f=open('test.txt', 'w')
```

Yuqoridagi dastur asosida test.txt fayli tozalandi.

Python dasturlash tili tarkibida fayllardan foydalanish jarayonlari yuqoridagi holatlar bo'yicha amalga oshiriladi. Fayldan ma'lumot o'qish, faylga ma'lumot yozish va fayldan ma'lumot o'chirish jarayonlari asosida fayllarga tegishli masalalarni hal etish imkoniyati mavjud.

Nazariy savollar

1. Fayl deb nimaga aytiladi?

2. Dasturlashda qanday kengaytmali fayllardan foydalaniladi?

3. Dasturlashda fayl nomi qanday qismlardan tashkil topgan?
4. Fayllarni fizik va mantiqiy nomlarini bog'lashning umumiy ko'rinishi?
5. Fayllarni o'qish uchun faollashtirishning umumiy ko'rinishini?
6. Fayldan ma'lumot o'qish komandasining umumiy ko'rinishi?
7. strip() va reversed() funksiyalarining vazifasi?
8. Fayllarni yozish uchun faollashtirishning qanday turlari mavjud?
9. Faylga ma'lumot yozishning ikki turdagi komandasining umumiy ko'rinishi?
10. Fayllar elementlarini o'chirish uchun faollashtirishning umumiy ko'rinishini?
11. Fayllarni yopish komandasining umumiy ko'rinishini?

Mustaqil ishlash uchun topshiriqlar

1. Elementlari sonli bo'lgan mavjud turlashgan faylni berilgan son bilan almashtiring (yangi qiymat klaviaturadan kiritiladi):

- a) birinchi elementni;
- b) beshinchi elementni;
- v) k-nchi elementni;
- g) oxirgi elementni.

2. Elementlari alohida so'zlar bo'lgan mavjud turlashgan faylni berilgan qiymat bilan almashtiring (yangi so'z klaviaturadan kiritiladi):

- a) birinchi elementni;
- b) uchinchi elementni;
- v) s-nchi elementni;
- g) oxirgi elementni.

3. Elementlari alohida so'z hisoblanuvchi turlashgan fayl berilgan. "t" harfi bilan boshlanuvchi hamma so'zlarni chop qiling. Bunda ikki variantni qarang:

- a) ma'lumki, mavjud faylga 30 ta so'z yozilgan;
- b) mavjud fayl o'lchami ma'lum emas.

4. Turlashgan fayl elementlari alohida so'z hisoblanadi. Uning barcha elementlarini boshqa so'z bilan almashtiring. Bunda ikkita variantni qarang:

- a) ma'lumki, mavjud faylga 12 ta so'z yozilgan;
- b) mavjud fayl hajmi ma'lum emas.

5. Sonli turlashgan fayl mavjud. Uning tartib nomeri 3 ga karrali bo'lgan hamma elementlarini almashtiring. Yangi qiymat klaviaturadan kiritiladi. Bunda ikkita variantni qarang:

a) ma'lumki, mavjud faylga 20 ta son yozilgan;

b) mavjud fayl hajmi ma'lum emas.

6. Butun sonli turlashgan fayl mavjud. Uning hamma juft elementlarini nol bilan almashtiring. Bunda ikkita variantni qarang:

a) ma'lumki, mavjud faylga 13 ta son yozilgan;

b) mavjud fayl hajmi ma'lum emas.

7. Elementlari alohida so'zlardan iborat turlashgan fayl mavjud. Agar so'z "k" harfi bilan boshlansa, uni "K" harfi bilan almashtiring.

8. Sonli turlashgan fayl mavjud. Toping (hamma holatlarda fayl hajmi ma'lum emas):

a) faylning birinchi va ikkinchi sonlar yig'indisini;

b) faylning k-nci va q-nci sonlari yig'indisi;

v) faylning birinchi va oxirgi sonlari ko'paytmasini;

g) faylning hamma sonlarining yig'indisini;

d) fayldagi sonlar miqdorini;

e) a sonidan oshib ketmaydigan fayldagi sonlar miqdorini;

j) faylning musbat sonlarining o'rta arifmetigini;

i) mavjud fayldagi maksimal sonni;

k) fayldagi minimal sonning tartib nomeri (agar bunday fayllar bir nechta bo'lsa, ularning birinchisini nomerini toping).

9. Elementlari alohida so'zlardan iborat turlashgan fayl mavjud. Toping (hamma holatlarda fayl hajmi ma'lum emas):

a) "m" harfi bilan boshlanuvchi so'zlar sonini;

b) "l" harfi bilan boshlanuvchi barcha so'zlar va ularning birinchisini tartib nomerini chop qiling;

v) tartib nomeri juft bo'lgan so'zlar va ulardan so'z tuzing;

g) eng uzun so'zni;

d) eng qisqa so'zni.

10. Turlashgan faylga Shaxar nomi va uning aholisi soni yozilgan. Har bir Shaxarning aholisini 5% ga oshiring (aholi miqdori – har doim butun son).

11. Butun sonli turlashgan fayl mavjud. Undan birinchi noldan keyin yozilgan sonni o‘chiring (faylda nol mavjud deb qabul qilinsin). Natija boshqa faylga yozilsin.

12. Elementlari alohida belgilardan iborat turlashgan fayl mavjud. Undan birinchi “o” harfini o‘chiring (“o” harfi faylda mavjud deb qabul qilinsin). Natija boshqa faylga yozilsin.

13. Butun sonli turlashgan fayl mavjud. Birinchi 100 sonidan keyin 100 sonini qo‘ying. Natija boshqa faylga yozilsin.

14. o‘ttizta sonli turlashgan fayl mavjud. Boshqa faylga mavjud faylni teskari tartibda yozilsin.

15. Elementlari 20 ta alohida belgidan iborat turlashgan fayl mavjud. Boshqa faylga mavjud fayl belgilarini teskari tartibda yozilsin.

16. Elementlari son bo‘lgan, o‘lchamlari bir xil turlashgan ikkita fayl mavjud. .

17. Elementlari alohida belgilardan iborat bo‘lgan, o‘lchamlari bir xil turlashgan ikkita fayl mavjud....

18. Elementlari sondan iborat, o‘lchamlari bir xil turlashgan ikkita fayl mavjud. Dastlab birinchi faylning, keyin ikkinchisini sonlarini yozib uchinchi faylni hosil qiling. Sonlar ketma-ketlik tartibi saqlansin.

19. Elementlari alohida belgilardan iborat turlashgan fayl mavjud. Bu faylning barcha raqamlarini ikkinchi faylga, qolgan belgilarni esa uchinchi faylga yozing. Sonlar ketma-ketlik tartibi saqlansin.

20. Elementlari alohida belgilardan iborat bo‘lgan, o‘lchamlari bir xil turlashgan ikkita fayl mavjud. Elementlar tartibi saqlangan holda birinchi fayl elementlarini ikkinchisiga, ikkinchisini birinchisiga yozing. Yordamchi fayldan foydalaning.

21. Elementlari alohida belgilardan iborat bo‘lgan, o‘lchamlari bir xil turlashgan ikkita fayl mavjud. Ularning elementlarini mos kelishini aniqlang. Agar mos kelmasa, u holda bu fayllarning bir-biridan farq qiluvchi birinchi komponenta nomerini oling.

22. Elementlari butun sonlardan iborat bo‘lgan, turlashgan ikkita tartiblashgan fayl mavjud. Berilgan ikkita faylni tartiblashgan holda birlashtirib, Yangi turlashgan faylni oling. Birlashtirish algoritmidan berilgan fayllar juda katta yozuvlardan iborat bo‘lish mumkinligi ham qaralishi kerak.

23. Tashqi matnli fayl mavjud. Uning satrlaridan birinchi eng qisqasini chop qiling.

25. T matnli faylga bo'sh bo'lmagan haqiqiy sonlar ketma-ketligi bo'sh joy bilan ajratilib yozilgan. Bu sonlarning eng kattasini topish uchun $\text{Max}(T)$ funksiyasini tavsiflang.

26. Matnli fayl mavjud. Undan elementlarni o'chiring:

VI-BOB. PYTHON DASTURLASH TILIDA GRAFIKA

Python dasturlash tili tarkibida boshqa dasturlash tillari kabi grafik imkoniyatlari anchagina keng e'tiborga olingan. Dasturlashda grafik rejim bilan ishlashda, albatta, grafik rejimga o'tish komandalari berilishi shart. Dasturlashning grafik rejimida geometriya fanining ba'zi bir elementlari, nuqtalar to'plami, funksiayalar va ikki va uch o'lchovli grafik elementlarni tasvirlash imkoniyatlari mavjud. Python dasturlash tilida yuqorida ta'kidlangan elementlardan tashqari ma'lum bir sohalar, shakllar, shakllar atrofiga izohli ma'lumotlar hamda rangli soha va rangli shakllar hosil qilish imkoniyatlari mavjud.

6.1 PYTHON DASTURLASH TILI TARKIBIDA GRAFIKLAR CHIZISH VA ULARNI QAYTA ISHLASH

Reja:

- 1. Grafik muhitini faollashtirish;*
- 2. Tekislikda chizma va shakllar chizish;*
- 3. Chizmalarni alohida faylda saqlash;*
- 4. Grafikga ma'lumot yozish;*
- 5. Matematik funksiayalar grafiklarini chizish.*

Tayanch so'zlar: *matplotlib, plot(y), show, savefig, ylabel, xlabel, title, color.*

Python dasturlash tilida ma'lum bir shakllar va chizmalarni hosil qilish uchun avval, albatta, grafik rejimni hosil qilish kerak, ya'ni grafik kutubxonani faollashtirish kerak. Python dasturlash tilida grafik rejim hosil qilingandan so'ng uning tarkibiga kerakli chizma va shakllarni hosil qilish buyruqlarini yozish mumkin.

Grafik muhitini faollashtirish

Python dasturlash tili tarkibida boshqa dasturlash tillari kabi grafik rejimi va uning imkoniyatlari mavjud. Chizmalar va sohalarni hosil qilish uchun python dasturlash tilida **matplotlib** kutubxoansini chaqirish kerak.

Matplotlib kutubxonasini faollashtirishning umumiy ko'rinishi quyidagicha.

from matplotlib.pyplot import*

Grafik rejimi hosil qilingandan so‘ng kompyuter ekranini koordinatalar sistemasini I-choraki deb qarash kerak. Bunda kompyuter ekraniga chiziladigan shakl va chizmalar xuddi koordinatalar sistemasining I-chorakida chiziladigandek buruqlar beriladi. Kompyuterning ekрани bir nechta nuqtalar matritsasidan tashkil topgan. Dasturchi tomonidan chizilgan shakl va chizmalar ekran rangi bilan bir xil rangda bo‘lsa, chizilgan shakl va chizmalar ko‘rinmasdan qoladi, shuning uchun chiziladigan shakl, chizma va nuqtalar uchun alohida ranglar ham berilish mumkin.

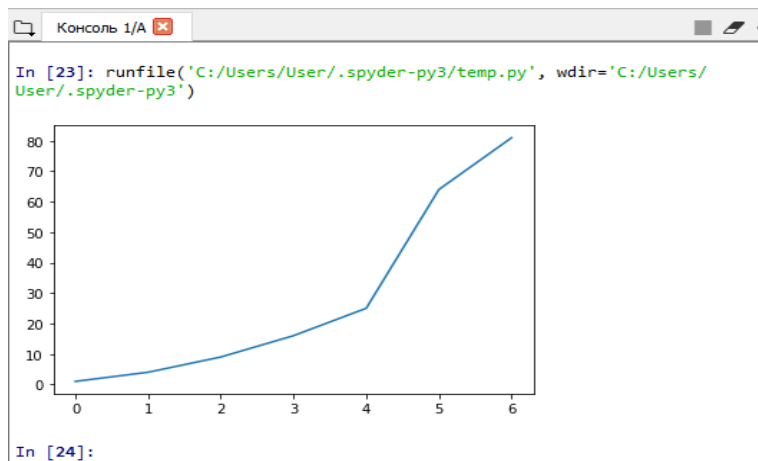
Tekislikda chizma va shakllar chizish

Python dasturlash tili tarkibida shakl va chizmalar nuqtalar ketma-ketligidan hosil bo‘ladi. Python dasturlash tilida nuqta, shakl va chizmalarni rangi va chizma turi alohida beriladi. Python dasturlash tili tarkibida grafik shakllarni quyidagi funksiyalar orqali chiziladi:

plot(y), ***show()***-funksiyasi y to‘plam yoki y ro‘yxat elementlarini ikki o‘lchovli koordinatalar sistemasida chizish uchun xizmat qiladi.

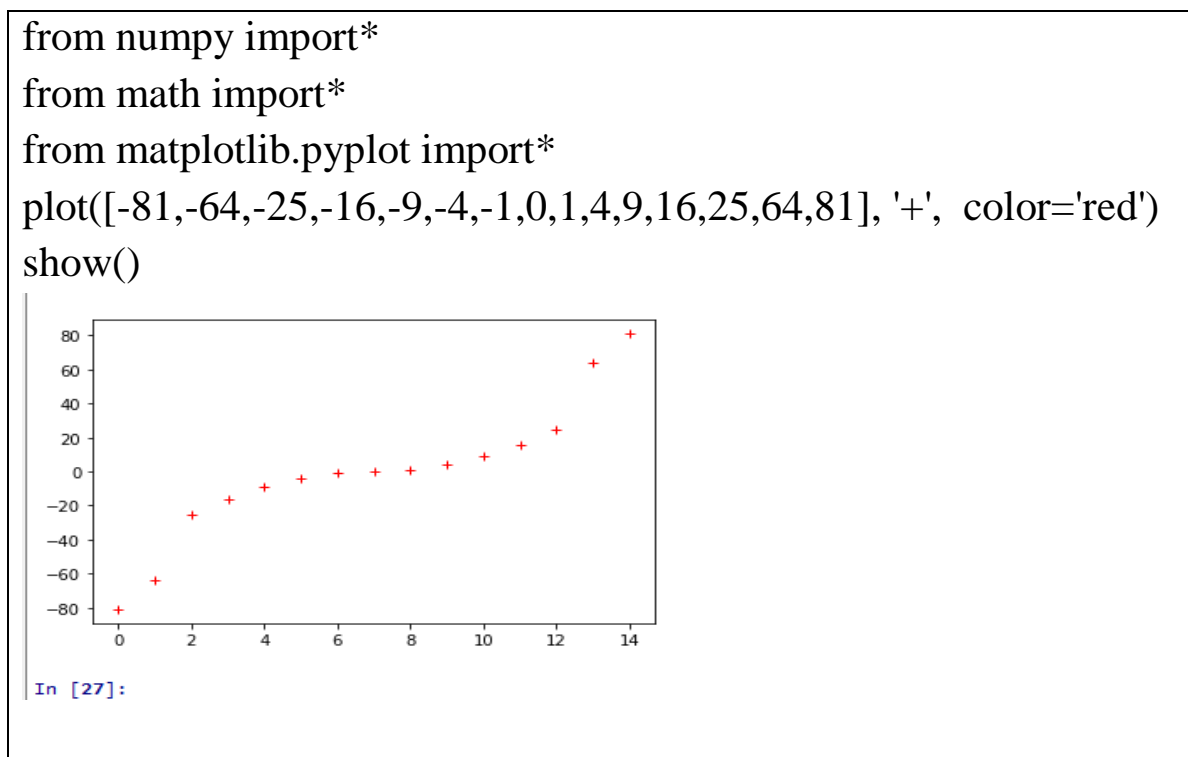
`plot()` funksiyasini ishlash jarayonini quyidagi dastur orqali qarab o‘tamiz.

```
from numpy import*
from math import*
from matplotlib.pyplot import*
plot([1,4,9,16,25,64,81])
show()
```



Chiziladigan shakl va chizmalarning chiziq turlari va ranglarini o‘zgartirish ham mumkin.

Misol. A ro‘yxat elementlarini + belgi bilan qizil rangda chizish dasturini tuzing.



Chiziladigan shakl va chizmalarning chiziq turlari '+', '_', '*', '-', 'v', 's', '>', '<', 'D', 'd', 'p', 'h', 'x', '|' kabi belgilar ko‘rinishida bo‘lishi mumkin.

Chiziladigan shakl va chizmalarning chiziq ranglari quyidagi jadval ko‘rinishida aniqlanadi.

Rang nomi(O‘zbek tilida)	Rang nomi(ingiliz tilida)	Rang kodi
Qora	Black	0
KO‘k	Blue	1
Yashil	Green	2
Och kO‘k	Cyan	3
Qizil	Red	4
Binafsha	Magenta	5

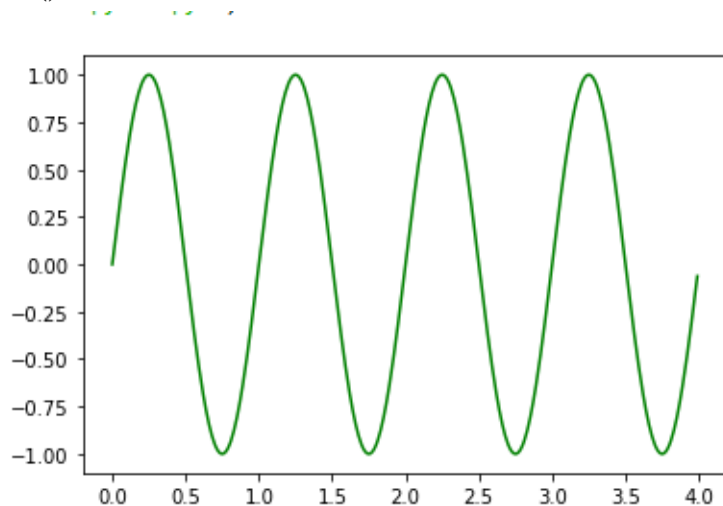
Malla	Brown	6
KO'lrang	Lightgray	7
Och qora	Darkgray	8
Och kO'k	Lightblue	9
Och yashil	Lightgreen	10
TO'q kO'k	Lightcyan	11
Och qizil	Lightred	12
Och binafsha	Lightmagenta	13
Sariq	Yellow	14
Oq	White	15

Misol. Ma'lum bir oraliqda $\sin(x)$ funksiyasi va uning argumentini koordinatalar sistemasida yashil rang bilan chizish dasturini yarating.

```

from numpy import*
from math import*
from matplotlib.pyplot import*
t=[]
x=[]
for i in range(400):
    t.append(i*0.01)
    x.append(sin(2*pi*t[i]))
plot(t, x, color='green')
show()

```



In [44]:

Chizmalarni alohida faylda saqlash

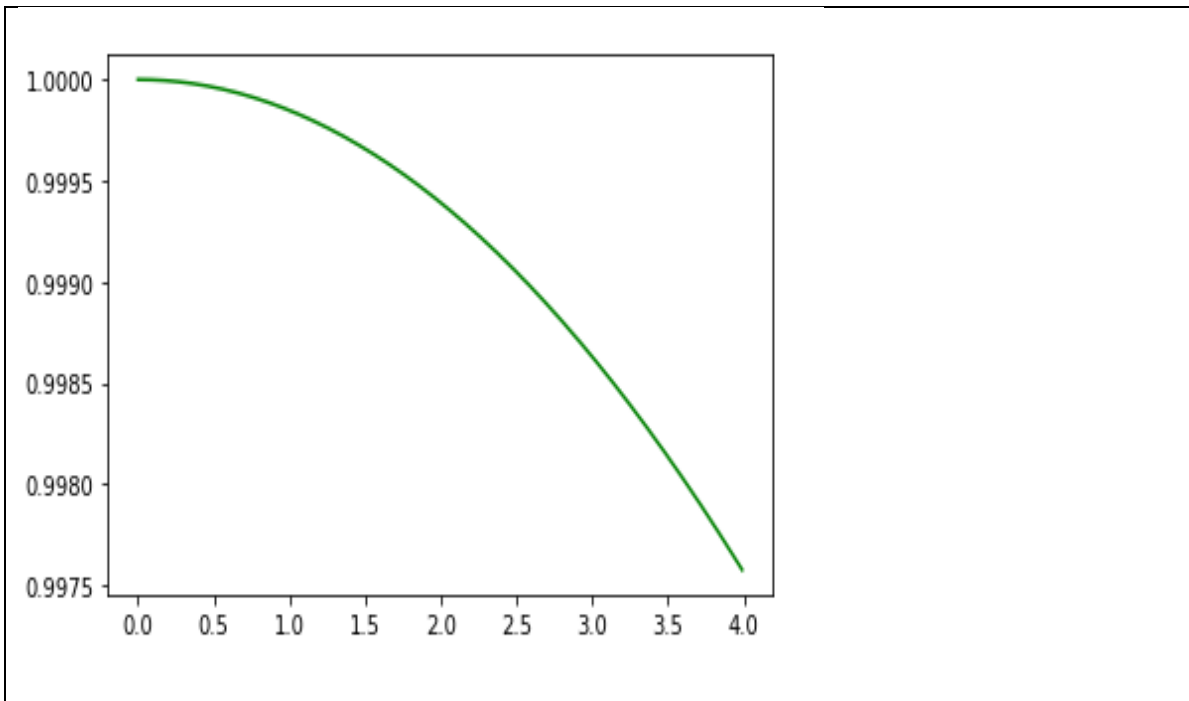
Python dasturlash tilida chiziladigan shakl va chizmalarni alohida .png kengaytmali fayllarga saqlash imkoniyati mavjud. Bu asosan katta turdagi ma'lumotlarni qayta ishlash vaqtida rasmlarni alohida fayl sifatida saqlash imkonini yaratadi. Chiziladigan shakl va chizmalarni alohida faylga quyidagi funksiya orqali amalga oshiramiz.

savefig('sincos.png')

Bu funksiya grafikni, dastur saqlangan papkaga saqlaydi, agar boshqa joyga saqlash kerak bo'lsa albatta adres " " belgi ichiga yozilish kerak. Yuqoridagi funksiyani ishlash jarayonini quyidagi dastur orqali qarab o'tamiz.

Misol. Ma'lum bir oraliqda $\cos(x)$ funksiya grafigini chizing va bu grafikni cosinus.png fayliga saqlash dasturini tuzing.

```
from numpy import*
from math import*
from matplotlib.pyplot import*
t=[]
x=[]
for i in range(400):
    t.append(i*0.01)
    x.append(cos(pi*t[i]/180))
plot(t, x, color='green')
savefig('cosinus.png')
show()
```

Grafikga ma'lumot yozish

Python dasturlash tilida chiziladigan shakl va chizmalarga ma'lumot yozish mumkin. Bu ma'lumot funksiyaga nom, OX va OY o'qi bo'yicha ma'lumot yozish mumkin. Ma'lumotlarni quyidagi funksiyalar amalga oshiradi.

Funksiyaga nom beris funksiyasi:

title('text')

OY o'qiga ma'lumot yozish:

ylabel('text')

OX o'qiga ma'lumot yozish:

xlabel('text')

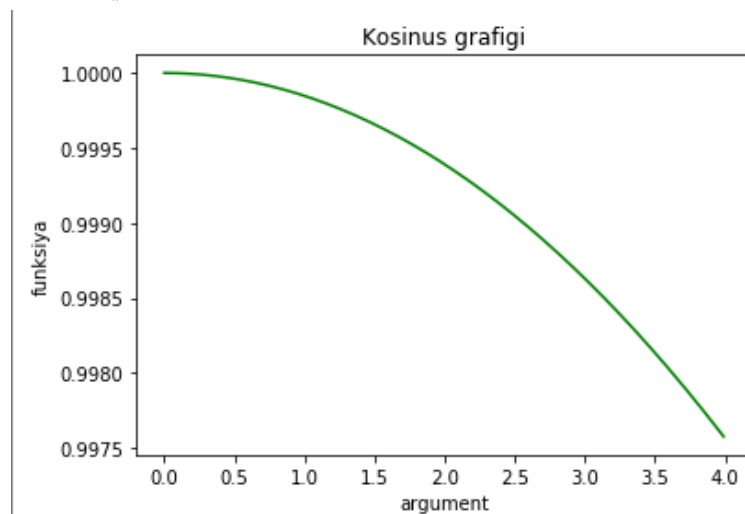
Yuqoridagi funksiyani ishlash jarayonini quyidagi dastur orqali qarab o'tamiz.

Misol. Ma'lum bir oraliqda $\cos(x)$ funksiya grafigini chizing va bu grafikni nomini kosinus grafigi, OX o'qini argument va OY o'qini funksiya deb nom beruvchi dastur tuzing.

```

from numpy import*
from math import*
from matplotlib.pyplot import*
t=[]
x=[]
for i in range(400):
    t.append(i*0.01)
    x.append(cos(pi*t[i]/180))
plot(t, x, color='green')
title('Kosinus grafigi')
xlabel('argument')
ylabel('funksiya')
show()

```



In [54]:

Matematik funksiyalar grafiklarini chizish

Python dasturlash tili tarkibida oddiy chizmalardan tashqari matematik funksiyalar grafiklarini chizish ham mumkin. Matematik funksiyalar grafiklarini chizishda avval, argumentning oraliq qiymati beriladi, bu xuddiki MAPLE tizimidagi kabi aniqlanadi. Argumentning qiymatlari qanchalik katta bo'lsa grafik ham shunch aniq chiziladi.

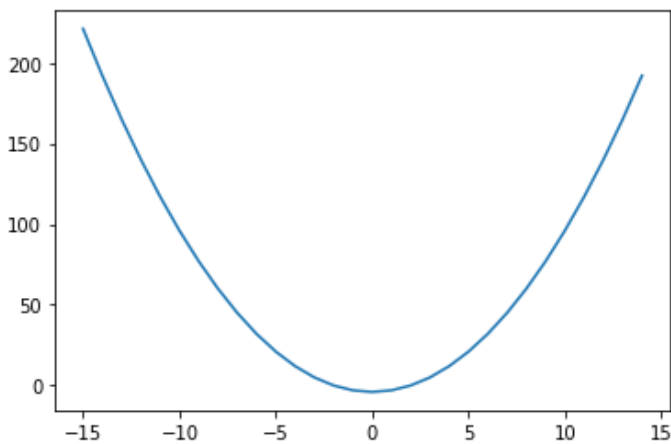
Yuqoridagi fikrlarni shakllantirishni quyidagi dastur orqali qarab o'tamiz.

Misol. Ma'lum bir oraliqda $y=x^2-4$ funksiya grafigini chizish dasturini tuzing.

```

from numpy import*
from math import*
from matplotlib.pyplot import*
x=arange(-15,15,1)
y=x**2-4
plot(x,y)
show()

```



In [66]:

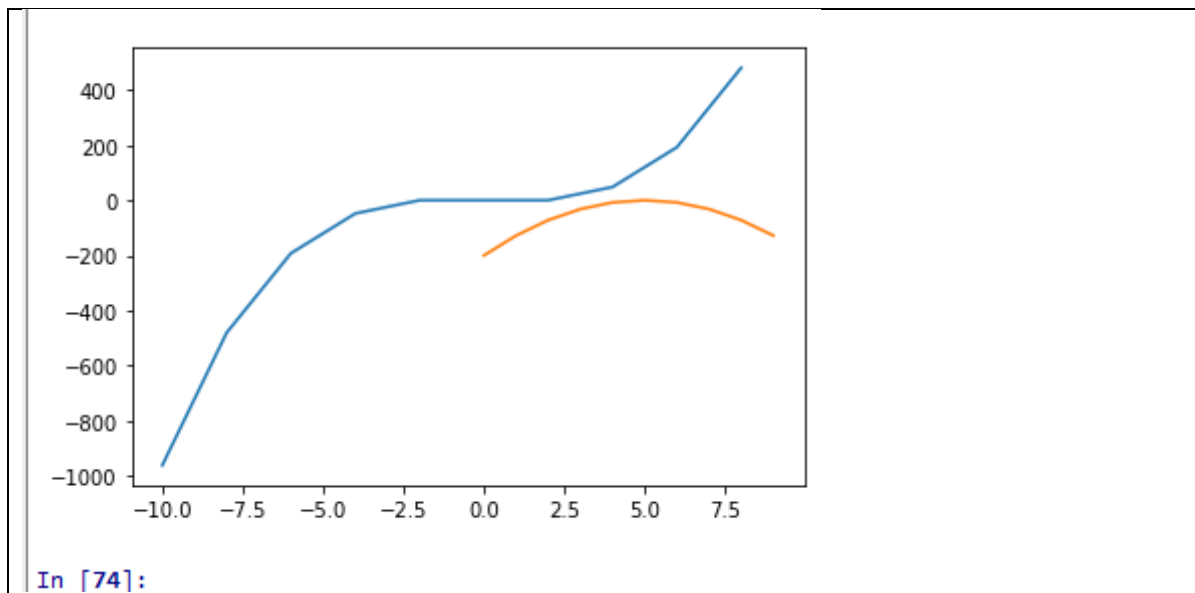
Ikki va undan ortiq funksiyalar grafiklarini bitta sistemaga ham chizish mumkin.

Misol. Ma'lum bir oraliqda $y=x^3-4x$ va $y=-2x^2$ funksiya grafiklarini chizish dasturini tuzing.

```

from numpy import*
from math import*
from matplotlib.pyplot import*
x=arange(-10,10,2)
y=x**3-4*x
z=-2*x**2
plot(x,y,z)
show()

```



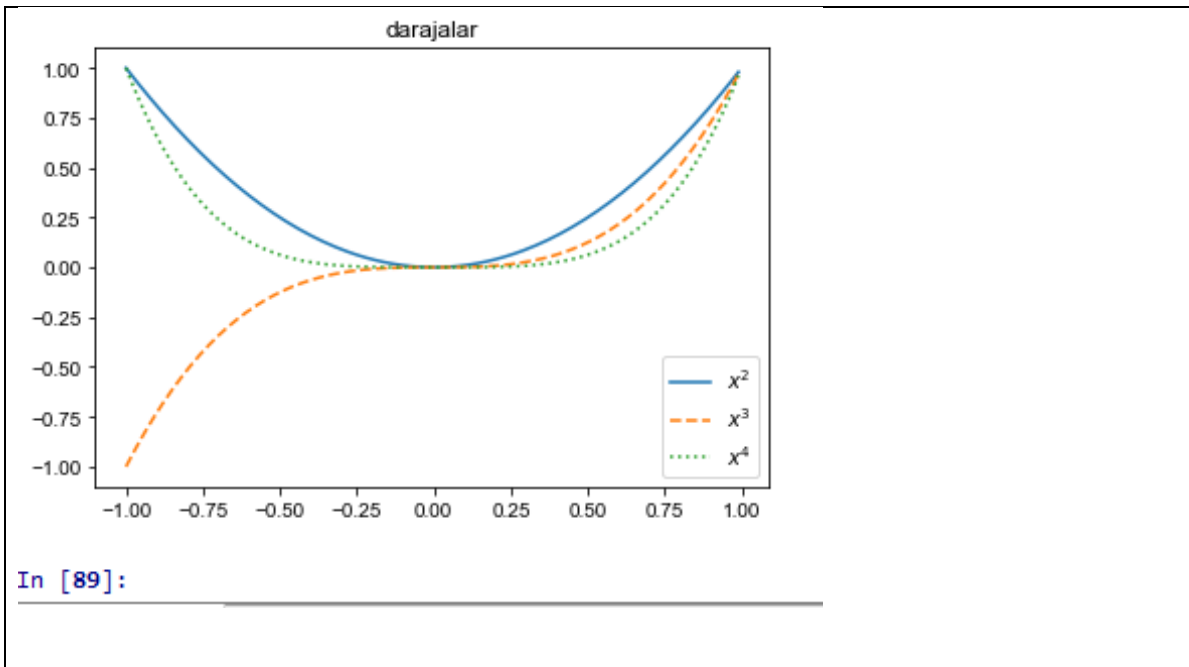
Bir nechta funksiyalar grafiklarini chizish va bu funksiyalarni ko‘rinishlarini ham chizish imkoniyatlari mavjud.

Misol. Ma’lum bir oraliqda $y=x^2$, $y=x^3$ va $y=x^4$ funksiya grafiklarini chizish dasturini tuzing.

```

from numpy import*
from math import*
from matplotlib.pyplot import*
from matplotlib import rcParams
rcParams ['font.sans-serif']='Arial'
t=arange(-1,1,0.01)
x=t**2
y=t**3
z=t **4
plot(t,x,label =r'$x ^2 $')
plot(t,y,'--',label =r'$x^3 $')
plot(t,z,':',label=r'$x^4 $')
legend ()
title ('darajalar')
show ()

```



Nazariy savollar

- 1 Grafik muhitini faollashtirish tushintirib bering.
- 2 Tekislikda chizma va shakllar chizish funksiyasi.
- 3 Chiziq turlari va ranglarini faollashtirishni tushintiri bering.
- 4 Chizmalarni alohida faylda saqlash funksiyasi va uning vazifasi.
- 5 Grafikga ma'lumot yozish, OX, OY va garfikga ma'lumot yozish funksiyalari.
- 6 Matematik funksiyalar grafiklarini chizish.

Mustaqil ishlash uchun topshiriqlar

- 1.Uchburchak uchlari koordinatalari berilganda qora rangli ekranda qizil rangli uchburchak hosil qiling va uchburchak ostiga "uchburchak" yozuvini hosil qiling.
- 2.Ikki nuqta koordinatalari berilganda qora rangli ekranda oq rangli kesma hosil qiling va kesma o'rtasidagi nuqtaga "kesma o'rtasi" yozuvini hosil qiling.
- 3.Koordinatalar sistemasi hosil qiling va markazi koordinata boshida bo'lgan aylana hosil qiling.
- 4.[a,b] ro'yxat elementlarini chiziqning bir nechta turlari bilan chizing.

5.[a,b] oraliqda to'rtta trigonometrik funksiyalarni har xil rang bilan chizing.

6.[a,b] oraliqda kvadrat va kubik funksiyalarni har xil rang bilan chizing.

7.[a,b] oraliqda ko'rsatkichli va logorifm funksiyalarni har xil rang bilan chizing.

8.[a,b] oraliqda aylanalarni har xil rang bilan chizing.

6.2 PYTHON DASTURLASH TILIDA DIAGRAMMALAR VA UCH O'LCHOVLI GRAFIKLAR CHIZISH

Reja:

Diagrammalar;

Uch o'lchovli grafika.

Tayanch so'zlar: *diagramma, subplot, mplot3d.*

Diagrammalar

Python dasturlash tili tarkibida ma'lumotlarni vizual tarzda nomoyon qilish uchun diagrammalarni shakllantirish imkoniyatlari mavjud. Diagrammalar python dasturlash tilida ikki turga ajratiladi:

- Ustun ko'rinishidagi diagrammalar;
- Aylana ko'rinishidagi diagrammalar.

Ustun ko'rinishidagi diagrammalar ma'lumotlarni OX va OY o'qi kesimida vertikal ko'rinishida shakllantiriladi. Ma'lumotlarni ustun ko'rinishida shakllantirish uchun quyidagi funksiyalar faollashtiriladi.

subplot()

hist()

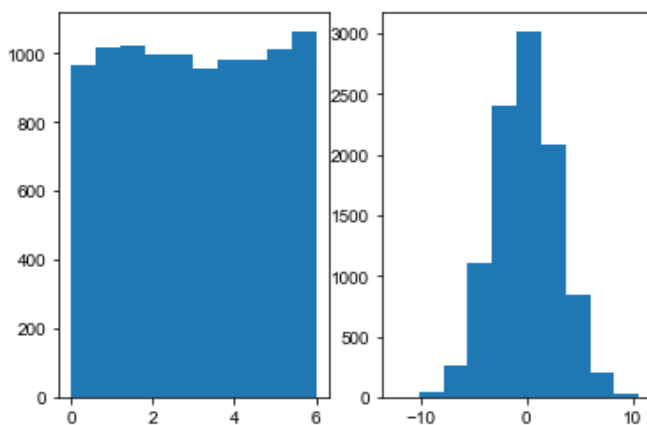
Yuqoridagi funksiyalar ish jarayonini quyidagi dastur orqali aniqlab olamiz.

Misol. Random funksiyasi orqali hosil qilingan ikkita ro'yxat elementlarini ustun ko'rinishidagi diagramma chizish dasturini tuzing.

```

from numpy import*
from math import*
from matplotlib import rcParams
from random import uniform , normalvariate
from matplotlib.pyplot import *
v=[]
for i in range(10000):
    v.append ( uniform (0,6))
subplot (1,2,1)
hist (v)
w = []
for i in range(10000):
    w.append(normalvariate(0, 3))
subplot(1 , 2, 2)
hist(w )
show()

```

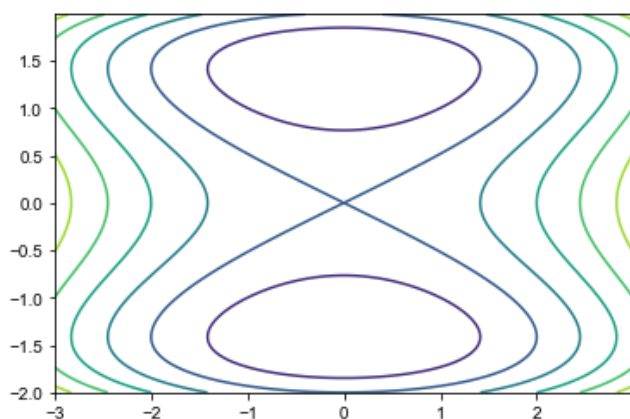


In [99]:

Aylana ko‘rinishidagi diagrammalar ma’lumotlarni OX va OY o‘qi kesimida aylana ko‘rinishida shakllantiradi. Ma’lumotlarni aylana ko‘rinishida shakllantirish uchun quyidagi aylana tenglamasidan ham foydalaniladi.

Misol. Ikkita to‘plam elementlarini aylana ko‘rinishidagi diagramma chizish dasturini tuzing.

```
from numpy import *
from matplotlib.pyplot import *
from matplotlib.mlab import *
x = arange( -3 , 3, 0.01)
y = arange ( -2 , 2, 0.01)
X , Y = meshgrid (x , y )
Z = X **2 -4* Y **2+ Y **4
contour (X,Y,Z )
show ()
```



In [102]:

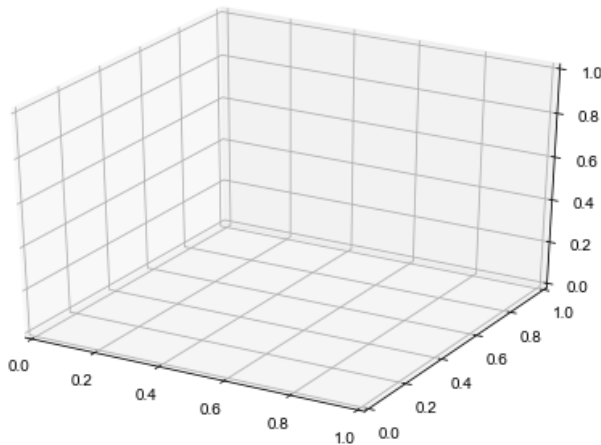
Uch o‘lchovli grafika

Python dasturlash tili tarkibidagi Matplotlib kutubxonasida, ikki o‘lchovli grafikaga qo‘shimcha ravishda, uch o‘lchovli grafiklarni hosil qilish imkoniyati mavjud. Buning uchun **mplot3d** moduli ishlatiladi. Uch o‘lchovli grafiklarni chizish uchun, birinchi navbatda siz uch o‘lchovli ma’lumotlarni shakllantirishingiz kerak bo‘ladi. Bunda **mpl_toolkits.mplot3d** funksiyasini **Axes3D** sinf ob’ekti sifatida o‘rnatiladi. Uch o‘lchovli grafikani faollashtirishni quyidagi dastur orqali qarab o‘tamiz.

```
from matplotlib . pyplot import *
from mpl_toolkits . mplot3d import Axes3D
fig = figure ()
```


Axes3D (fig)

show ()



In [104]:

Misol. Trigonometrik funksiyalar bo'yicha uch o'lchovli grafik chizish dasturini tuzing.

```
from matplotlib . pyplot import *
import numpy as np
from mpl_toolkits . mplot3d import Axes3D
fig = figure ()
ax = Axes3D ( fig )
u = np . linspace ( 0 , 2 * np . pi , 100 )
v = np . linspace ( 0 , np . pi , 100 )
x = 10 * np . outer ( np . cos ( u ) , np . sin ( v ) )
y = 10 * np . outer ( np . sin ( u ) , np . sin ( v ) )
z = 10 * np . outer ( np . ones ( np . size ( u ) ) , np . cos ( v ) )
ax . plot_surface ( x , y , z , color = ' grey ' )
savefig ( ' 3D . png ' )
show ()
```

Dastur natija 3D.png faylida saqlandi.

Nazariy savollar

- 1 Diagrammalar necha turda shakllantiriladi?
- 2 Ustun shaklidagi diagramma hosil qilish funksiyalari?
- 3 Aylana shaklidagi diagramma hosil qilish funksiyalari?

4 Uch o'lchovli grafiklarni shakllantirish?

Mustaqil ishlash uchun topshiriqlar

1. Ikkita a va b ro'yxat elementlaridan ustunli diagrammalar hosil qiling.
2. Ikkita a va b ro'yxat elementlaridan aylanalik diagrammalar hosil qiling.
3. $[a, b]$ oraliqda kvadrat va kubik funksiyalarning qiymatlarini diagrammalarda ifodalang.
4. Uch o'lchovli sistemada sharni hosil qiling.
5. Uch o'lchovli sistemada bir vaqtning o'zida ikkita shakl hosil qiling.
6. O'zidan oldingisini markazidan o'tuvchi yassi ellipslarni n millisekund oralig'i bilan gorizontal yo'nalish bo'yicha ekranni to'ldirib hosil qiling.
7. O'zidan oldingisini markazidan o'tuvchi yassi ellipslarni n millisekund oralig'i bilan vertikal yo'nalish bo'yicha ekranni to'ldirib hosil qiling.
8. Sektor yordamida 60 millisekund oralig'i bilan vertikal yo'nalish bo'yicha ekranni to'ldirib sektorlar hosil qiling.
9. $[a, b]$ oraliqda sinus funksiyaning qiymatlarini 15 gradus farqi bilan diagrammalarda ifodalang.
10. $[a, b]$ oraliqda kosinus funksiyaning qiymatlarini 5 gradus farqi bilan diagrammalarda ifodalang.
11. $[a, b]$ oraliqda tanginus funksiyaning qiymatlarini 30 gradus farqi bilan diagrammalarda ifodalang.
12. $[a, b]$ oraliqda katangens funksiyaning qiymatlarini 10 gradus farqi bilan diagrammalarda ifodalang.
14. kvadrat shaklini hosil qilish dasturini tuzing.
15. Teng tomonli uchburchak shaklini hosil qilish dasturini tuzing.
16. Teng yonli uchburchak shaklini hosil qilish dasturini tuzing.
17. To'g'ri burchakli to'rtburchak shaklini hosil qilish dasturini tuzing.
18. Ikkita to'plam elementlari va ularning kesishmasini ustunli diagrammalarda hosil qiling.
19. Ikkita kortej elementlari va ularning birlashmasini ustunli diagrammalarda hosil qiling.
20. Ikkita ro'yxat elementlari va ularning kesishmasini aylanalik diagrammalarda hosil qiling.

Xulosa

Qoʻllanmada axborot texnologiyalarining avtomatlashtirilgan elementlarini qoʻllash va avtomatlashtirish asosida yangi axborot texnologiyasini yaratish bosqichini amalga oshirish mexanizmlari keltirib oʻtildi. Python dasturlash tili tarkibidagi barcha turdagi operatorlar, kalit soʻzlar va qoidalar batafsil keltirib oʻtildi. Qoʻllanmada python tilining barcha imkoniyatlari boʻyicha nazariy tushunchalar hamda bu tushunchalarni oʻzlashtirish uchun masalalar yechimlari keltirib oʻtildi. Har bir mavzu boʻyicha mavzuni mustahkamlash uchun nazariy savollar hamda mustaqil ishlash uchun topshiriqlar ham keltirib oʻtildi. Har bir mavzu boʻyicha tayanch iboralar va glossariy qismi shakllantirildi. Mazkur qoʻllanma soʻz boshida belgilangan vazifalarni hal etish uchun, asosiy uslubiy taʼminot vazifasini bajaradi. Bu qoʻllanma nafaqat matematik va muhandis yoʻnalishidagi talabalar, balki mustaqil oʻrganuvchilar uchun ham asosiy qoʻllanma hisoblanadi. Qoʻllanma Amaliy matematika va kompyuter ilmlari va dasturlash texnologiyalari yoʻnalishlarining oʻquv rejasi hamda oʻquv dasturi asosida yaratildi. Tayyorlangan qoʻllanma asosida, ilmiy ishlanmalarni dasturiy tizimlarini yaratish mumkin.

GLOSSARY

Algoritm – masala yoki muommoni hal etish uchun beriladigan chekli sondagi bo‘yruqlar ketma ketligi.

Algoritm xossalari – algoritmni amalga oshiradigan beshta xususiyatlar majmuasi.

Algoritm turlari – algoritmning bajarilishiga ko‘ra uning turlari.

Chiziqli algoritmlar – Algoritmning shartlarsiz chiziqli bajarilishi.

Tarmoqlanuvchi algoritmlar - Algoritmning shartlar asosida bajarilishi.

Takrorlanuvchi algoritmlar - Algoritmning ma‘lum bir qismi ikki va undan ortiq bajarilishi

Axborot tizimi – katta hajmdagi ma‘lumotlarni saqlash, uzatish va qayta ishlash dasturi.

Quyi darajadagi til – kompyuter qurilmalari bilan bog‘liq bo‘lib, buyruqlar ularning kodlari bilan yoziladi.

O‘rta darajadagi til - buyruqlarida faqat raqamlar emas, balki insonlar tushunadigan ba‘zi so‘zlar ishlatila boshlandi.

Yuqori darajadagi til – translayotorlar yordamida mashina tiliga o‘tkaziladigan til.

Translayator - mashina tiliga kodni tarjima qiladi.

Komplyator – mashina tiliga kodni to‘liq tarjima qiladi.

Interpretator – mashina tiliga kodni qism qilib tarjima qiladi.

IDLE rejimi – Python tilining ish rejimi.

Interaktiv - Python tilining qismaniy faol ish rejimi.

Faylli dastur – alohida nom bilan saqlanuvchi tuzilgan dastur.

ENTER – faollashtirish tugmasi.

Restart - Python tilining natijasini ko‘rsatuvchi kalitli so‘z.

Int – butun tur.

Float – haqiqiy tur.

Str- satrli tur.

Ifoda – matematik amallar bilan birlashtirilgan yozuv.

Import – kutubxonlarni chaqirishning kalitli so‘zi.

Math – Matematik kutubxona.

Standart funksiya – dasturlash tili tarkibidagi funksiyalar.

Min – minimum funksiya.

Max – maksimum funksiya.
Sum – summator funksiya
Type – turni aniqlovchi funksiya.
Help – ma'lumot beruvchi funksiya.
Operator – malum bir bo'yruqli kalit so'z.
Chiziqli dastur –chiziqli algorit asosida tuzilgan dastur
If - mantiqiy kalitli so'z.
Else – aks holda qismi uchun kalitli so'z.
Elif – aks holdani ichidagi shart qismi uchun kalitli so'z.
For -parametrlil sikl operatorining kalitli so'zi.
Sikl – takrorlanish.
Parametrlil takrorlanish – ma'lum bir parametr asosidagi takrorlanish.
Break operatori –majburiy to'xtatish bo'yruq'i.
Continue – qadamni tashlash bo'yruq'i
While - shartli sikl operatorining kalitli so'zi.
Qism dasturlar –dastur tarkibidagi kichik dastur
Funksiya – bir yoki ko'p qiymat qaytaruvchi funksiya.
Global – butun dastur davomida faol hisoblanadi.
Lokal – dasturning ma'lum bir qismi faol hisoblanadi.
Argument – funksiya qabul qiladigan qiymat.
Def – funksiyani faollashtirish so'zi.
Return –qiymat qaytarish kalitli so'zi.
Rekursiya – o'z o'ziga murojaat.
Rekursiv funksiya - o'z o'ziga murojaat qiluvchi funksiya.
Ro'yxat - Har xil turga mansub bo'lgan yagona nom bilan saqlanuvchi tartiblangan ma'lumotlar majmuasi.
Kortej- o'zgartirilmaydigan ro'yxat.
Dinamik xotira –o'zgaruvchan xotira.
Sorted – saralash funksiyasi.
Append – ma'lumot qo'shish funksiyasi.
Del - ma'lumot o'chirish funksiyasi.
Numpy – massiv kutubxonasi.
Fayl – alohida nom bilan saqlanuvchi ma'lumotlar majmuasi.
Open – faylni ochish bo'yruq'i.
Read – fayldan ma'lumot o'qish bo'yruq'i.

Write – faylga ma'lumot yozish bo'yruqi.

Matplotlib - grafik kutubxonasi.

Plot – grafik chizish funksiyasi.

Savefig – grafikli ma'lumotni saqlash bo'yruqi

Ylabel – Oy koordinatalar sistemasiga matn joylashtirish.

Xlabel – Ox koordinatalar sistemasiga matn joylashtirish.

Title – koordinatalar sistemasiga matn joylashtirish.

Color – grafikga rang berish.

Subplot – diagrammalarni chizish funksiyasi.

Mplot3d – 3d grafika kutubxonasi.

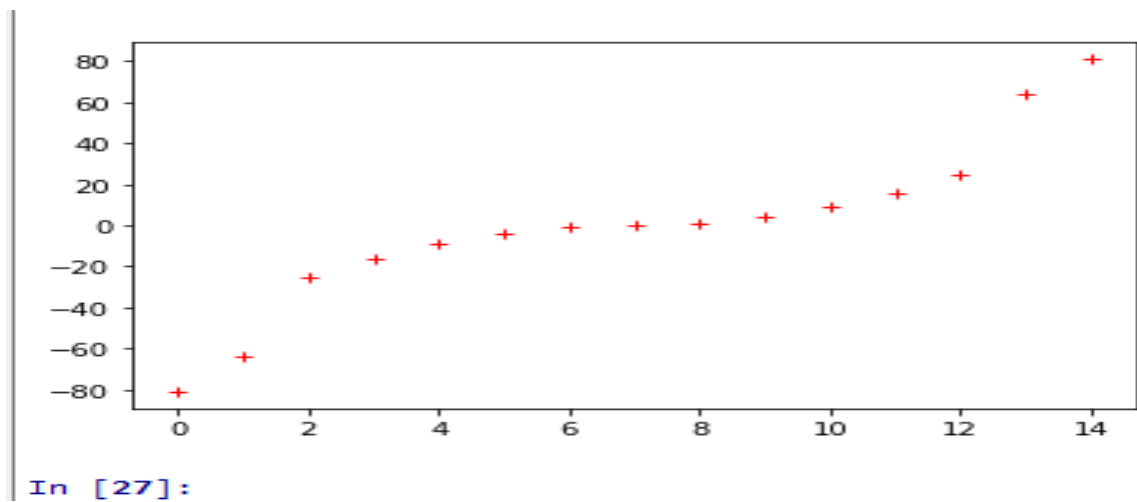
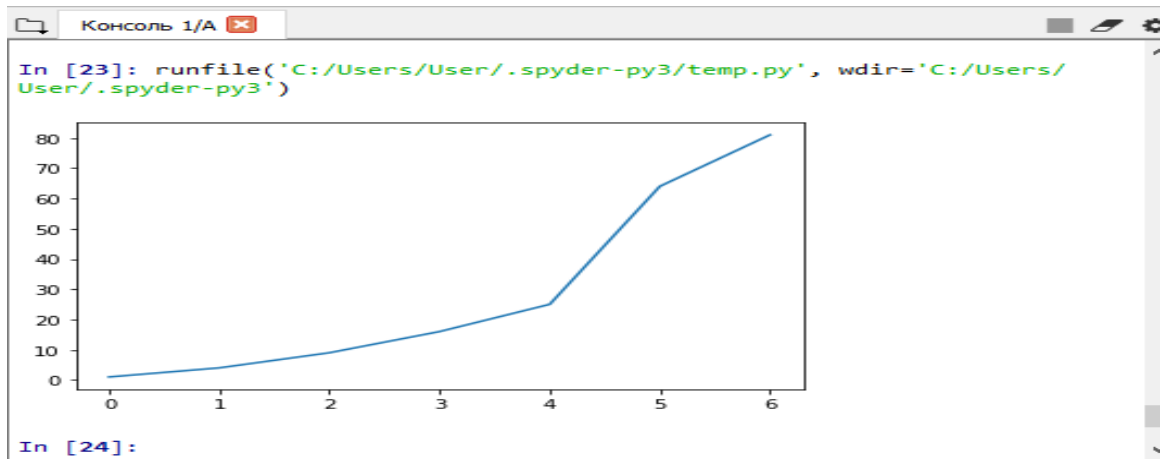
FOYDALANILGAN ADABIYOTLAR RO‘YXATI

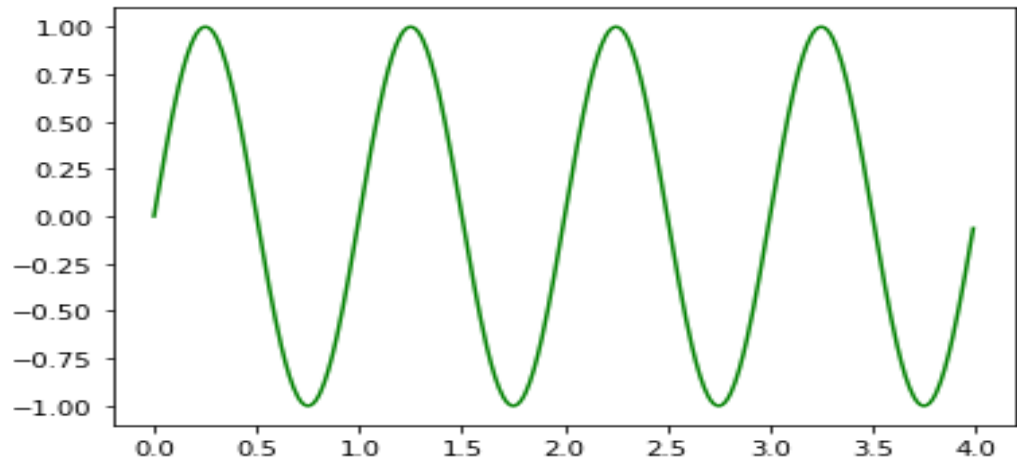
1. Eric Matthes. Python Crash Course Paperback. England 2015. 205p.
2. Krishna Rungta. Learn Python in 1 Day: Complete Python Guide with Examples. India 2016. -182 p.
3. Narasimha Karumanchi. Data Structure and Algorithmic Thinking with Python Paperback. India 2015. 170p.
4. Сысоева М.В., Сысоев И. В. Программирование для «нормальных» с нуля на языке Python Москва. 2018. -180с.
5. Федоров Д. Ю. Основы программирования на примере языка Python. Санкт-Петербург 2018. -167 с.
6. Eshtemirov S. Nazarov F. Algoritmlash va dasturlash asoslari. O‘quv qo‘llanma. Samarqand 2019. -208 b.
7. Nazarov F. C++ tilida dasturlash asoslari. Uslubiy qo‘llanma. Samarqand 2017. -208 b.
8. Вирт Н. Алгоритмы + структуры данных = программа.- М.:Мир,1985.-405с.
9. Информатика. Базовой курс. Учебник для Вузов., Санкт-Петербург,2001. под редакцией С.В.Симоновича.
10. Непейвода Н.Н, Скоплин И.Н. Основания программирования. -Москва Ижевск: Институт компьютерных исследований, 2003 г. 864 с.
11. Лойко В.Л. Структуры и алгоритмы обработки данных. Учебное пособие для вузов.- Краснодар: КубГАУ. 2000. - 261 с.
12. Алфред В. Хоп Крофт, Джефри Д. Ульман. Структуры данных и алгоритмов. Издательский дом «Вильямс» Москва — Санкт-Петербург - Киев, 2003-384 с.

Поляр

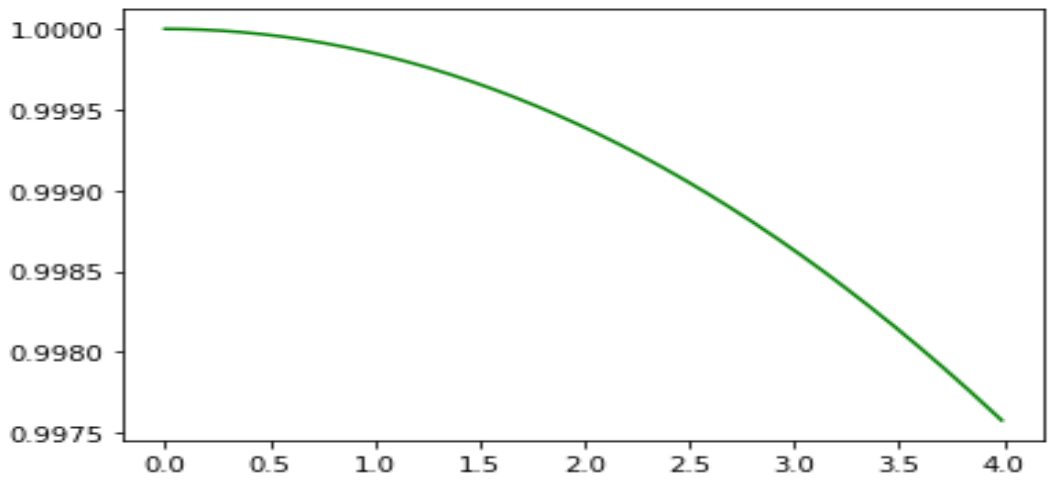
```
Python 3.7.1 Shell
File Edit Shell Debug Options Window Help
Python 3.7.1 (v3.7.1:260ec2c36a, Oct 20 2018, 14:57:15) [MSC v.1915 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> |
```

```
test — Блокнот
Файл Правка Формат Вид Справка
1 2 3 4 5
```

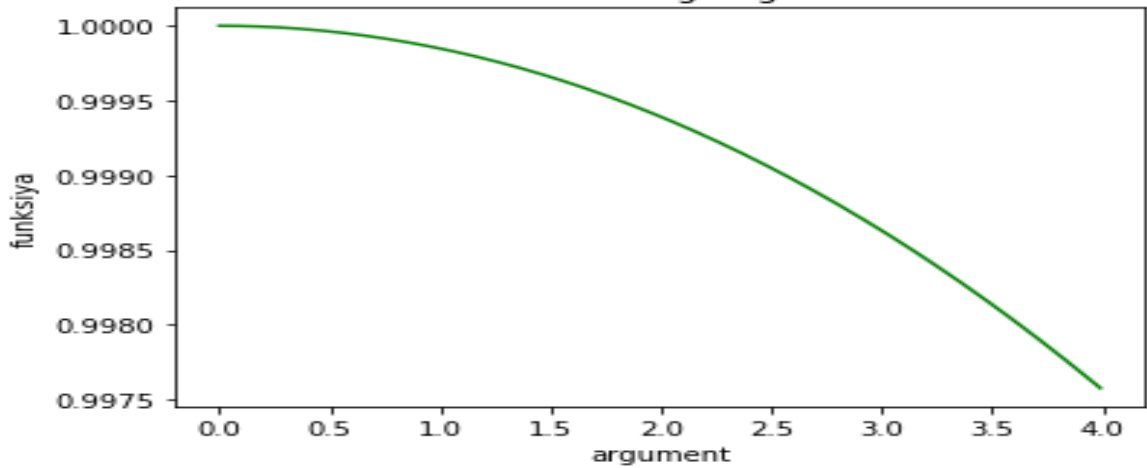




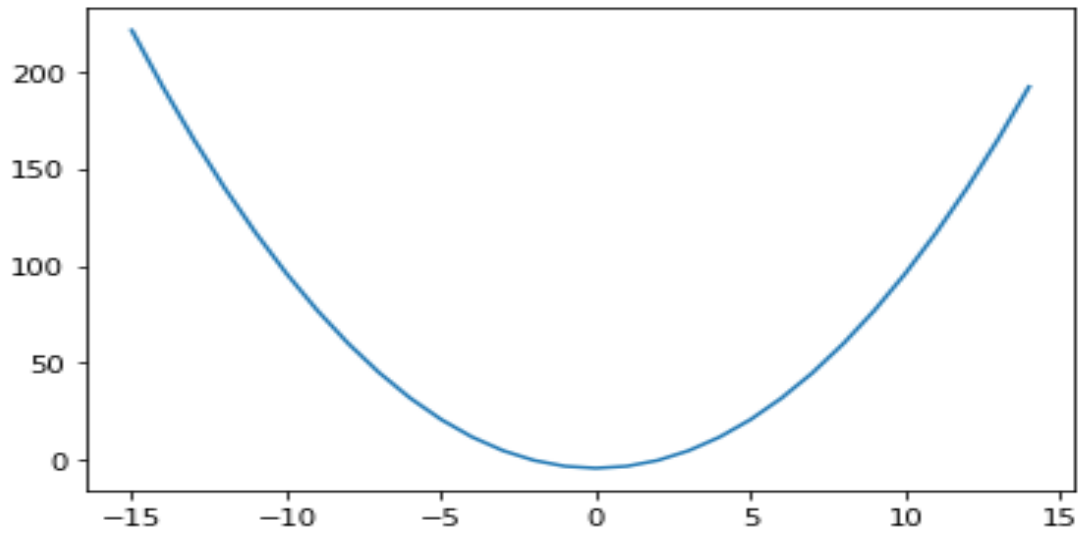
In [44]:



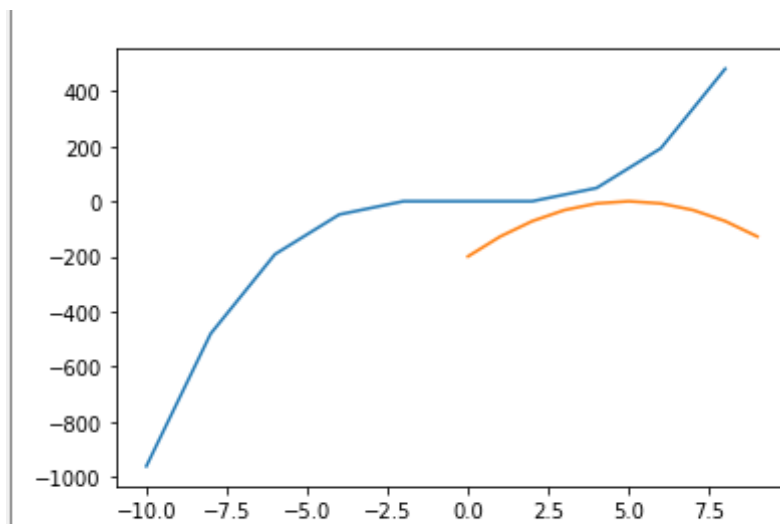
Kosinus grafiği



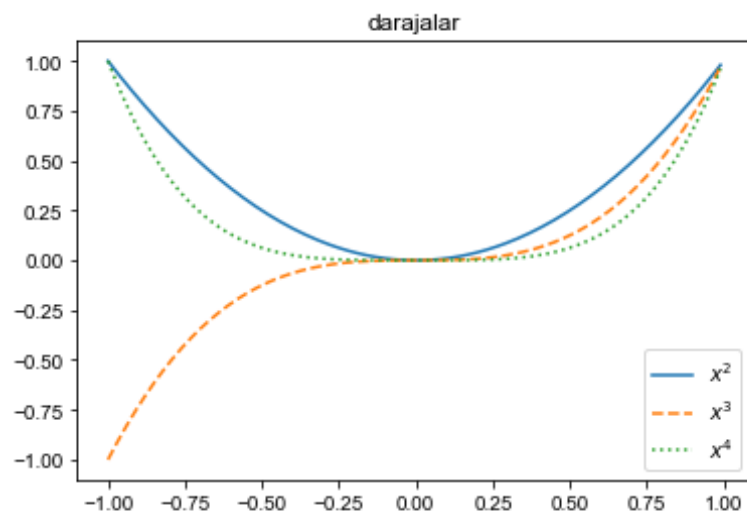
In [54]:



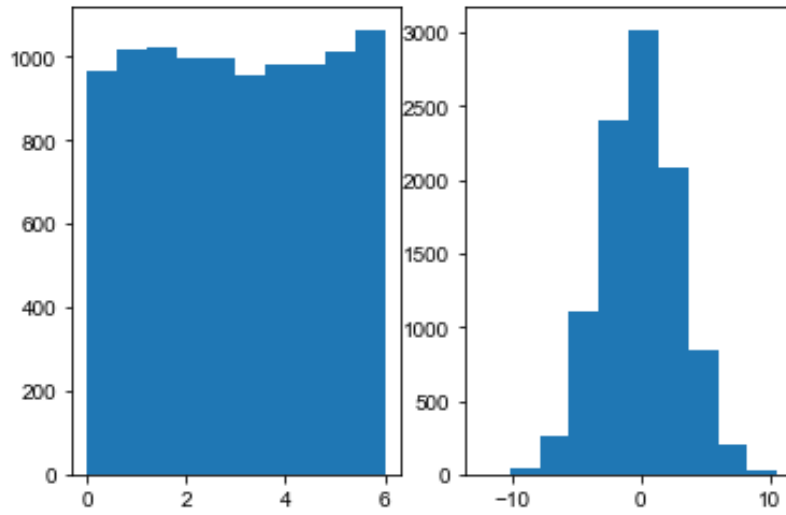
In [66]:



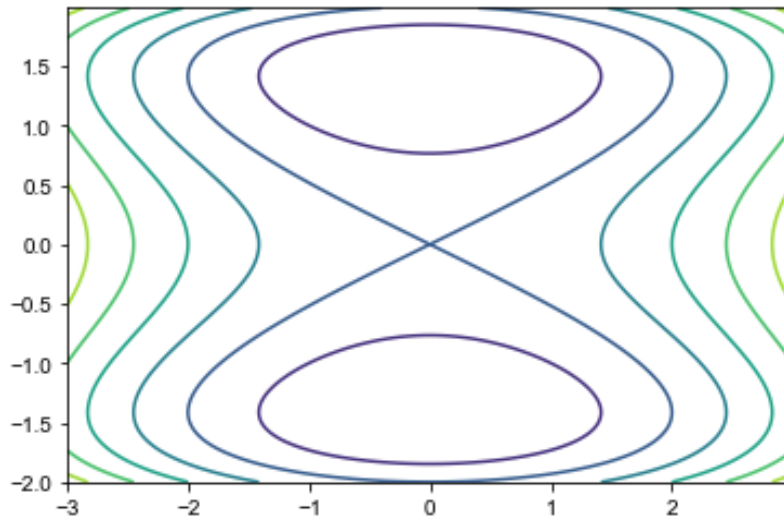
In [74]:



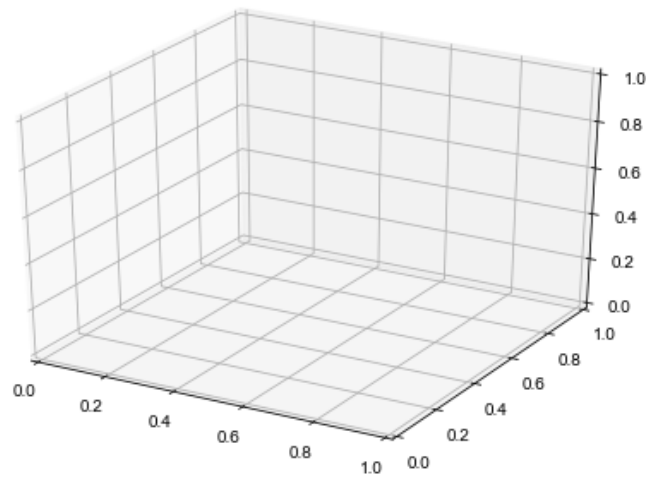
In [89]:



In [99]:



In [102]:



In [104]:

AXATOV AKMAL, NAZAROV FAYZULLO

**PYTHON
TILIDA DASTURLASH ASOSLARI
(I-QISM)**

*(Amaliy matematika, kompyuter ilmlari va dasturlash texnologiyalari
yoʻnalishlari uchun oʻquv qoʻllanma)*

Muharrir
Musahhah
Texnik muharrir

J.Bozorova
L.Xoshimov
B.Egamberdiyev

ISBN 978-9943-6646-8-5

2020-yil 20 noyabrda tahririy-nashriyot boʻlimiga qabul qilindi.
2020-yil 18 dekabrda original-maketdan bosishga ruxsat etildi.
Qogʻoz bichimi 60x84. “Times New Roman” garniturasini.
Offset qogʻoz. Shartli bosma tabogʻi – 11,25.
Adadi 50 nusxa. Buyurtma № 346

SamDU tahririy-nashriyot boʻlimida chop etildi.
140104, Samarqand sh., Universitet xiyoboni, 15.





Axatov Akmal Rustamovich 1974 yil 18 – dekabr kuni Samarqand viloyati Pstdarg‘om tumanining Juma shaxrida ziyoli oilada tavallud topgan. Axatov Akmal 3 ta o‘quv qo‘llanma, 10 dan ortiq uslubiy qo‘llanma, 100 dan ortiq ilmiy ommobob maqolalar va 10 ga yaqin dasturiy maxsulotlar uchun guvoxonmalar muallifi hisoblanadi. Hozirgi vaqtda O‘zMUJF ilmiy va innovatsiyalar bo‘yicha direktor o‘rinbosari lavozimida faoliyat olib bormoqda. A.R.Axatov texnika fanlar doktori, professor, tel:+998902716418, akmalar@rambler.ru



Nazarov Fayzullo Maxmadiyarovich 1990 yil 18 – avgust kuni Samarqand viloyati Urgut tumanining Taroqli mahallasida ziyoli oilada tavallud topgan. Nazarov Fayzullo Maxmadiyarovich 2 ta o‘quv qo‘llanma, 4 ta uslubiy qo‘llanma, 50 dan ortiq ilmiy ommobob maqolalar va 10 ga yaqin dasturiy maxsulotlar uchun guvoxonmalar muallifi hisoblanadi. Hozirgi vaqtda Samarqand davlat universiteti tizimli tahlil, boshqaruv va axborotlarni qayta ishlash ixtisosligi bo‘yicha tayanch doktoranti hisoblanadi. tel:+998944798640, fayzullo-samsu@mail.ru