**AMALIY MASHG‘ULOT UCHUN O‘QUV MATERIALLARI**

**1-Mavzu:** “Python dasturlash tili” faniga kirish va asosiy tushunchalari.

**5-mashg‘ulot.** Pythonda takrorlanuvchi jarayonlarni dasturlash.

**O‘quv savollari:**

1. Sikl operatorlari – For va while bilan ishlash.
2. Break, continue va else operatorlarining qo‘llanilishi.

**1. Sikl operatorlari – for va while bilan ishlash.**

Sikllar biror bir shartning bajarilishiga qarab ba’zi takroriy harakatlarni bajarishga imkon beradi. Pythonda quyidagi turdagi sikllar mavjud: **while**va **for.**

**WHILE sikli.** while sikli qandaydir bir shartning to‘g‘riligini tekshiradi va agar shart True bo‘lsa, u holda sikl tanasidagi blokni bajaradi. U quyidagicha sintaksisga ega:

**while** <shartli\_ifoda>:

[sikl tanasi]

**while** kalit so‘zidan keyin shartli ifoda keladi va bu ifoda True qiymat qaytarsa, sikl tanasiga kirib u yerdagi operatorlar ketma-ket bajariladi.

**while** sikliga tegishli barcha bloklar oldidan bo‘sh joy qoldirilib ketishi va boshlanish qismlari bir ustunga joylashgan bo‘lishi kerak va while kalit so‘zidan bir xilda chekingan bo‘lishi kerak.

son = 1

**while** son < 5:

print(f"son = {son}")

son += 1

print("Dastur tugatildi")

Bunday holda, son o‘zgaruvchisi 5 dan kichik bo‘lsa, while sikli ishlaydi. Aks holda sikldan keyingi amal bajariladi.

Bu sikl tanasi ikkita ifodadan iborat:

print(f"son = {son}")

son += 1

Shuni ham yodda tutingki, oxirgi print(“Dastur tugallandi”) satr boshidan chekinmaydi, shuning uchun u while siklining bir qismi emas.

*Siklning butun jarayonini quyidagicha ifodalash mumkin:* Birinchi, **son** o‘zgaruvchisining qiymati 5 dan kichik yoki kichik emasligini tekshiriladi. Va o‘zgaruvchi dastlab 1 ga teng bo‘lganligi sababli, bu shart True ni qaytaradi va shuning uchun sikl operatorlari bajariladi.

Sikl ko‘rsatmalari satr **son**=1 ni konsolga chiqaradi. Va keyin raqam o‘zgaruvchisining qiymati bittaga oshiriladi - endi u 2 ga teng bo‘ladi. sikl ko‘rsatmalari blokini bir marta bajarish iteratsiya deb ataladi. Ya’ni, shu tarzda, birinchi takrorlash siklda amalga oshiriladi.

**son** **< 5** sharti yana tekshiriladi. Yana True qaytariladi, chunki son 2 ga teng, shuning uchun sikl operatorlari yana bir marta bajariladi.

Sikl ko‘rsatmalari konsolga satr **son** = 2 ni chiqaradi. Va keyin **son** o‘zgaruvchisining qiymati yana bittaga oshiriladi - endi u 3 ga teng bo‘ldi. Shunday qilib, ikkinchi takrorlash amalga oshiriladi.

**son < 5** shart yana tekshiriladi. Yana True qiymat qaytariladi, chunki son = 3, shuning uchun sikl operatorlari bajariladi.

Sikl ko‘rsatmalari konsolga satr **son** = 3 ni chiqaradi. Va keyin **son** o‘zgaruvchisining qiymati yana bittaga oshiriladi - endi u 4 ga teng. Ya’ni uchinchi iteratsiya amalga oshiriladi.

**son < 5** sharti yana tekshiriladi. Yana True qiymat qaytariladi, chunki son = 4, shuning uchun sikl ko‘rsatmalari bajariladi.

Sikl ko‘rsatmalari konsolga son= 4 qator raqamini chiqaradi.Va keyin son o‘zgaruvchisining qiymati yana bittaga oshiriladi - endi u 5 ga teng. Ya’ni to‘rtinchi takrorlash amalga oshiriladi.

  Va shart son < 5 yana tekshiriladi. Lekin endi u False qiymat qaytaradi, chunki son o‘zgaruvchisining qiymati endi 5 ga teng, shuning uchun sikl tugatiladi. Shundan so‘ng sikldan keyin operatorlar bajariladi. Shunday qilib, bu sikl to‘rtta o‘tish yoki to‘rtta takrorlashni amalga oshiradi

 Natijada, kodni bajarishda biz quyidagi konsol chiqishini olamiz:

son = 1

son = 2

son = 3

son = 4

Dastur tugatildi

While sikli bilan yana bir oprator – else ham ishlatiladi. U while siklidagi shart False qiymat qaytarganida ishga tushadi va tarkibidagi blok bajariladi:

raqam = 1

**while** raqam <5:

print(f"raqam = {raqam}")

raqam += 1

**else**:

print(f"raqam = {raqam}. sikl tugallandi")

print("Dastur tugatildi")

Ya’ni, bu holda birinchi navbatda shart tekshiriladi va while operatorlari bajariladi. Keyin shart False ga aylanganda else blokidagi gaplar bajariladi. E’tibor bering, else blokidagi iboralar ham sikl konstruktsiyasining boshidan chekinadi. Natijada, bu holda biz quyidagi konsol chiqishini olamiz:

raqam = 1

raqam = 2

raqam = 3

raqam = 4

raqam = 5. sikl tugallandi

Dastur tugadi

**For sikli.** Sikl ning yana bir turi for konstruktsiyasidir. Ushbu sikl qiymatlar to‘plamini takrorlaydi, har bir qiymatni o‘zgaruvchiga qo‘yadi va keyin siklda biz ushbu o‘zgaruvchi bilan turli harakatlarni bajarishimiz mumkin. For siklining strukturasi quyidagicha:

**for** [o‘zgaruvchi] in [qiymatlar\_to‘plami]:

[sikl tanasi]

For kalit so‘zidan keyin qiymatlar joylashtiriladigan o‘zgaruvchining nomi keladi. Keyin in operatoridan keyin qiymatlar to‘plami va ikki nuqta qo‘yiladi.

Va keyingi qatordan sikl tanasidagi operatorlar boshlanadi. Bu operatorlar siklning bir qismi ekanligi uchun ular for kalit so‘zi boshidan ichkariga kiritilishi kerak.

Siklni bajarishda Python ketma-ket barcha qiymatlarni to‘plamdan oladi va ularni o‘zgaruvchiga uzatadi. To‘plamdagi barcha qiymatlar takrorlangandan keyin sikl tugaydi.

Masalan, qiymatlar to‘plami sifatida siz aslida belgilar to‘plamini ifodalovchi satrni ko‘rib chiqishingiz mumkin. Keling, bir misolni ko‘rib chiqaylik :

string = "Hello"

**for** char **in** string:

print(char)

**char** o‘zgaruvchisi siklda aniqlanadi, **in** operatoridan keyin **string** o‘zgaruvchisi "Hello" satrini saqlaydigan takrorlangan to‘plam sifatida ko‘rsatiladi. Natijada, for sikli string tarkibidagi barcha belgilarni ketma-ketlikda takrorlaydi va ularni **char** o‘zgaruvchisiga o‘zlashtiradi. Sikl blokining o‘zi **char** o‘zgaruvchining qiymatini konsolga chop etadigan bitta funksiyadan iborat. Dasturning konsol chiqishi:

H

e

l

l

o

For siklida sikl tugagandan keyin bajariladigan qo‘shimcha else bloki ham bo‘lishi mumkin:

string = "Hello"

**for** char **in** string:

print(char)

**else**:

print(f"oxirgi simvol: {char}

print("Dastur yakunlandi")

Bunday holda, biz quyidagi konsol chiqishini olamiz:

H

e

l

l

o

oxirgi simvol: o

Dastur yakunlandi

Shuni ta’kidlash kerakki, else bloki for siklida aniqlangan barcha o‘zgaruvchilarga kirish huquqiga ega.

**Ichma-ich sikllar.** Ba’zi sikllar o‘zlarida boshqa sikllarni o‘z ichiga olishi mumkin. Ko‘paytirish jadvalining chiqishi misolini ko‘rib chiqing:

a = 1

b = 1

while a < 10:

while b < 10:

print(a \* b, end="\t")

b += 1

print("\n")

b = 1

a += 1

Tashqi sikl while a < 10: a o‘zgaruvchisi 10 ga teng bo‘lgunga qadar 9 marta ishlaydi. Bu sikl ichida ichki sikl while b < 10: ishlaydi. b o‘zgaruvchisi 10 ga teng bo‘lgunga qadar ichki sikl ham 9 marta ishlaydi. Bundan tashqari, ichki siklning barcha 9 ta takrorlanishi tashqi siklning bir iteratsiyasi ichida ishga tushiriladi.

Ichki halqaning har bir iteratsiyasida konsolda a va b raqamlarining mahsuloti ko‘rsatiladi. Keyin b o‘zgaruvchining qiymati bittaga oshiriladi. Ichki sikl o‘z ishini tugatgandan so‘ng, b o‘zgaruvchining qiymati 1 ga o‘rnatiladi va a o‘zgaruvchining qiymati bittaga oshiriladi va tashqi siklning keyingi takrorlanishiga o‘tish sodir bo‘ladi. Va a o‘zgaruvchisi 10 ga teng bo‘lgunga qadar hamma narsa takrorlanadi. Shunga ko‘ra, ichki sikl tashqi siklning barcha iteratsiyasi uchun faqat 81 marta ishlaydi. Natijada, biz quyidagi konsol chiqishini olamiz:

1          2        3        4        5        6        7        8        9

2          4        6        8        10      12      14      16      18

3          6        9        12      15      18      21      24      27

4          8        12      16      20      24      28      32      36

5          10      15      20      25      30      35      40      45

6          12      18      24      30      36      42      48      54

7          14      21      28      35      42      49      56      63

8          16      24      32      40      48      56      64      72

9          18      27      36      45      54      63      72      81

for siklini xuddi shunday aniqlanishi mumkin:

**for** c1 **in** "ab":

**for** c2 **in** "ba":

        print(f"{c1}{c2}")

Bunday holda, tashqi sikl "ab" qatoridan o‘tadi va har bir belgini c1 o‘zgaruvchisiga qo‘yadi. Ichki sikl "ba" satrini takrorlaydi, satrdagi har bir belgini c2 o‘zgaruvchisiga qo‘yadi va ikkala belgi kombinatsiyasini konsolga chiqaradi. Ya’ni, oxirida biz a va b belgilarning barcha mumkin bo‘lgan kombinatsiyalarini olamiz:

ab

aa

bb

ba

**2. Break, continue va else operatorlarining qo‘llanilishi**

Biz siklni boshqarish uchun maxsus break va continue operatorlaridan foydalanishimiz mumkin. Break operatori siklni to‘xtatadi. continue operatori esa siklning keyingi iteratsiyasiga o‘tkazadi.

Agar siklda uning keyingi bajarilishiga mos kelmaydigan shartlar hosil bo‘lsa, break operatoridan foydalanish mumkin. Quyidagi misolni ko‘rib chiqing :

raqam = 0

**while** raqam <5:

raqam += 1

**if** raqam == 3: # agar raqam = 3 bo‘lsa,sikldan chiqamiz

break

print(f"raqam = {raqam}")

Bu yerda while sikli raqam<5 shartini rostlikka tekshiradi. Va agar raqam o‘zgaruvchisining qiymati hozircha 5 ga teng bo‘lmasa, raqamning qiymati konsolda chop etiladi. Ammo ungacha sikl ichida yana bir shart ham tekshiriladi: if raqam == 3. Ya’ni **raqam**ning qiymati 3 ga teng bo‘lsa, u holda break operatori yordamida sikldan chiqamiz. Va oxirida biz quyidagi konsol chiqishini olamiz:

raqam = 1

raqam = 2

**Break** operatoridan farqli o‘laroq, **continue** operatori siklning keyingi iteratsiyasiga uni tugatmasdan o‘tadi. Misol uchun, oldingi misolda break ni continue bilan almashtiramiz:

raqam = 0

**while** raqam <5:

raqam += 1

**if** raqam == 3: # agar raqam = 3 bo‘lsa, siklning yangi iteratsiyasiga o‘ting

continue

print(f" raqam = {raqam}")

Va bu holda, agar raqam o‘zgaruvchisining qiymati 3 ga teng bo‘lsa, continue operatori siklni keyingi aylanishga o‘tkazib yuboradi va navbatdagi qadam amalga oshiriladi:

raqam = 1

raqam = 2

raqam = 4

raqam = 5

**Nazorat savollari :**

1. Dasturlashda takrorlanuvchi jarayonlarni dasturlash qanday ro‘l egallaydi?
2. Takrorlanuvchi jarayonlarni qanday operatorlar yordamida dasturini tuziladi?
3. **for** operatorining ishlash prinsipi qanday?
4. **while** operatorining sintaksisi va ishlash prinsipi qanday?
5. **for** va **while** operatorlarining bir biridan asosiy farqlari nimada?
6. **else** operatorini siklik jarayonlarda nima maqsadda foydalaniladi?
7. **range** operatorining vazifasi va ishlash prinsipi qanday?
8. **break** operatorining vazifasi nima?
9. **continue** operatori qanday holatlarda ishlatiladi?