PYTHON DASTURLASH TILIDA RO‘YXATLAR

Tojimamatov Israil Nurmamatovich
Farg‘ona davlat universiteti amaliy matematika va informatika kofedrasi katta o‘qituvchisi
israiltojimamatov@gmail.com
Mahmudov Oybekjon Shuxratjon o‘g‘li
Farg‘ona Davlat Universiteti 3- kurs talabasi
oybekjonmahmudov155@gmail.com
https://doi.org/10.5281/zenodo.15718149

Annotatsiya

Ushbu maqolada Python dasturlash tilidagi ro‘yxat (list) ma’lumotlar tuzilmasi haqida batafsil ma’lumot berilgan. Ro‘yxatlarning asosiy xususiyatlari, ularni yaratish usullari, elementlarga indeks orqali murojaat qilish, salbiy indekslash, ro‘yxatga element qo‘shish, olib tashlash, tartiblash, ko‘paytirish, nusxa olish va boshqa amallar ko‘plab kod misollari bilan izohlangan. Maqola dasturchilarga ro‘yxatlar bilan samarali ishlash, turli real hayotdagi masalalarni Python yordamida yechish imkoniyatlarini tushuntirishga qaratilgan.

Kalit so‘zlar

Python, ro‘yxat, list, dasturlash, ma’lumotlar tuzilmasi, indeks, append, pop, sort, slicing, list comprehension, mutable, tuple, dictionary, set.

Аннотация

В данной статье подробно рассматривается структура данных список (list) в языке программирования Python. Описаны основные характеристики списков, способы их создания, доступ к элементам по индексам, добавление, удаление, сортировка и другие операции. Цель статьи — обучить программистов эффективно работать со списками в Python.

Ключевые слова

Python, список, list, программирование, структура данных, индекс, append, pop, sort, срезы (slicing), list comprehension, изменяемость (mutable), кортеж (tuple), словарь (dictionary), множество (set)

Annotation

This article provides a detailed overview of the list data structure in the Python programming language. It explains the key features of lists, how to create them, access elements by index, add or remove elements, sort and manipulate them. The article aims to help developers use Python lists effectively in programming tasks.

Keywords

Python, list, programming, data structure, index, append, pop, sort, slicing, list comprehension, mutable, tuple, dictionary, set

Zamonaviy dasturlash tillari orasida Python o‘zining soddaligi va kuchli funksional imkoniyatlari bilan ajralib turadi. Python tilida dastur tuzishda eng ko‘p qo‘llaniladigan tuzilmalaridan biri bu — ro‘yxat (list) hisoblanadi. Ro‘yxatlar orqali turli xil turdagi qiymatlarni bitta o‘zgaruvchi ichida saqlab, ular ustida qulay amallar bajarish mumkin. Ayniqsa, foydalanuvchi tomonidan kiritilgan ma’lumotlarni vaqtincha saqlash, ularni tartiblash, tahlil qilish yoki filtrdan o‘tkazish kabi ko‘plab muammolarni ro‘yxatlar orqali oddiy va samarali hal qilish mumkin. Ushbu maqola Python ro‘yxatlarini chuqurroq o‘rganish, ularni yaratish, tahrirlash va qo‘llash imkoniyatlari bilan tanishtirishga qaratilgan.

Python dasturlash tilida ro‘yxatlar haqida umumiy tushuncha

Python dasturlash tilida **ro‘yxat (list)** — bu tartiblangan va o‘zgaruvchan (mutable) ma’lumotlar to‘plami bo‘lib, turli xil qiymatlarni bitta o‘zgaruvchi ichida saqlash imkonini beradi. Ro‘yxatlar dasturchilarga ma’lumotlar bilan qulay ishlash, ularni tartiblash, filtrlash, tahrirlash, va ko‘plab boshqa amallarni bajarish uchun juda keng imkoniyatlar yaratadi. Ro‘yxat yaratish uchun kvadrat qavslar [] ishlatiladi va elementlar vergul bilan ajratiladi. Masalan, mevalar = ["olma", "banan", "gilos"] bu yerda mevalar nomli ro‘yxat uchta satr tipidagi elementdan tashkil topgan.

Ro‘yxat elementlariga **indeks orqali** murojaat qilinadi va indekslash har doim **0 dan boshlanadi**. Shuningdek, Python’da **salbiy indekslar** ham mavjud bo‘lib, ular ro‘yxatning oxiridan boshlab sanaydi. Ro‘yxatlar o‘zgaruvchan bo‘lgani sababli, ularning istalgan elementini o‘zgartirish, yangi elementlar qo‘shish (append(), insert()) yoki mavjudlarini o‘chirish (remove(), pop(), del) mumkin.

Python ro‘yxatlari **aralash turdagi** qiymatlarni ham saqlashi mumkin. Ya’ni, siz bitta ro‘yxatda matn (string), butun son (int), haqiqiy son (float), mantiqiy qiymat (bool) va hatto boshqa ro‘yxatlarni ham saqlashingiz mumkin. Bu esa ro‘yxatni kuchli va moslashuvchan tuzilma qiladi.

Ro‘yxatlar bilan ishlashda ko‘pincha for sikli yordamida har bir element ustida amallar bajariladi. Bundan tashqari, in operatori yordamida ma’lum bir qiymat ro‘yxatda mavjudligini tekshirish mumkin. len() funksiyasi ro‘yxatdagi elementlar sonini aniqlaydi. Python’da ro‘yxatlar bilan ko‘plab funksiyalar va metodlar mavjud bo‘lib, ular dasturchiga real hayotdagi ko‘plab masalalarni soddalashtirgan holda yechishga yordam beradi.

Python dasturlash tilida **ro‘yxatning asosiy maqsadi** — bu bir nechta qiymatlarni bitta o‘zgaruvchida saqlash va ular ustida samarali ishlov berishdir. Dastur yozishda ko‘plab holatlarda birdan ortiq qiymatlar bilan ishlashga to‘g‘ri keladi. Masalan, o‘quvchilar ro‘yxati, mahsulotlar narxi, kundalik harorat, foydalanuvchilar kiritgan ma’lumotlar kabi ko‘p sonli va tartibli ma’lumotlarni alohida-alohida o‘zgaruvchilar o‘rniga yagona ro‘yxatda jamlab olish qulay va samarali yechim hisoblanadi.

Ro‘yxatlar orqali ma’lumotlar **tartibli** saqlanadi. Bu degani — har bir element o‘z indeksiga ega bo‘ladi. Shu tufayli, ro‘yxatdagi istalgan elementga bevosita murojaat qilish, uni o‘zgartirish yoki o‘chirish mumkin bo‘ladi. Dastur tuzishda bu xususiyat **ma’lumotlar oqimini boshqarish**, foydalanuvchidan kelgan kirish qiymatlarini qayta ishlash va **natijalarni chiqarish** jarayonlarini sezilarli darajada soddalashtiradi.

Dasturdagi ro‘yxatlarning o‘rni ayniqsa **sikl (loop)** va **shartli operatorlar** bilan birgalikda ishlatilganda kuchli seziladi. Masalan, for sikli yordamida ro‘yxatdagi barcha elementlarni aylantirib chiqish, ularning ustida amallar bajarish yoki if yordamida ularni tahlil qilish juda qulay. Shuningdek, ro‘yxatlar foydalanuvchidan kiritilgan bir nechta qiymatlarni vaqtincha saqlab turish yoki bazadagi obyektlarni qayta ishlashda ham keng qo‘llaniladi.

Ko‘pgina dasturlar, ayniqsa **veb ilovalar, hisob-kitob tizimlari, ro‘yxat asosida filtrlovchi dasturlar** va hatto sun’iy intellekt algoritmlarida ham ro‘yxatlar asosiy tuzilmalardan biri sifatida ishlatiladi. Ular yordamida **dastur modulliligi, kod soddaligi va o‘qilishi** oshadi.

Python dasturlash tilida bir nechta asosiy ma’lumotlar tuzilmalari mavjud: **ro‘yxat (list)**, **kortej (tuple)**, **to‘plam (set)** va **lug‘at (dictionary)**. Ularning har biri o‘ziga xos tuzilishga va foydalanish holatlariga ega. Ro‘yxat esa ulardan eng moslashuvchan va ko‘p qo‘llaniladigan tuzilmadir. Quyida ro‘yxatning boshqa tuzilmalar bilan asosiy farqlari yoritiladi.

Birinchidan, **ro‘yxat va kortej** orasidagi farq shundaki, ro‘yxat — **o‘zgaruvchan (mutable)**, ya’ni unga yangi element qo‘shish, o‘chirish yoki mavjud elementni o‘zgartirish mumkin. Kortej esa **o‘zgarmas (immutable)** bo‘lib, yaratilgandan keyin uni o‘zgartirib bo‘lmaydi. Misol uchun:

python
ro‘yxat = [1, 2, 3]
kortej = (1, 2, 3)
ro‘yxat[0] = 10 — to‘g‘ri ishlaydi, kortej[0] = 10 — xato beradi.

Ikkinchidan, **ro‘yxat va to‘plam (set)** farqlanadi. Ro‘yxat tartibli tuzilmadir — unda elementlar indekslangan va ketma-ket saqlanadi. To‘plam esa **tartibsiz** va **faqat noyob qiymatlarni** saqlaydi. Bu degani, ro‘yxatda bir xil elementlar bir necha marta bo‘lishi mumkin, lekin to‘plamda faqat bittasi qoladi:

python
ro‘yxat = [1, 2, 2, 3]
to‘plam = {1, 2, 2, 3} # to‘plamda faqat {1, 2, 3}

Uchinchidan, **ro‘yxat va lug‘at (dictionary)** o‘rtasida ham sezilarli farq mavjud. Ro‘yxat elementlari **indeks** orqali murojaat qilinadi, lug‘at elementlari esa **kalit-qiymat (key-value)** juftliklari orqali boshqariladi. Ro‘yxatlar ketma-ketlikda saqlangan qiymatlar uchun mos kelsa, lug‘atlar aniqlangan atributlarga ega bo‘lgan obyektlar yoki ma’lumotlar uchun mos keladi:

python
ro‘yxat = ["Ali", "Vali"]
lug‘at = {"ism": "Ali", "yosh": 25}

Ro‘yxatlar boshqa tuzilmalar bilan solishtirilganda, ularning eng katta afzalligi — **moslashuvchanligi va foydalanishdagi qulayligi**dir. Ro‘yxat yaratish oson, ularga ishlov berishning ko‘plab metodlari mavjud (append(), pop(), sort() va hokazo), shuningdek ular deyarli barcha algoritmlarda qo‘llaniladi.

Python dasturlash tilida **ro‘yxat (list)** — bu bir nechta qiymatlarni tartibli ravishda saqlovchi ma’lumotlar tuzilmasidir. Ro‘yxat yaratishning bir nechta usullari mavjud va ular dasturdagi ehtiyojga qarab tanlanadi. Quyida Python’da ro‘yxat yaratishning asosiy va keng tarqalgan usullari yoritiladi.

1. Kvadrat qavslar yordamida ro‘yxat yaratish

Python’da ro‘yxat yaratishning eng sodda usuli — bu [] kvadrat qavslar ichida elementlarni yozishdir. Har bir element vergul bilan ajratiladi:

python
mevalar = ["olma", "banan", "shaftoli"]
sonlar = [1, 2, 3, 4, 5]
aralash = ["salom", 3.14, True, 100]

Ushbu usul eng ko‘p qo‘llaniladi, ayniqsa statik ma’lumotlar bilan ishlaganda.

2. list() funksiyasi yordamida ro‘yxat yaratish

list() — bu Python’ning ichki funksiyasi bo‘lib, boshqa ma’lumotlar tuzilmalarini ro‘yxatga aylantirish uchun ishlatiladi. Misol uchun:

python

raqamlar = list((1, 2, 3)) # kortejdan ro‘yxat
matn = list("salom") # stringdan ro‘yxat: ['s', 'a', 'l', 'o', 'm']

Bu usul boshqa ma’lumotlar tuzilmalarini (tuple, set, string) ro‘yxatga aylantirishda foydalidir.

3. Bo‘sh ro‘yxat yaratish

Ba’zan dasturchi dastur davomida ma’lumot qo‘shish uchun avval bo‘sh ro‘yxat yaratadi:

python
bo‘sh = []
yoki
bo‘sh = list()

Keyinchalik .append() yoki .insert() metodlari orqali unga elementlar qo‘shiladi.

4. Takrorlanuvchi (generator) ifodalar orqali ro‘yxat yaratish

Python’da **list comprehension** orqali qisqa va aniq ro‘yxatlar yaratish mumkin:

python
kvadratlar = [x**2 for x in range(1, 6)] # [1, 4, 9, 16, 25]
juftlar = [x for x in range(10) if x % 2 == 0] # [0, 2, 4, 6, 8]

Bu usul kodni ixcham va o‘qilishi oson qiladi.

5. Dastur davomida foydalanuvchidan ma’lumot olib ro‘yxat yaratish

Ko‘pincha foydalanuvchidan ma’lumotlar olinib, ro‘yxatga qo‘shiladi:

python
raqamlar = []
for i in range(5):
son = int(input(f"{i+1}-sonni kiriting: "))
raqamlar.append(son)

Bu dinamik ro‘yxat yaratish usuli hisoblanadi.

Python dasturlash tilida **ro‘yxat (list)** — bu tartiblangan ma’lumotlar to‘plami hisoblanadi. Tartibli tuzilma bo‘lgani sababli, ro‘yxat ichidagi har bir element o‘z **indeks raqami** bilan aniqlanadi. Indeks orqali biz ro‘yxatdagi har qanday elementga tezda murojaat qilishimiz, uni o‘zgartirishimiz yoki ustida amallar bajarishimiz mumkin.

Indekslash (indexing)

Python’da ro‘yxatlar **0-indeksdan** boshlanadi. Ya’ni, ro‘yxatdagi birinchi elementning indeksi — 0, ikkinchi elementniki — 1, va hokazo. Quyidagi misolni ko‘rib chiqamiz:

```
python
mevalar = ["olma", "banan", "shaftoli", "uzum"]
print(mevalar[0])  # "olma"
print(mevalar[2])  # "shaftoli"
```

Bu yerda mevalar[0] birinchi elementni, mevalar[2] esa uchinchi elementni chaqiradi. Indeks orqali nafaqat o‘qish, balki o‘zgartirish ham mumkin:

```
python
mevalar[1] = "anor"
print(mevalar)  # ["olma", "anor", "shaftoli", "uzum"]
```

Salbiy indekslar (negative indexing)

Python ro‘yxatlari **salbiy indekslash** imkoniyatiga ham ega. Bu imkoniyat ro‘yxat oxiridan boshlab elementlarga murojaat qilishni bildiradi. Oxirgi element indeksi -1, undan oldingisi -2, va hokazo:

```
python
print(mevalar[-1])  # "uzum"
print(mevalar[-2])  # "shaftoli"
```

Salbiy indekslar dasturchiga ro‘yxatning oxiridan elementlarni tez va qulay chaqirish imkonini beradi, ayniqsa ro‘yxat uzun bo‘lsa va oxirgi elementlar bilan ishlash talab qilinsa.

Xatoliklardan ehtiyot bo‘lish

Agar siz ro‘yxatda mavjud bo‘lmagan indeksni chaqirmoqchi bo‘lsangiz, Python IndexError xatosini chiqaradi:

```
python
print(mevalar[10])  # IndexError: list index out of range
```

Shuning uchun, har doim len(ro‘yxat) yordamida ro‘yxat uzunligini bilib olish va indekslar chegarasidan chiqmaslik tavsiya etiladi.

Python dasturlash tilida **ro‘yxatlar** (lists) bilan ishlash uchun ko‘plab qulay **amallar**, **ichki metodlar** va **funksiyalar** mavjud. Ular dasturchiga ro‘yxatni yaratish, unga element qo‘shish, olib tashlash, tartiblash, ko‘paytirish, bo‘shatish kabi amallarni tez va qulay bajarish imkonini beradi.

1. Ro‘yxatga element qo‘shish metodlari

- append(element) — ro‘yxat oxiriga element qo‘shadi:

```
python
mevalar = ["olma", "banan"]
mevalar.append("anor")  # ["olma", "banan", "anor"]
```

- insert(indeks, element) — belgilangan joyga element qo‘shadi:

```
python
```

```
mevalar.insert(1, "gilos")  # ["olma", "gilos", "banan", "anor"]
```

• extend(boshqa_ro‘yxat) — boshqa ro‘yxat elementlarini birlashtiradi:

```
python
sonlar = [1, 2]
sonlar.extend([3, 4])  # [1, 2, 3, 4]
```

2. Elementni o‘chirish va ajratib olish

• remove(element) — ro‘yxatdagi birinchi uchragan elementni o‘chiradi:

```
python
mevalar.remove("banan")  # ["olma", "gilos", "anor"]
```

• pop([indeks]) — indeks bo‘yicha elementni olib tashlaydi va uni qaytaradi:

```
python
mevalar.pop()     # oxirgi elementni o‘chiradi
mevalar.pop(0)    # birinchi elementni o‘chiradi
```

• del operatori — indeks yoki kesma bo‘yicha o‘chiradi:

```
python
del mevalar[1]
del mevalar[:]  # ro‘yxatni tozalash
```

• clear() — butun ro‘yxatni bo‘shatadi:

```
python
mevalar.clear()  # []
```

3. Elementlarni tartiblash va teskari aylantirish

• sort() — ro‘yxatni o‘sish tartibida saralaydi:

```
python
sonlar = [5, 2, 9]
sonlar.sort()  # [2, 5, 9]
```

• reverse() — ro‘yxatni teskari tartibda aylantiradi:

```
python
sonlar.reverse()  # [9, 5, 2]
```

• sorted() — saralangan ro‘yxat nusxasini qaytaradi (aslini o‘zgartirmaydi):

```
python
yangi = sorted(sonlar)
```

4. Ko‘paytirish, kesish, uzunlik

• Ro‘yxatni ko‘paytirish:

```
python
[1, 2] * 3  # [1, 2, 1, 2, 1, 2]
```

• Ro‘yxat kesmasi (slicing):

```
python
```

raqamlar = [0, 1, 2, 3, 4, 5]
raqamlar[1:4] # [1, 2, 3]

• len() — ro‘yxat uzunligini aniqlaydi:

python
len(raqamlar) # 6

5. Nusxa olish va tekshiruvlar

• copy() — ro‘yxatning nusxasini yaratadi:

python
a = [1, 2, 3]
b = a.copy()

• in operatori — element mavjudligini tekshiradi:

python
if "olma" in mevalar:
print("Olma mavjud")

Xulosa

Maqoladan ko‘rinib turibdiki, Python dasturlash tilidagi ro‘yxatlar — bu dasturchi uchun juda qulay, moslashuvchan va ko‘p funksiyali ma’lumotlar tuzilmasidir. Ularning tartibli va o‘zgaruvchan bo‘lishi sababli, dastur tuzishda keng ko‘lamli muammolarni hal qilishda ro‘yxatlar markaziy o‘rin egallaydi. Ro‘yxatlar yordamida ma’lumotlarni tizimli saqlash, ularni qayta ishlash va foydalanuvchi ehtiyojiga mos holda tahrirlash mumkin. Dasturchilar uchun ro‘yxatlar bilan ishlashni chuqur o‘zlashtirish — samarali va soddalashtirilgan kod yozish yo‘lida muhim qadamlardan biridir. Shuningdek, ro‘yxatlarni boshqa tuzilmalar bilan taqqoslab o‘rganish, ularning imkoniyatlarini to‘g‘ri tanlashda muhim ahamiyat kasb etadi.

Foydalanilgan adabiyotlar:

1. Python rasmiy hujjatlari – https://docs.python.org
2. Al Sweigart, Automate the Boring Stuff with Python, No Starch Press
3. Eric Matthes, Python Crash Course, No Starch Press
4. W3Schools Python Tutorial – https://www.w3schools.com/python
5. Geeks for Geeks Python List – https://www.geeksforgeeks.org/python-list/
6. Real Python – https://realpython.com