List va tuple

1. Python’da ro‘yxatlar (list) va o‘zgarmas ro‘yxatlar (tuple) bilan ishlashni o‘rganish

1.1 Kirish

Ma’lumot tuzilmalari nima?

List va Tuple nima uchun kerak?

1.2 list — bu bir nechta qiymatlarni bitta o‘zgaruvchida saqlash imkonini beruvchi ma’lumot tuzilmasi.

1.3 Tasavvur qiling, sizda bir nechta mevalar bor: olma, banan, anor. Har birini alohida saqlash o‘rniga, ularni bir sumkaga solib qo‘yish mumkin. Shu sumka — bu *list*.

mevalar = ["olma", "banan", "anor"]

2. List qanday yaratiladi?

2.1 Listlar 3 turga bo’linadi

2.2 bo’sh list, sonlardan iborat va har xil turdagi qiymatlar

bosh\_list = []                    # Bo‘sh ro‘yxat

raqamlar = [1, 2, 3, 4, 5]         # Sonlardan iborat

ism\_va\_yosh = ["Ali", 20]         # Har xil turdagi qiymatlar

3.Elementlarga murojaat qilish

3.1 pythonda listdagi elementlarga murojaat qilish uchun uning indexi bilan chaqiriladi

3.2 Index => bu listni ichidage elementning tartib soni

3.3 Lekin dasturlash taribni doim 0 dan boshlaydi shuning uchun 1-element 0-indexda deyiladi

3.4 Agar oxirgi elementni chaqirmoqchi bo’lsak -1 deb yozamiz

print(mevalar[0])   # "olma"

print(mevalar[1])   # "banan"

print(mevalar[-1])  # oxirgi element: "anor"

4. Elementni o‘zgartirish

4.1 Listlarni elementlarini o’zgartirish uchun uning indexi orqali chaqirib turib

4.2 boshqa o’zgaruvchiga tenglanadi

mevalar[1] = "shaftoli"

print(mevalar)  # ["olma", "shaftoli", "anor"]

6.bularni ishlatishdan oldin listni for orqali aylanishni ham korib chiqamiz

6.1 for orqali ishlaganimizda in methodi orqali ishlaymiz

mevalar = ["olma", "banan", "anor"]

for meva in mevalar:

    print(meva)

7.Endi list da Nested List degan narsa mavjud => bu listni ichidagi list bo’ladi

7.1 Uni ichidagi ma’lumotni ham chaqirish oson

talabalar = [["Ali", 20], ["Laylo", 22], ["Aziz", 19]]

7.2 Bu holatda birinchi o’rinda katta listning indexi chaqiriladi

7.3 keyin kichkina listning ma’lumot indexi chaqiriladi

7.4 Ya’ni bizga bu yerda Aziz kerak bo’lsa birinchi uning list indexini chaqiramiz ya’ni => 2

7.5 Keyin shuni ichidagi 0 – indexni olishimiz kerak

talabalar = [["Ali", 20], ["Laylo", 22], ["Aziz", 19]]

print(talabalar[2][0])

8.in va in not orqali tekshirish

8.1 in bu ichidagi element bor yoki yo’qligini bildiradi

8.2 in not esa uni ichida shu element yo’qligini bildiradi

mevalar = ["olma", "banan", "anor"]

if "olma" in mevalar:

    print("Olma bor!")

if "shaftoli" not in mevalar:

    print("Shaftoli yo‘q.")

9.Listdan nusxa olish

9.1 Listdan nusxa olish method orqali bo’ladi => .copy()

9.2 bu method orqali listni ikkinchi o’zgaruvchiga nusxalashimiz mumkin

9.3 Bu holatda 2-listga ma’lumot qo’shsak yoki ma’lumot olib tashlasak 1-ga ta’sir o’tkazmaydi

9.4 Lekin agar bir 1-listni 2-o’zgaruvchiga tenglab qo’ysak 2-listda qanday o’zgarish bo’lsa 1-ga ham shunday ta’sir qiladi

9.5 Bu holat katta xatoliklarga olib kelishi mumkin

a = [1, 2, 3]

b = a.copy()

b.append(4)

print(a)  # [1, 2, 3]

print(b)  # [1, 2, 3, 4]

copy methodisiz shunchaki tenglash

a = [1, 2, 3]

b = a

b.append(4)

print(a)  # [1, 2, 3, 4]

print(b)  # [1, 2, 3, 4]

10.Listni ichidan eng katta sonni toppish

10. bu uchun max methodi kerak bo’ladi

sonlar = [3, 5, 2, 8, 1]

print(max(sonlar))  # 8

11.Eng kichigini toppish

11.1 bu uchun min methodi kerak

sonlar = [3, 5, 2, 8, 1]

print(min(sonlar))  # 1

12.number listdagi ma’lumotlarni umumiy yig’ib chiqarish

12. Bu uchun sum methodi kerak bo’ladi

sonlar = [3, 5, 2, 8, 1]

print(sum(sonlar))  #  19

13.pythonda listlarni qo’shish ko’paytirish holatlari mavjud

13.1 2 ta listni bir biriga qo’shadigan bo’lsak ular umumiy bitta listga ma’lumotlari to’planib qoladi

a = [1, 2, 3]

b = [4, 5]

c = a + b

print(c)  # [1, 2, 3, 4, 5]

14.Listni ko’paytirish

14.1 agar listni qandaydur songa ko’paytiradigan bo’lsak uni barcha ma’lumotlarini shu songa qadar ko’paytirib beradi

a = [0, 1]

b = a \* 3

print(b)  # [0, 1, 0, 1, 0, 1]

15.Slicing(Tilimlash)

15.1 Slicing bu listni ma’lum bir joyidan 2-joyga qadar olish

sonlar = [10, 20, 30, 40, 50]

print(sonlar[1:4])  # [20, 30, 40] (1-pozitsiyadan boshlab 4-gacha, 4 kirmaydi)

print(sonlar[:3])   # [10, 20, 30]

print(sonlar[::2])  # [10, 30, 50] (har 2-ta qadam bilan)

16.Listlarni taqqoslash

16.1 lislarni taqqoslashda python ularni ichidagi ma’lumotlarni jamlab keyin true yoki false qaytaradi

16.2 Bu amal faqat number holatidagi listlar uchun ishlaydi

a = [1, 2, 3]

b = [1, 2, 3]

c = [1, 2, 4]

print(a == b)  # True

print(a != c)  # True

print(a < c)   # True  (3 < 4)

17. Listlarni ajratish (split()) va birlashtirish (join())

17.1 String ma’lumotlarni split methodi orqali bitta list qilib qo’yish mumkin

17.2 Bu uchun gapning qaysi joyidan qirqib olish kerakligini buyruq qilib beriladi

Masalan: !

a = "Salom! Qalesan! Nimagap"

b = a.split("!")

print(b) # ['Salom', ' Qalesan', ' Nimagap']

18.join shuning teskarisi listning ichidagi ma’lumotlarni orasiga biror narsa qo’shib bitta string qilad

a = ["nima", "gap"]

b = "-".join(a)

print(b) # nima-gap

19.List methodlari

19.1 listlardan ma’lumot olish, ularni ochirish, tartiblash kabi holatlar uchun holatlar ishlatiladi

19.2 bular methodlar deyiladi

| **Metod** | **Tushuntirishi** | **Misol** |
| --- | --- | --- |
| append(x) | Oxiriga element qo‘shadi | a.append(5) |
| insert(i, x) | Indeksga element qo‘shadi | a.insert(1, 10) |
| remove(x) | Elementni o‘chiradi (birinchi) | a.remove(3) |
| pop([i]) | Elementni olib tashlaydi | a.pop() |
| clear() | Barcha elementlarni o‘chiradi | a.clear() |
| index(x) | Element indeksini qaytaradi | a.index(2) |
| count(x) | Element necha marta chiqqanini topadi | a.count(2) |
| sort() | Tartiblaydi | a.sort() |
| reverse() | Teskari qiladi | a.reverse() |
| copy() | Nusxa qaytaradi | b = a.copy() |
| extend(list2) | Ikkinchi listni birlashtiradi | a.extend(b) |

20.append methodi listni ichiga ma’lumotlar qo’shish uchun ishlatiladi

20.1 bu method listni oxiriga qo’shib beradi

mevalar = ["olma", "banan"]

mevalar.append("anor")

print(mevalar)  # ["olma", "banan", "anor"]

21.insert methodi => Listning istalgan index-iga yangi element qo‘shadi.

21.1 u 2 ta parameter bilan ishlaydi

21.2 1-qo’shilishi kerak bo’lgan index

21.3 2-qo’shilishi kerak bo’lgan element

raqamlar = [1, 3, 4]

raqamlar.insert(1, 2)  # 1-pozitsiyaga 2 ni qo‘yadi

print(raqamlar)  # [1, 2, 3, 4]

22.remove methodi => Listdan **birinchi uchragan** x elementini o‘chiradi.

22.1 Nima uchun 1-uchragan, chunki 2ta bir xil ma’lumot bo’lsa birinchi yozilganini o’chiradi ikkalasini ham emas

ismlar = ["Ali", "Laylo", "Ali"]

ismlar.remove("Ali")

print(ismlar)  # ["Laylo", "Ali"]

23.pop(index) (yoki pop()) => pop methodi listning indexiga qarab o’sha elementni tenglangan o’zgaruvchiga beradi

23.1 agar unga index bermasak oxirgi elementni olib beradi

a = [10, 20, 30]

oxirgi = a.pop()

print(oxirgi)  # 30

print(a)       # [10, 20]

birinchi = a.pop(0)

print(birinchi)  # 10

24.clear()

24.1 bu method listni ma’lumotlarini tozalab yuboradi

a = [1, 2, 3]

a.clear()

print(a)  # []

25.index(x) => bu method bizga kerakli element nechanchi indexda turganini topib beradi

a = ["a", "b", "c", "b"]

print(a.index("b"))  # 1

26.count(x) => Listda x nechta marta qatnashganini sanaydi.

a = [1, 2, 2, 3, 2]

print(a.count(2))  # 3

27.sort() => Listni o‘sish tartibida tartiblaydi (default: katta-kichik).

27.1 uning teskari tartibi ham mavjud shunchaki reverse holatini true qilinadi

a = [3, 1, 2]

a.sort()

print(a)  # [1, 2, 3]

teskari tartib

a.sort(reverse=True)

print(a)  # [3, 2, 1]

28.reverse() => List tartibini teskariga o‘zgartiradi.

a = [1, 2, 3]

a.reverse()

print(a)  # [3, 2, 1]

29.extend(list2) => Boshqa listni birlab qo‘shadi, .append() dan farqli ravishda bir necha elementni qo‘shadi.

a = [1, 2]

b = [3, 4]

a.extend(b)

print(a)  # [1, 2, 3, 4]

30.1 nega append qila qolmadik chunki agar append qilsak 2-listni o’zi ham 1-listga kirib ketadi

a = [1, 2]

b = [3, 4]

a.append(b)

print(a)  # [1, 2, [3, 4]]

31.tuple — bu Python’da bir nechta qiymatni tartiblangan holda saqlash uchun ishlatiladigan o‘zgarmas ma’lumot tuzilmasi.

31.1 Ya’ni, listga o‘xshaydi, lekin **o‘zgartirib bo‘lmaydi**.

31.2 Agar list – oddiy daftar bo‘lsa, unga yozish, o‘chirish mumkin.

tuple esa – bosmadan chiqqan kitob: faqat o‘qish mumkin, o‘zgartirib bo‘lmaydi.

31.3 tupleni yaratish uchun [] qavs orniga () qoyiladi

t1 = (1, 2, 3)

t2 = ("olma", "banan", "anor")

t3 = ()                      # bo‘sh tuple

t4 = (5,)                    # bitta elementli tuple — vergul majburiy!

32.tuplega ham indexlar orqali murojaat qilinadi

mevalar = ("olma", "banan", "anor")

print(mevalar[0])    # "olma"

print(mevalar[-1])   # "anor"

33.tupleda ham for orqali aylanish mumkin

t = ("olma", "banan", "anor")

for meva in t:

    print(meva)

34.Nega o’zgarmas

34.1 tupleda yozilgan ma’lumotlarni o’zgartirib bo’lmaydi

34.2 aynan tupleni ichida yozilgan ma’lumotlarni

t = (1, 2, 3)

t[0] = 100  # ❌ Xatolik beradi: TypeError

35.tuplening ham o’zining 2 ta methodi bor

35.1 .count(), index() => count bir element necha marta qaytarilganini korsatadi

35.2 index => elementning nomini kiritsak indexini topadi

t = (1, 2, 3, 2)

print(t.count(2))

t = (1, "olma", 3, 2)

print(t.index("olma"))

36.tuple nega o’zgarmaydi?

36.1 tuple yaratilib bo‘lgach, kompyuter uni xotirada bitta qat'iy joyga joylashtiradi. U elementlar soni va joylashuvi bilan birga aniq struktura bo‘ladi.

Tuple va List farqi

| **cons** | **list** | **tuple** |
| --- | --- | --- |
| O‘zgaruvchanmi? | ✅ Ha | ❌ Yo‘q |
| Tezlik | Sekinroq | Tezroq (ko‘proq joyda ishlatiladi) |
| Qavs turi | [] | () |
| Metodlar | Ko‘p | Juda kam (count, index) |

t = (1, 2, [3, 4])

t[2][0] = 100

print(t)       # (1, 2, [100, 4])

37. nega o’zgaryapti?

37.1 Bu yerda tupleni emas balki listni o’zgaryotganini ko’rishimiz mumkin

37.2 Bu yerda tuple ning ichidagi listni chaqirib olinmoqda va shu listni ichidagi ma’lumotni o’zgartirilmoqda

38.tupleni ustida amal qilish

38.1 bu ham huddi list kabi ishlaydi

a = (1, 2)

b = (3, 4)

c = a + b      # (1, 2, 3, 4)

d = a \* 3      # (1, 2, 1, 2, 1, 2)

39.tupleni listga o’tkazish ham mumkin

39.1 yoki aksincha listni tuplega

t = (1, 2, 3)

l = list(t)

l[0] = 100

print(l)       # [100, 2, 3]

list => tuple

l = [1, 2, 3]

t = tuple(l)

print(t)       # (1, 2, 3)

