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

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

**6-mashg‘ulot.** Pythonda roʻyxatlar bilan ishlash.

**O‘quv savollari:**

1. Ro‘yxatlar va ularning qo‘llanilishi.
2. Ro‘yxatlarni yaratish usullari.
3. Ro‘yxatlar bilan ishlovchi metodlar.

**1. Ro‘yxatlar va ularning qo‘llanilishi.**

Ma’lumotlar to‘plamlari bilan ishlash uchun Python ro‘yxat, kortej va lug‘atlar kabi o‘rnatilgan turlarni taqdim etadi.

**Ro‘yxat (list)** - elementlar to‘plami yoki ketma-ketligini saqlaydigan ma’lumotlar turi. Ko‘pgina dasturlash tillarida massiv deb ataladigan o‘xshash ma’lumotlar tuzilmasi mavjud.

**2. Ro‘yxatlarni yaratish usullari**

Ro‘yxatni yaratish uchun kvadrat qavslar [] ishlatiladi. Ularning ichida ro‘yxat elementlari vergul bilan ajratilgan holda saqlanadi. Masalan, raqamlar ro‘yxatini yarataylik:

raqamlar = [0, 2, 12, 44, 51]

Xuddi shunday, boshqa turdagi ma’lumotlar bilan ro‘yxatlarni belgilash mumkin, masalan, satrlar (ya’ni string turdagi ma’lumotlar) ro‘yxatini yaratish quyidagicha amalga oshiriladi:

odamlar = ["Jasur", "Akmal", "Bobur"]

Roʻyxat yaratish uchun **list()** konstruktor funksiyasidan ham foydalanish mumkin **:**

raqamlar1 = **[]**

raqamlar2 = **list()**

Ushbu ikkala ro‘yxat ta’riflari o‘xshash - ular bo‘sh ro‘yxat yaratadi.

Ro‘yxatda faqat bir xil turdagi ob’ektlar bo‘lishi shart emas. Bir vaqtning o‘zida satrlarni, raqamlarni va boshqa ma’lumotlar turlarining ob’ektlarini bir xil ro‘yxatga joylashtirish mumkin:

obyektlar = [1, 2.26, "Salom dunyo", True, False]

Roʻyxat elementlarini ekranga chiqarish uchun standart print funksiyasidan foydalanish mumkin. Bu roʻyxat tarkibini to‘liq shaklda konsolga chop etadi:

raqamlar = [1, 2, 3, 4, 5]

odamlar = ["Jasur", "Akmal", "Bobur"]

print(raqamlar)*# [1, 2, 3, 4, 5]*

print(odamlar)    *# ["Jasur", "Akmal", "Bobur"]*

**list()** konstruktori qiymatlar to‘plamini qabul qilishi mumkin, ular asosida ro‘yxat tuziladi:

raqamlar1 = [1, 2, 3, 4, 5]

raqamlar2 = **list**(raqamlar1)

print(raqamlar2)   *# [1, 2, 3, 4, 5]*

harflar = **list**("Salom")

print(harflar)       *# [‘S’, ‘a’, ​​’l’, ‘o‘, ‘m’]*

Agar bir xil qiymat bir necha marta takrorlanadigan ro‘yxatni yaratish kerak bo‘lsa, unda yulduzcha - \* belgisidan foydalanish mumkin, ya’ni aslida mavjud ro‘yxatga ko‘paytirish amalini qo‘llaniladi:

raqamlar = [15]\*6      *# 15 marta 6 takrorlash*

print(raqamlar)       *# [15, 15, 15, 15, 15, 15]*

odamlar = ["Jasur"]**\*3**  *# "Tom" ni 3 marta takrorlash*

print(odamlar)        *# ["Jasur", "Jasur", "Jasur"]*

talabalar = ["Bobur", "Saman"] **\* 2**

*# "Bobur", "Saman" ni 2 marta takrorlash*

print (talabalar)

*# ["Bobur", "Saman", "Bobur", "Saman"]*

**Ro‘yxat elementlariga murojaat qilish.** Ro‘yxat elementlariga murojaat qilish uchun ro‘yxatdagi element sonini ifodalovchi indekslardan foydalanish kerak. Indekslar noldan boshlanadi. Ya’ni, birinchi element indeksi 0, ikkinchi element indeksi 1 ga teng bo‘ladi va hokazo. Elementlarga teskari tartibda ya’ni oxiridan boshiga qarab murojaat qilish uchun manfiy indeks tushunchasi kiritilgan. Bu manfiy indekslar -1 dan boshlanadi. Ya’ni, oxirgi element -1 indeksiga ega bo‘ladi, oxirgidan bitta oldingisi esa -2 indeksiga ega bo‘ladi va hokazo.

odamlar = ["Jasur", "Samat", "Bobur"]

*# ro‘yxat boshidan elementlarni olish*

print(odamlar[0])    # Jasur

print(odamlar[1])    # Samat

print(odamlar[2])    # Bobur

*# roʻyxat oxiridagi elementlarni olish*

print(odamlar[-2])    *# Samat*

print(odamlar[-1])    *# Bobur*

print(odamlar[-3])    *# Jasur*

Ro‘yxat elementini o‘zgartirish uchun unga yangi qiymat yuklash kifoya:

odamlar = ["Jasur", "Samat", "Bobur"]

odamlar[1] = "Akmal"   *# ikkinchi elementni o‘zgartirish*

**print**(odamlar[1])     *# Akmal*

**print**(odamlar)        *# ["Jasur", " Akmal", "Bobur"]*

**Ro‘yxat elementlarini ajratish.** Python sizga ro‘yxatni alohida elementlarga ajratish imkonini beradi:

odamlar = ["Jasur", "Bobur", "Samat"]

jasur, bobur, samat = odamlar

**print** (jasur)       *# Jasur*

**print** (bobur)       *# Bobur*

**print** (samat)       *# Samat*

Bunday holda, jasur, bobur va samat o‘zgaruvchilari odamlar ro‘yxatidan ketma-ket tayinlangan elementlardir. Biroq, o‘zgaruvchilar soni tayinlangan ro‘yxat elementlari soniga teng bo‘lishi talab qilinadi. Aks xolda xatolik kelib chiqadi.

Ro‘yxat elementlarini olish uchun for yoki while tsiklidan foydalanish mumkin.

For tsikli bilan takrorlash:

odamlar = ["Tomson", "Sems", "Bob"]

**for** odam **in** odamlar:

**print** (odam)

Bu yerda odamlar ro‘yxati takrorlanadi va har bir element shaxs o‘zgaruvchisiga joylashtiriladi.

Iteratsiyani while tsikli bilan ham bajarish mumkin:

odamlar = ["Tomson", "Sems", "Bob"]

i = 0

**while** i < **len**(odamlar):

**print**(odamlar[i])

# elementni olish uchun indeksdan foydalanish

    i += 1

**len()** funktsiyasidan foydalanib takrorlash uchun ro‘yxat uzunligini olinadi. Bu dasturda i - hisoblagich qiymati ro‘yxat uzunligiga teng bo‘lguncha chop etadi. i ning qiymati esa har safar birga oshirilib boraveradi.

**Ro‘yxatni taqqoslash.** Ikki ro‘yxat, agar ular bir xil elementlar to‘plamini o‘z ichiga olsa, teng hisoblanadi:

raqamlar1 = [1, 2, 3, 4, 5]

raqamlar2 = list ([1, 2, 3, 4, 5])

**if** raqamlar1 == raqamlar2:

**print**("1 - raqamlar 2 - raqamlarga teng")

**else**:

**print**("1-raqamlar 2-raqamlarga teng emas")

Bunday holda, ikkala ro‘yxat ham teng bo‘ladi.

**3. Ro‘yxatlar bilan ishlovchi metodlar**

Ro‘yxat elementlarini boshqarish uchun bir qancha metodlar mavjud. Ulardan asosan ko‘p marta murojaat qilinadiganlarini quyida keltirilgan:

***append(element):*** **element** - ni roʻyxat oxiriga qoʻshish;

***insert (indeks, element):*** ro‘yxatga ***element***ni ***indeks*** – element qilib qo‘shish;

***extend (items):*** roʻyxat oxiriga ***items*** toʻplamini qoʻshish;

***remove(item):*** ro‘yxatdan ***item*** elementini olib tashlaydi. Ro‘yxatdan birinchi uchragan ***item*** elementi o‘chiriladi. Agar element topilmasa, ValueError xatoligi qaytariladi;

***clear():*** roʻyxatdagi barcha elementlarni oʻchirish;

***index(item):*** ***item*** elementining indeksini qaytaradi. Agar element topilmasa, ValueError xatoligi qaytariladi;

***pop([indeks]):*** ***indeks*** indeksidagi elementni olib tashlaydi va qaytaradi. Agar indeks ko‘rsatilmay faqat pop() holida foydalanilsa, oxirgi elementni olib tashlanadi.

***count(item):*** ro‘yxatdagi ***item*** elementining takrorlanishlar sonini qaytaradi. Agar mavjud bo‘lmas, 0 qaytaradi;

***sort([key]):*** ro‘yxat elementlarini tartiblaydi. Standart bo‘yicha o‘sish tartibida saralanadi. Lekin ***key*** parametr yordamida saralashni qanday tartibda bo‘lishini boshqarish mumkin.

***reverse():*** Roʻyxatdagi barcha elementlarni teskari tartibda qayta joylashtiradi.

***copy():*** roʻyxatni nusxalaydi.

Bundan tashqari, Python ro‘yxatlar bilan ishlash uchun bir qator standart funktsiyalarni taqdim etadi:

***len(list):*** ro‘yxat uzunligini aniqlash;

***sorted(list, [key]):*** tartiblangan roʻyxatni qaytaradi;

***min(list):*** ro‘yxatning eng kichik elementini topish;

***max(ro‘yxat):*** ro‘yxatdagi eng katta elementni topish.

**Ro‘yxatga element qo‘shish uchun append()** , **extension** va **insert** metodlari, ro‘yxatdan elementni olib tashlash uchun **remove()**, **pop()** va **clear()** usullari qo‘llaniladi.

**Foydalanish usullari:**

list1= ["Timur", "Bobur"]

*# roʻyxat oxiriga qoʻshish*

list1.**append**("Elmurod") #["Timur", "Bobur", "Elmurod"]

*# ikkinchi pozitsiyaga qo‘shish*

list1.**insert**(1, "Bilol") #["Timur", " Bilol", "Bobur", "Elmurod"]

*# elementlar to‘plamini qo‘shish ["Malik", "Samat"]*

list1.**extend**(["Malik", "Samat"])

# ["Timur", " Bilol", "Bobur", "Elmurod", "Malik", "Samat"]

*# element indeksini olish*

index\_of\_tom = list1.**index**("Bobur") # 2

*# ushbu indeksda o‘chirish*

removed\_item = list1.**pop**(index\_of\_tom)

# ["Timur", " Bilol", "Elmurod", "Malik", "Samat"]

*# oxirgi elementni olib tashlang*

oxirgi\_element = list1.**pop**()  # "Samat"

print(list1) # ["Timur", " Bilol", "Elmurod", "Malik"]

*# "Elmurod" elementini olib tashlash*

list1.**remove**("Elmurod")

print(list1)         # ["Timur", " Bilol", "Malik"]

# barcha elementlarni o‘chirish

list1.**clear**()

print (list1)        # []

**IN kalit so‘zining ishlatirlishi**

Agar biron bir element topilmasa, olib tashlash va indekslash usullari xatolik qaytaradi. Bunday vaziyatni oldini olish uchun elementni ishlatishdan oldin **in** kalit so‘zidan foydalanib uning mavjudligini tekshirish lozim:

list\_odamlar = ["Timur", "Bobur", "Samat"]

**if** "Ali" **in** list\_odamlar:

    list\_odamlar.**remove**("Ali")

print(list\_odamlar) # ["Timur", "Bobur", "Samat"]

Odamlar ro‘yxatida "Ali" elementi bo‘lsa, **if "Ali" in list\_odamlar** ifodasi "True" qiymati qaytaradi. Shunday ekan bu vaziyatda False qiymati qaytadi va list\_odamlar.remove(“Ali”) funksiyasi bajarilmaydi va list\_odamlar nomli ro‘yxat tarkibi o‘zgarishsiz qoladi.

**DEL metodi**

Pythonda ro‘yxatdan elementni o‘chirish uchun yana bir **del** nomli operatordan ham foydalaniladi. O‘chiriladigan element yoki elementlar to‘plami ushbu operatorga parametr sifatida uzatiladi:

list\_odamlar = ["Timur", " Bilol", "Bobur", "Elmurod", "Malik", "Samat"]

*# ikkinchi elementni olib tashlash*

**del** list\_odamlar [1]

print (list\_odamlar) # ["Timur", "Bobur", "Elmurod", "Malik", "Samat"]

*#   to‘rtinchi elementgacha o‘chirish*

list\_odamlar **[:3]**

print(list\_odamlar) #["Elmurod", "Malik", "Samat"]

**del** list\_odamlar**[1:]**#ikkinchi elementdan oxirigacha

print(list\_odamlar) #["Timur"]

**COUNT() metodi**

Agar element roʻyxatda necha marta borligini bilish kerak boʻlsa, count() metodidan foydalanish maqsadga muvofiq bo‘ladi:

odamlar = ["Bobur", " Bilol", "Bobur", "Elmurod", "Malik", "Bobur"]

odamlar\_soni = odamlar.**count**("Bobur")

print(odamlar\_soni)       # 3

**SORT() metodi**

Sort() metodi standart holatda o‘sish tartibida saralash uchun ishlatiladi:

odamlar = ["Timur", "Bilol", "Bobur", "Elmurod", "Malik", "Samat"]

odamlar.**sort**()

print(odamlar)

#['Bilol','Bobur','Elmurod','Malik','Samat', 'Timur']

Agar ma’lumotlarni teskari tartibda saralash zarur bo‘lsa, sort() metodidan keyin reverse() usulini qo‘llash mumkin:

odamlar = ["Timur", "Bilol", "Bobur", "Elmurod", "Malik", "Samat"]

odamlar.**sort**()

odamlar.**reverse**()

print(odamlar)

# ['Timur', 'Samat', 'Malik', 'Elmurod', 'Bobur', 'Bilol']

Saralash aslida ikkita ob’ektni taqqoslaydi va qaysi biri "kichik" bo‘lsa, "katta" ob’ektdan oldin joylashtiriladi. "katta" va "kichik" tushunchalari albatta nisbiydir. Va agar ro‘yxatning barcha elementlari raqamlardan tarkib topgan bo‘lsa, unda barchasi oddiy - raqamlar o‘sish tartibida joylashtiriladi. Agar ro‘yxatda satrlar va boshqa ob’ektlar mavjud bo‘lsa va saralash kerak bo‘lsa, unda vaziyat yanada murakkabroq bo‘ladi. Bunday holda harflar sonllardan oldin joylashtiriladi. Albatta sonli satrlar haqida gap ketmoqda. Chunki butun tipli ma’lumot bilan string tipli ma’lumot solishtirish xatolik keltirib chiqaradi. Demak, string tiplarni solishtirishda, agar ularning birinchi belgilar teng bo‘lsa, ikkinchi belgilar solishtiriladi va hokazo. Bunda raqamli belgi alifbodagi bosh harfdan "kichik" deb hisoblanadi va o‘z navbatida bosh harf ham kichik harfdan kichik hisoblanadi.

Shunday qilib, agar ro‘yxat tarkibida katta va kichik harflar mavjud bo‘lsa va saralash amali bajarilsa, unda unchalik to‘g‘ri bo‘lmagan natijalarni olinishi mumkin, chunki asosan "bob" matni "Tom" matnidan oldin kelishi kerak. Bunday holda sort metodining standart tartiblashini o‘zgartirish uchun unga parametr sifatida funktsiyani beriladi:

odamlar = ["Timur", "bilol", "Bobur", "elmurod", "Malik", "Samat"]

odamlar.**sort**()   *# standart tartiblash*

print(odamlar) #['Bobur', 'Malik', 'Samat', 'Timur', 'bilol', 'elmurod']

odamlar.**sort(key=str.lower)**

*# kichik katta-kichik tartiblash*

print ( odamlar )

# ['bilol', 'Bobur', 'elmurod', 'Malik', 'Samat', 'Timur']

**SORTED metodi**

Saralash usuliga qo‘shimcha ravishda ikkita shaklga ega bo‘lgan tartiblangan standart funksiyadan foydalanish mumkin:

***sorted(list):*** ***list*** ro‘yxatini tartiblaydi

***sorted(list, key):*** ***list*** ro‘yxati elementlariga ***key*** funksiyasini qoʻllash orqali roʻyxatni tartiblaydi

list\_per = ["Tom", "bob", "alis", "Sem", "Bill"]

sorted\_per = ***sorted***(list\_per, key=str.lower)

print (sorted\_per)

# ["alisa", "Bill", "bob", "Sem", "Tom"]

Ushbu funktsiyadan foydalanganda shuni yodda tutish kerakki, bu funksiya tartiblangan ro‘yxatni o‘zgartirmaydi, ammo u barcha tartiblangan elementlarni yangi ro‘yxatga joylashtiradi va natijani qaytaradi.

**MIN va MAX funksiyalari**

Pythonning **min()** va **max()** funksiyalari mos ravishda minimum va maksimum qiymatlarni topishga imkon beradi:

raqamlar = [9, 221, 12, 10, 3, 125, 18]

print (**min**(raqamlar))      # 3

print (**max**(raqamlar))      # 221

**COPY() funksiyasi**

Ro‘yxatlarni nusxalashda, ro‘yxatlar o‘zgaruvchan tur ekanligini yodda tutish kerak. Shuning uchun ikkala o‘zgaruvchi ham bir xil ro‘yxatga ishora qilsa, bitta o‘zgaruvchini o‘zgartirish boshqa o‘zgaruvchiga ta’sir qiladi:

list\_per = ["Timur", "Bobur", "Alisey"]

list\_per2 = list\_per

list\_per2.**append**("Samat") # elementni ikkinchi ro‘yxatga qo‘shish

# kishi1 va odamlar2 bir xil roʻyxatga ishora qiladi

print(list\_per)    # ["Timur", "Bobur", "Alisey", " Samat"]

print(list\_per2)    # ["Timur", "Bobur", "Alisey", " Samat"]

Bu yuqorida keltirilgan dasturdagi bir o‘zgaruvchining qiymatini ikkinchisiga yuklab qo‘yish holati nusxalash deb atalmaydi. Nushalash maqsadida bunday hatti harakatni qilmagan ma’qul. Elementlarni nusxalash uchun chuqurroq nusxalash talab qilinadi. Shundagina bir vaqtning nusxalangan va aslnusxadagi ro‘yxatlar qiymatlari bir xil bo‘lsada, aslida turli xil ro‘yxatlarni ifodalaydi. Buning uchun oddiygina **copy()** metodidan foydalanib nusxalash kerak bo‘ladi:

list\_per = ["Tom", "Bob", "Alis"]

list\_per2 = list\_per.**copy**() # elementlarni list\_per dan list\_per2 ga nusxalash

# elementni FAQAT ikkinchi ro‘yxatga qo‘shish

list\_per2.**append**("Sem")

# list\_per va list\_per2 turli roʻyxatlarni bildiradi

print(list\_per)    # ["Tom", "Bob", "Alis"]

print(list\_per2)    # ["Tom", "Bob", "Alis", "Sem"]

**Ro‘yxatning bir qismini nusxalash**

Agar butun ro‘yxatni emas, balki uning ma’lum bir qismini nusxalash kerak bo‘lsa, unda quyidagi shakllarni qabul qilishi mumkin bo‘lgan maxsus sintaksisdan foydalanish kerak bo‘ladi:

***list[:end]:*** roʻyxat boshidan ***end*** elementigacha bo‘lgan oraliqni tanlash.

***list[start:end]:*** ro‘yxatning ***start*** va ***end*** elementlari oralig‘ini belgilaydi

***list[start:end:step]:*** ro‘yxatning ***start*** va ***end*** elementlari oralig‘ini ***step*** qadam bilan belgilaydi. Odatiy bo‘lib, bu ***step*** parametr 1 ga teng bo‘ladi.

list\_per = ["Tomson", "Bobi", "Elison", "Semmi", "Timati", "Billi"]

slice\_people1 = list\_per **[:3]** # 0 dan 3 gacha

print(slice\_people1)    # ['Tomson', 'Bobi', 'Elison']

slice\_people2 = list\_per **[1:3]** #1 dan 3 gacha

print(slice\_people2)    # ["Bobi", "Elison"]

slice\_people3 = list\_per**[1:6:2]** #1 dan 6 gacha 2 ga oshib

print(slice\_people3)    #["Bobi", "Semmi", "Billi"]

**Manfiy indeks**

Ro‘yxatning oxirgi elementidan boshlab murojaatni boshlash kerak bo‘lganda manfiy indeksdan foydalanish kerak bo‘ladi. Masalan, -1 – element oxirgi element, -2 – element oxiridan bitta oldingi va hokazo.

list\_per  = ["Tomson", "Bobi", "Elison", "Semmi", "Timati", "Billi"]

*# nolinchidan oxirgisigacha, oxirgi xisobga kirmaydi*

slice\_people1 = list\_per**[:-1]**

print(slice\_people1)

*# ['Tomson', 'Bobi', 'Elison', 'Semmi', 'Timati']*

*# oxirgi uchinchi elementdan oxirgisigacha, oxirgisini olmaydi*

slice\_people2 = list\_per **[-3:-1]**

print(slice\_people2)    # ["Semmi", "Timati"]

**Ro‘yxatlarni ulash**

Qo‘shish (+) operatori ro‘yxatlarni birlashtirish uchun ishlatiladi:

people1 = ["Tomson", "Bobi", "Alison"]

people2 = ["Tomson", "Semmi", "Timati", "Billi"]

people3 = people1 + people2

print (people3)

*# ["Tomson", "Bobi", "Elison", "Tomson", "Semmi", "Timati", "Billi"]*

**Ichma-ich Ro‘yxatlar**

Ro‘yxatlar satrlar, raqamlar kabi standart ma’lumotlardan tashqari, boshqa ro‘yxatlarni ham o‘z ichiga olishi mumkin. Bunday ro‘yxatlarni jadvallar bilan bog‘lash mumkin, bu yerda ichki ro‘yxatlar qatorlar vazifasini bajaradi. Masalan:

odamlar = [

    ["Tomson", 29],

    ["Elison", 33],

    ["Bobi", 27]

]

print(odamlar[0])          *# ["Tomson", 29]*

print(odamlar[0][0])       *# "Tomson"*

print(odamlar[0][1])       *# 29*

Ichki ro‘yxatning elementiga murojaat qilish uchun bir juft indeksdan foydalanish kerak: odamlar[0][1] - birinchi joylashtirilgan ro‘yxatning ikkinchi elementiga murojaat.

Umumiy ro‘yxatni, shuningdek, ichki ro‘yxatlarni qo‘shish, o‘chirish va o‘zgartirish oddiy (bir o‘lchovli) ro‘yxatlar kabi amalga oshiriladi:

odamlar = [

    ["Tomson", 29],

    ["Elison", 33],

    ["Bobi", 27]

]

*# ichki ro‘yxat yaratish*

odam = **list**()

odam.**append**("Billi")

odam.**append**(41)

*# ichki ro‘yxat qo‘shish*

odamlar.**append**(odam)

print(odamlar[-1])   # ["Billi", 41]

*# ichki roʻyxatga qoʻshish*

odamlar[-1].**append**("+79996543210")

print(odamlar[-1])  # ["Billi", 41, "+79996543210"]

*# oxirgi elementni o‘rnatilgan ro‘yxatdan olib tashlash*

odamlar**[-1].pop**()

print(odamlar[-1])          # ["Billi", 41]

*# oxirgi kiritilgan roʻyxatni butunlay oʻchirish*

odamlar.**pop(-1)**

*# birinchi elementni o‘zgartirish*

odamlar[0] = ["Semmi", 18]

print (odamlar)

*# [ ["Semmi", 18], ["Alison", 33], ["Bobi", 27]]*

Ichki ro‘yxatlar ustida takrorlash:

odamlar = [

    ["Tomson", 29],

    ["Elison", 33],

    ["Bobi", 27]

]

**for** odam **in** odamlar:

**for** item **in** odam:

print(item, end=" | ")

Konsol chiqishi :

Tomson | 29 | Elison | 33 | Bobi | 27 |

**Nazorat savollari:**

1. Ro‘yxat deb nimaga aytiladi?
2. Ro‘yxatlarni qanday yaratiladi?
3. Ro‘yxatlarning elementlari bir turga mansub bo‘lishi shartmi?
4. ro‘yxat elementlariga murojaatlar qanday amalga oshiriladi?
5. Ro‘yxat uzunligini qanday aniqlanadi?
6. Ro‘yxat elementini biro songa ko‘paytirish nimani beradi?
7. Manfiy index tushunchasi nimani anglatadi?
8. tomson, bobi, semmi = odamlar – ushbu ifodaning ma’nosi nima?
9. Ro‘yxatga ma’lumotni sikl orqali kiritish qanday amalga oshiriladi?
10. Ro‘yxat elementlarini saralash uchun qaysi metodni ishlatiladi?