# Python programming language

Week 5. Python lists, tuple

**Alisher Ismailov**
Lecturer

Department of Information technology
Andijan Branch of Tashkent Financial Institute
Email: alisherismailov534@gmail.com

# Python dasturlash tili

5-Mavzu. Python lists, tuple

**Ma’ruzachi: Alisher Ismoilov**

Axborot texnologiyalari kafedrasi
Toshkent moliya instituti Andijon filiali

Elektron pochta: alisherismailov534@gmail.com

# 5-Mavzu. Python lists, tuple

# Reja:

1. List (Ro‘yxat)
2. (Ro‘yxat) List to‘plami elementlari
3. List to‘plami tartiblangan bo‘ladi
4. List to‘plami elementlarini o‘zgartirish mumkin
5. List to‘plamida elementlarni bir xil bo‘lishiga ruxsat beriladi
6. List to‘plami uzunligini aniqlash
7. List to‘plami elementlari berilganlar turlari
8. List to‘plami elementlariga kirish
9. List to‘plami manfiy indeks holatidan foydalanish
10. List to‘plami indekslar oralig‘idan foydalanish
11. List to‘plami elementlarini o‘zgartirish
12. List to‘plami element qiymatlari oralig‘ini o‘zgartirish

1. List to‘plamiga yangi elementlarni kiritish
2. append() funksiyasi
3. extent() funksiyasi (ikkita listni elementini bir-biriga o‘tkazish)
4. List to‘plami elementlarini o‘chirish
5. pop() funksiyasi
6. del kalit so‘zi
7. clear() funksiyasi
8. List to‘plami elementlarini sikl orqali chop etish
9. List to‘plami elementlarini tartiblash
10. List to‘plami elementlarini teskari tartiblash. (o‘ngdan chapga)
11. List to‘plamidan nusxa ko‘chirish
12. Ikkita list to‘plamini birlashtirish

# 5-Mavzu. Python lists, tuple

# Reja:

1. Tuple to‘plami elementlari
2. Tuple to‘plami elementlariga kirish
3. Tuple to‘plamida manfiy indeksiya
4. Tuple to‘plami indekslar oralig‘idan foydalanish
5. Tuple to‘plami manfiy indekslar oralig‘idan foydalanish
6. Tuple to‘plami qiymatlarini o‘zgartirish
7. Tuple to‘plamiga elementlarni qo‘shish
8. Tuple to‘plamini elementlarini sikl orqali chop etish
9. Sikl orqali tuple to‘plami elementlarini indeks raqamlari bo‘ylab chop etish
10. Tuple to‘plami elementlarini chop etish uchun while siklidan foydalanish
11. Ikkita Tuple to‘plamini bir-biriga qo‘shish
12. Tuple to‘plami elementlarini ko‘paytirish

# Python dasturlash tilida

## List (Ro`yxatlar)

**List** bir o‘zgaruvchida bir nechta qiymatlarni saqlash uchun ishlatiladi. Ro‘yxatlar (List) Python dasturlash tilida ma’lumotlar to‘plamini saqlash uchun ishlatiladigan 4 ta berilganlar turlaridan biri hisoblanadi, qolgan 3 tasi Tuple, Set va Dictionary bo‘lib, ularning barchasi turli sifat va foydalanishga ega

# Python dasturlash tilida List (Ro`yxatlar)

- **Ro‘yxatlar (List) kvadrat qavslar [] yordamida tuziladi:**

```
In [24]: list_toplami = ["Lingivistika", "Dasturlash"]

         print(list_toplami)

         ['Lingivistika', 'Dasturlash']
```

# Python dasturlash tilida

## List (Ro`yxatlar)

List to‘plami elementlari tartiblangan holatda bo‘ladi va elementlarni o‘zgartirish mumkin va takroriy qiymatlarga ruxsat beriladi. **List** to‘plami elementlari indekslangan holatda bo‘ladi, birinchi element [0] indeksda, ikkinchi element [1] indeksga to‘g‘ri keladi. Soddaroq tushintirganda, yuqoridagi suratda 3 ta element mavjud, olma elementi [0] indeksda turibdi, behi elementi [1] indeksda va gilos elementi [2] indeksda joylashgan.

# Python dasturlash tilida

# List (Ro`yxatlar)

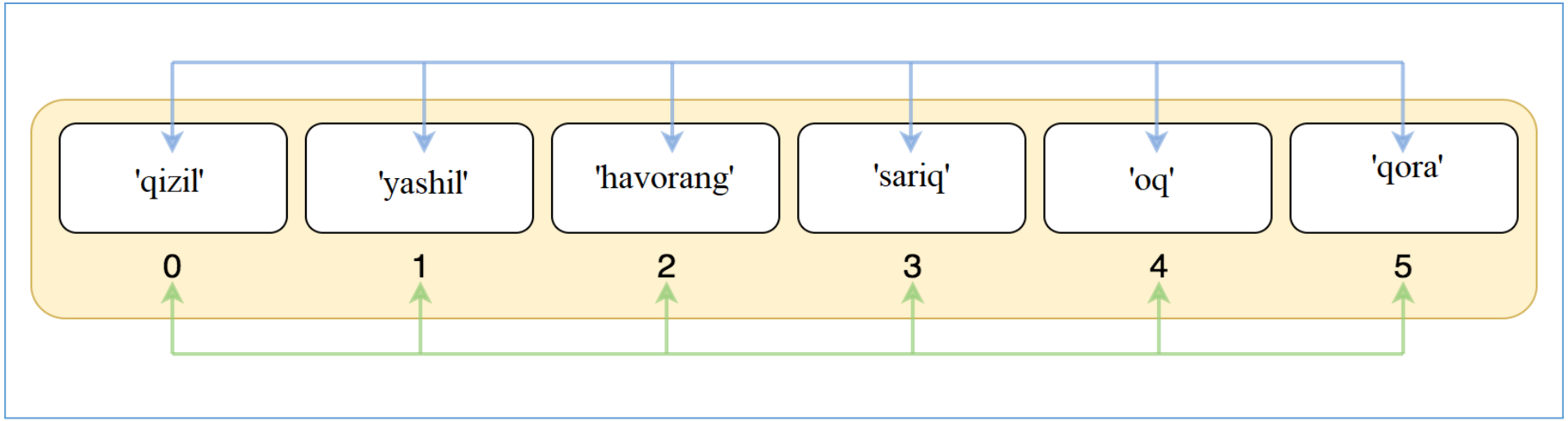
**Indekslash**

Python dasturlash tilida list(ro‘yxat) boshqa dasturlash tillaridagi (Ruby, JavaScript, PHP) massivlarga o‘xshaydi. Bu sizga sanab o‘tilgan elementlar to‘plamini bir joyda saqlash va indekslari bo‘yicha kirish imkonini beradi.

misol: ranglar = ['qizil', 'yashil', 'havorang', 'sariq', 'oq', 'qora’]

Bu yerda biz ranglar ro‘yxatini aniqladik. Ro‘yxatdagi har bir elementning qiymati (rang nomi) va indeksi (ro‘yxatdagi o‘rni) mavjud. Python nolga asoslangan indekslashdan foydalanadi. Bu shuni anglatadiki, birinchi element (qiymat "qizil") 0-indeksiga ega, ikkinchisi ("yashil" qiymat) indeks 1 va hokazo.

# Python dasturlash tilida List (Ro`yxatlar)



# Python dasturlash tilida List (Ro`yxatlar)

- List to‘plami elementlari indekslanganligi sababli, list elementlari bir xil qiymatga ega bo‘lishi mumkin:

```
In [6]: bu_list = ["olma", "behi", "gilos", "behi", "gilos"]
        print(bu_list)

        ['olma', 'behi', 'gilos', 'behi', 'gilos']
```

# Python dasturlash tilida List (Ro`yxatlar)

- List to‘plamida nechta element borligini aniqlash uchun **len()** funksiyasidan foydalaniladi:

```
In [25]: list_toplami = ["Lingivistika", "Dasturlash"]

         print(len(list_toplami))

         2
```

# Python dasturlash tilida List (Ro`yxatlar)

- List to‘plami elementlari berilganlar turlari har xil bo‘lishi mumkin:

```
In [27]: list_toplami = ["Lingivistika", "Dasturlash", 2022, True]

         print(list_toplami)

         ['Lingivistika', 'Dasturlash', 2022, True]
```

# Python dasturlash tilida List (Ro`yxatlar)

- List to‘plami elementlari indekslangan holatda bo‘ladi va siz ularga indeks raqamiga murojaat
- qilish orqali kirishingiz mumkin:

```
In [29]: list_toplami = ["Python", "C++", "Java", "Delphi"]

         print(list_toplami[0])

         Python
```

# Python dasturlash tilida List (Ro`yxatlar)

- Manfiy indekslash list to‘plami elementlarini oxiridan boshlanadi degan ma’noni anglatadi.
- -1 oxirgi elementga, -2 ikkinchi oxirgi elementga to‘g‘ri keladi va hokazo

```
In [12]: bu_list = ["olma", "behi", "gilos", 55, 10.7]
         print(bu_list[-1])

         10.7
```

# Python dasturlash tilida List (Ro`yxatlar)

- Siz list to‘plami elemetlarini chop etish uchun indeksni qayerdan boshlash va qayerdan tugatishni belgilash orqali bir qator indekslarni belgilashingiz mumkin

```
In [30]: list_toplami = ["Python", "C++", "Java", "Delphi"]

         print(list_toplami[0:2])

         ['Python', 'C++']
```

# Python dasturlash tilida List (Ro`yxatlar)

- List to‘plami elementining qiymatini o‘zgartirish uchun indeks raqamidan foydalanish mumkin:

```
In [31]: list_toplami = ["Python", "C++", "Java", "Delphi"]
         list_toplami[0] = "Javascript"

         print(list_toplami[0:2])
```

```
['Javascript', 'C++']
```

# Python dasturlash tilida List (Ro`yxatlar)

- List to‘plami elementlari qiymatini ularning indeks oralg‘ida o‘zgartirish uchun yangi qiymatlar bilan listni belgilang va yangi qiymatlarni kiritmoqchi bo‘lgan indeks raqamlari oralig‘iga joylashtiring:

```
In [18]: bu_list = ["olma", "behi", "gilos", 55, 10.7]
         bu_list[0:2] = ["Shaftoli", "O`rik"]
         print(bu_list[1])

         O`rik
```

# Python dasturlash tilida List (Ro`yxatlar)

- List to‘plamidagi mavjud qiymatlarni almashtirmasdan yangi ro‘yxat elementini kiritish uchun
- **insert()** funksiyasidan foydalanish mumkin.
- **insert()** funksiyasi elementni belgilangan indeksga joylashtiradi:

```
In [6]: sport = ["futbol", "boks", "kurash"]

        sport.insert(2, "volleybol")

        print(sport[2])

        volleybol
```

# Python dasturlash tilida List (Ro`yxatlar)

- List to‘plami oxiriga element qo‘shish uchun **append()** funksiyasidan ham foydalanish mukin:

```
In [7]: aktyorlar = ["Jeki Chan", "Vandam", "Bryus Li"]

        aktyorlar.append("Chak No`ris")

        print(aktyorlar)

        ['Jeki Chan', 'Vandam', 'Bryus Li', 'Chak No`ris']
```

# Python dasturlash tilida List (Ro`yxatlar)

- Bir list to‘plamidagi elementlarni boshqa list to‘plamiga qo‘shish uchun **extend()** funksiyasidan
- foydalanish mumkin.

```
tillar = ["O`zbek", "Ingliz", "Turk"]
mamlakat = ["O`zbekiston", "Angliya", "Turkiya"]

tillar.extend(mamlakat)

print(tillar)
```

```
['O`zbek', 'Ingliz', 'Turk', 'O`zbekiston', 'Angliya', 'Turkiya']
```

# Python dasturlash tilida List (Ro`yxatlar)

- **remove()** funksiyasi belgilangan elementni o‘chirib tashlaydi.

```
In [32]: list_toplami = ["Python", "C++", "Java", "Delphi"]

         list_toplami.remove("Java")

         print(list_toplami)

         ['Python', 'C++', 'Delphi']
```

# Python dasturlash tilida List (Ro`yxatlar)

- **pop()** funksiyasi belgilangan indeksdagi elementni o‘chirib tashlaydi.

```
In [33]: list_toplami = ["Python", "C++", "Java", "Delphi"]

         list_toplami.pop(1)

         print(list_toplami)

         ['Python', 'Java', 'Delphi']
```

# Python dasturlash tilida List (Ro`yxatlar)

- Agar siz element indeksini ko‘rsatmasangiz, **pop()** funksiyasi oxirgi elementni o‘chirib tashlaydi.

```
In [7]: meva = ["olma", "behi", "gilos"]

        meva.pop()
        print(meva)

        ['olma', 'behi']
```

# Python dasturlash tilida List (Ro`yxatlar)

- del kalit so‘zi list to‘plamida belgilangan indeksdagi elementni o‘chirib tashlaydi:

```
In [12]: tillar = ["O`zbek", "Ingliz", "Turk"]

         del tillar[1]

         print(tillar)

         ['O`zbek', 'Turk']
```

# Python dasturlash tilida List (Ro`yxatlar)

- **del** kalit so‘zi list to‘plamini butunlay o‘chirib tashlashi ham mumkin, agar indeks raqami
- belgilanmagan bo‘lsa.

```
list_toplami = ["Python", "C++", "Java", "Delphi"]

del list_toplami

print(list_toplami)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most
~\AppData\Local\Temp/ipykernel_6668/1672541435.py in <mod
      3 del list_toplami
      4
----> 5 print(list_toplami)

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

# Python dasturlash tilida List (Ro`yxatlar)

- **clear()** funksiyasi list to‘plami elementlarini o‘chirib listni bo‘shatadi. List to‘plami bo‘sh to‘plam
- sifatida saqlanib qoladi.

```
In [35]: list_toplami = ["Python", "C++", "Java", "Delphi"]

         list_toplami.clear()

         print(list_toplami)

         []
```

# Python dasturlash tilida List (Ro`yxatlar)

- Siz list to‘plami elementlarini **for** siklidan foydalanib ekranga chop etishingiz mumkin:

```
In [13]: meva = ["olma", "behi", "gilos"]

         for x in meva:
             print(x)

         olma
         behi
         gilos
```

# Python dasturlash tilida List (Ro`yxatlar)

- List to‘plami elementlarini chop etishni yana bir yo‘li listni indeks raqamiga murojaat qilib, list
- elementlarini chop etish mumkin. Tegishli iteratsiyani yaratish uchun **range()** va **len()**
- funksiyalaridan foydalanish mumkin

```
In [14]: tillar = ["O`zbek", "Ingliz", "Turk"]

         for x in range(len(tillar)):
             print(tillar[x])
```

```
O`zbek
Ingliz
Turk
```

# Python dasturlash tilida List (Ro`yxatlar)

- List to‘plami elementlarini alfanumerik tartibda tartiblash uchun **sort()** funksiyasidan
- foydalaniladi. Quyidagi misollarda ko‘rishingiz mumkin:

```
In [36]: list_toplami = ["Python", "C++", "Java", "Delphi"]

         list_toplami.sort()

         print(list_toplami)

         ['C++', 'Delphi', 'Java', 'Python']
```

```
In [17]: son = [100, 30, 60, 10, 5]

         son.sort()
         print(son)

         [5, 10, 30, 60, 100]
```

# Python dasturlash tilida List (Ro`yxatlar)

- List to‘plami elementlarini tartiblash uchun teskari **reverse= True** kalit so‘zidan foydalaniladi:

```
In [16]: buyukshaxs = ["Amir Temur", "Faboriy", "Buxoriy"]

         buyukshaxs.sort(reverse = True)

         print(buyukshaxs)

         ['Faboriy', 'Buxoriy', 'Amir Temur']
```

# Python dasturlash tilida List (Ro`yxatlar)

- **reverse()** funksiyasidan foydalanib ham elementlarning teskari tartiblash mumkin.

```
In [37]: list_toplami = ["Python", "C++", "Java", "Delphi"]

         list_toplami.reverse()

         print(list_toplami)

         ['Delphi', 'Java', 'C++', 'Python']
```

# Python dasturlash tilida List (Ro`yxatlar)

- List to‘plamidan nusxa olish uchun python dasturlash tilida funksiyalar mavjud, ulardan biri **copy()** funksiyasi hisoblanadi. Quyidagi misolda meva1 to‘plamidan meva2 to‘plamiga nusxa ko‘chirilganini ko‘rish mumkin:

```
In [17]: imperiya = ["Mo`gul", "Temuriy", "Boburiy", "O`smonli"]

         nusxaolish = imperiya.copy()

         print(nusxaolish)

         ['Mo`gul', 'Temuriy', 'Boburiy', 'O`smonli']
```

# Python dasturlash tilida List (Ro`yxatlar)

- List to‘plamida nusxa olishning yana bir usuli bu **list()** funksiyasidan foydalanishdir.

```
In [18]: mahalla_nomlari = ["Mustaqillik", "Mehnat", "Nurli Diyor"]

         nusxaolish = list(mahalla_nomlari)

         print(nusxaolish)

['Mustaqillik', 'Mehnat', 'Nurli Diyor']
```

# Python dasturlash tilida List (Ro`yxatlar)

- Python dasturlash tilida ikki yoki undan ortiq listlarni birlashtirishni bir necha usuli mavjud.
- Eng oson usullaridan biri + operatoridan foydalanishdir. Misolda ko‘rishingiz mumkin:

```
list_toplami = ["Python", "C++", "Java", "Delphi"]
fanlar = ["Dasturlash", "Axborot texnologiyalari"]

birlashtirish = list_toplami + fanlar

print(birlashtirish)
```

```
['Python', 'C++', 'Java', 'Delphi', 'Dasturlash', 'Axborot texnologiyalari']
```

# Python dasturlash tilida List (Ro`yxatlar)

- Ikkita listni bir-biriga qo‘shishning yana bir usuli bu meva listidagi barcha elementlarni sabzavot listiga birma-bir qo‘shishdir:

```
In [2]: meva = ["olma", "behi", "gilos"]
        sabzavot  = ["sabzi", "sholg`om", "ukrop"]

        for x in meva:
            sabzavot.append(x)
        print(sabzavot)

        ['sabzi', 'sholg`om', 'ukrop', 'olma', 'behi', 'gilos']
```

# Python dasturlash tilida List (Ro`yxatlar)

Yana bir usuli bu bir listdagi elementlarni boshqa listga qo‘shish uchun **extend()** funksyasidan foydalanishingiz mumkin:

```
In [39]: list_toplami = ["Python", "C++", "Java", "Delphi"]
         fanlar = ["Dasturlash", "Axborot texnologiyalari"]

         list_toplami.extend(fanlar)

         print(list_toplami)

         ['Python', 'C++', 'Java', 'Delphi', 'Dasturlash', 'Axborot texnologiyalari']
```

# Python dasturlash tilida

## Tuple

Tuple - bu Python dasturlash tilida ma’lumotlar to‘plamini saqlash uchun ishlatiladi. Tuple to‘plami tartiblangan va o‘zgarmas to‘plam hisoblanadi.  Tuple to‘plamlari oddiy qavslar **()** bilan yoritiladi.

# Python dasturlash tilida Tuple

- Tuple to‘plami elementlari tartiblangan, o‘zgarmas va takroriy qiymatlarga ruxsat berilgan bo‘ladi. Tuple to‘plami elementlari indekslangan holatda bo‘ladi, birinchi element indeks [0]ga teng, ikkinchi element indeks[1] va hokazo bo‘ladi.
- Tuple to‘plami takroriy qiymatlarga ruxsat beradi:

```
In [5]: meva = ("olma", "behi", "gilos", "behi")

        print(meva)

        ('olma', 'behi', 'gilos', 'behi')
```

# Python dasturlash tilida Tuple

- Tuple to‘plamida nechta element borligini aniqlash uchun **len()** funksiyasidan foydalanish mumkin:

```
In [41]: dostlarim = ("Ulug`bek", "Kamoldin", "Omer Kamal")

         print(len(dostlarim))

         3
```

# Python dasturlash tilida Tuple

- Tuple to‘plami turli xil berilganlar turlarini o‘z ichiga olishi mumkin:

```
dostlarim = ("Ulug`bek",77, "Kamoldin", False, "Omer Kamal")

print(dostlarim)
```

```
('Ulug`bek', 77, 'Kamoldin', False, 'Omer Kamal')
```

# Python dasturlash tilida Tuple

• To‘rtburchak qavslar [] orqali indeks raqamiga murojaat qilib, **tuple** to‘plami elementlariga kirishingiz mumkin:

```
In [9]: meva = ("olma", 55, "gilos", "behi", True)

        print(meva[2])

        gilos
```

# Python dasturlash tilida Tuple

• Manfiy indekslash oxiridan boshlanadi degan ma’noni anglatadi. -1 oxirgi elementga, -2 ikkinchi oxirgi elementga to‘g‘ri keladi.

```
dostlarim = ("Ulug`bek",77, "Kamoldin", False, "Omer Kamal")

print(dostlarim[-3])
```

```
Kamoldin
```

# Python dasturlash tilida Tuple

• Siz tuple to‘plami indekslar oralig‘ini qayerdan boshlash va qayerdan tugatishni belgilash orqali bir qator indekslarni belgilashingiz mumkin. Indeks oralig‘ini belgilashda qaytariladigan qiymat belgilangan elementlarga ega yangi **tuple** to‘plami bo‘ladi.

```
In [45]: hayvonotOlami = ("arslon", "burgut", "yo`lbars")

         print(hayvonotOlami[0:2])

         ('arslon', 'burgut')
```

# Python dasturlash tilida Tuple

- Qidiruvni tuple to‘plami oxiridan boshlashni istasangiz, manfiy indeksdan foydalanishingiz mumkin:

```
In [50]: kompyuter_turi = ("lenovo", "dell", "acer")

         print(kompyuter_turi[-3:-1])

         ('lenovo', 'dell')
```

# Python dasturlash tilida Tuple

• Tuple to‘plami yaratilgandan so‘ng uning qiymatlarini o‘zgartirish mumkin emas. Tuple to‘plami o‘zgarmas to‘plam hisoblanadi. Buning yechimi, tuple to‘plamini list to‘plamiga aylantirish mumkin, list to‘plamini o‘zgartirish va list to‘plamini yana tuple to‘plamiga aylantirish mumkin. Quyidagi misolda amaliy holati keltirilgan:

In [53]:
```
kompyuter_turi = ("lenovo", "dell", "acer")
yangi_turi = list(kompyuter_turi)

yangi_turi[1] = "macintosh"

kompyuter_turi = tuple(yangi_turi)

print(kompyuter_turi)
```

```
('lenovo', 'macintosh', 'acer')
```

# Python dasturlash tilida Tuple

• Tulple to‘plamlar o‘zgarmas bo‘lgani uchun ular **append()** funksiyasiga ega emas, lekin tuple to‘plamiga elementlar qo‘shishning boshqa usullari mavjud. Birinchi usuli tuple to‘plamini list to‘plamiga aylantirish. Quyidagi misolda tuple to‘plamini listga aylantirib, "artel" elementini qo‘shish va uni yana tuple to‘plamiga aylantirish keltirilgan:

In [54]:
```
kompyuter_turi = ("lenovo", "dell", "acer")
yangi_turi = list(kompyuter_turi)

yangi_turi.append("artel")

kompyuter_turi = tuple(yangi_turi)

print(kompyuter_turi)
```

```
('lenovo', 'dell', 'acer', 'artel')
```

# Python dasturlash tilida Tuple

• Ikkinchi usuli bir tuple to‘plamini ikkinchi tuple to‘plamiga qo‘shish. Buning uchun + belgisidan foydalaniladi.

```
In [56]: telefon_turi = ("Mi", "iPhone", "Samsung", "Motorola")
         yangi_turi = ("Nokia",)

         telefon_turi += yangi_turi

         print(telefon_turi)

         ('Mi', 'iPhone', 'Samsung', 'Motorola', 'Nokia')
```

# Python dasturlash tilida Tuple

• Tuple to‘plami o‘zgarmas bo‘lganiligi uchun biz undagi elementlarni o‘chira olmaymiz. Tuple to‘plamidan elementlarni o‘chirish uchun yuqorida foydalanilganidek tuple to‘plamini list to‘plamiga o‘zgartirib undan elementlari o‘chirish mumkin. Quyidagi suratda amaliy misol keltirilgan:

```
In [59]: uni_nomlari = ("UMT", "ADU", "ANDMI", "UM")
         vaqtincha = list(uni_nomlari)
         vaqtincha.remove("ANDMI")
         uni_nomlari = tuple(vaqtincha)
         print(uni_nomlari)

         ('UMT', 'ADU', 'UM')
```

# Python dasturlash tilida Tuple

- Yoki siz tuple to‘plamini butunlay o‘chirib tashlashingiz mumkin:

```
In [60]: uni_nomlari = ("UMT", "ADU", "ANDMI", "UM")

         del uni_nomlari

         print(uni_nomlari)

         ---------------------------------------------------------------
         NameError                                 Traceback (most
         ~\AppData\Local\Temp/ipykernel_6668/3991707589.py in <modu
               3 del uni_nomlari
               4
         ----> 5 print(uni_nomlari)

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

# Python dasturlash tilida Tuple

- **For** siklidan foydalanib tuple to‘plami elementlarini ekranga chop etish mumkin.

```
In [61]: tillar = ("o`zbek", "ingliz", "rus", "olmon")

         for x in tillar:
             print(x)

         o`zbek
         ingliz
         rus
         olmon
```

# Python dasturlash tilida Tuple

- Tuple to‘plami elementlarini indeks raqamiga murojaat qilib sikl orqali chop etish mumkin.
- Buning uchun **range()** va **len()** funksiyalaridan foydalanish mumkin

```
In [64]: tillar = ("o`zbek", "ingliz", "rus", "olmon")

         for x in range(len(tillar)):
             print(tillar[x])

         o`zbek
         ingliz
         rus
         olmon
```

# Python dasturlash tilida Tuple

• Siz tuple to‘plami elementlarini while siklidan foydalanib ham chop etishingiz mumkin. Tuple to‘plami uzunligini aniqlash uchun **len()** funksiyasidan foydalanish mumkin, so‘ngra 0 dan boshlab ularning indekslariga murojaat qilib, tuple to‘plami elementlarini siklidan foydalanib chop etish mumkin.

```
In [66]: harflar = ("a", "b", "d", "e")

         i = 0

         while i < len(harflar):
             print(harflar[i])
             i = i + 1
```

```
a
b
d
e
```

# Python dasturlash tilida Tuple

- Ikki yoki undan ortiq tuple to‘plamlarini birlashtirish uchun + operatoridan foydalanish mumkin:

```
In [1]: uzbek_harflar = ('a', 'b', 'd', 'e', 'f')
        ruscha_harflar = ('а', 'б', 'д', 'э', 'ф')

        birlashtirish = uzbek_harflar + ruscha_harflar

        print(birlashtirish)

        ('a', 'b', 'd', 'e', 'f', 'а', 'б', 'д', 'э', 'ф')
```

# Python dasturlash tilida Tuple

• Agar tuple to‘plami tarkibini bir necha marta ko‘paytirmoqchi bo‘lsangiz, * operatoridan foydalanishingiz mumkin:

```
In [68]: raqamlar = ("1", "2", "3", "4")

         kopaytirish = raqamlar * 2

         print(kopaytirish)

         ('1', '2', '3', '4', '1', '2', '3', '4')
```

# Python dasturlash tilida Tuple

- **count()** funksiyasiga misol

```
In [3]: thistuple = (1, 3, 7, 3, 7, 5, 3, 6, 3, 5)

        x = thistuple.count(3)

        print(x)
```

```
4
```

# Python dasturlash tilida Tuple

- **index()** funksiyasiga misol

```
In [5]: thistuple = (1, 3, 7, 3, 7, 5, 3, 6, 3, 5)

        x = thistuple.index(7)

        print(x)

        2
```

# Foydalanilgan adabiyotlar


1. Mastering Object-Oriented Python: Build powerful applications with reusable code using OOP design patterns and Python 3.7, 2nd Edition, Steven F. Lott, Packt Publishing (June 14, 2019)

2. Learning Python, 5th Edition Fifth Edition, Mark Lutz , O'Reilly Media, June 12, 2013

3. Python Programming for Beginners: The Ultimate Guide for Beginners to Learn Python Programming: Crash Course on Python Programming for Beginners, AMZ Publishing, Independently published (July 13, 2021)

4. https://www.python.org/

5. https://www.w3schools.com/

6. https://www.codecademy.com/catalog/language/python

7. https://realpython.com/

8. https://www.anaconda.com/

# E’tiboringiz uchun rahmat!