PYTHON DASTURLASH TILIDA MA'LUMOT TO'PLAMLARI VA TURLARI HAMDA ULAR BILAN ISHLASH.

Qulliyev Javohirbek G'anijon o'g'li
Buxoro davlat pedagogika instituti "Aniq fanlar" kafedrasi o'qituvchisi

Annotatsiya: *Python dasturlash tili ma'lum oddiy yoki chiziqli dasturlarni yozish uchun emas balki bu tilda turli murakkablikdagi loyihalarni ishlab chiqish mumkin . Bu til dasturchilarga yangi va yangi yo'nalishlarga kirish imkonini bermoqda. Python quyidagi sohalarda qo'llaniladi: Kiber xavfsizlikga oid masalalaeni hal qilishda, Web va Internet dasturlash, kompyuter o'yinlarini yaratish, ma'lumotlar bazasi bilan ishlash (DB), computer vision, suniy intelekt, juda tez rivojlanayotgan buyumlar interneti (IoT) texnologiyasi va hokazo. Ushbu maqolada biz python dasturlash tilida ma'lumotlar to'plamlari bilan tanishamiz hamda ular bilan ishlashni ko'rib chiqamiz.*

Kalit so'zlar: *list, append, remove, pop, reverse, index.*

КОЛЛЕКЦИИ, ТИПЫ И РАБОТА С НИМИ НА ЯЗЫКЕ ПРОГРАММИРОВАНИЯ PYTHON.

Куллиев Жавохирбек Ганижон угли
Бухарский государственный педагогический институт Преподаватель кафедры «Точные науки»

Аннотация: *Язык программирования Python не предназначен для написания каких-то простых или линейных программ, но на этом языке можно разрабатывать проекты различной сложности. Этот язык позволяет программистам выходить на все новые и новые направления. Python используется в следующих областях: решение проблем кибербезопасности, веб- и интернет-программирование, создание компьютерных игр, управление базами данных (БД), компьютерное зрение, искусственный интеллект и быстро развивающаяся технология Интернета вещей (IoT). и так далее. В этой статье мы хорошо узнаем о наборах данных на языке программирования Python и о том, как с ними работать.*

Ключевые слова: *list, append, remove, pop, reverse, index.*

COLLECTIONS, TYPES AND WORKING WITH THEM IN THE PYTHON PROGRAMMING LANGUAGE.

Qulliyev Javohirbek G'anijon o'g'li
Bukhara State Pedagogical Institute Teacher of the "Exact Sciences" department

Abstract: *Python programming language is not for writing certain simple or linear programs, but it is possible to develop projects of various complexity in this language. This language allows programmers to enter new and new directions. Python is used in the following areas: Cybersecurity problem solving, Web and Internet programming, computer game creation, database management (DB), computer vision, artificial intelligence, and the rapidly developing Internet of Things (IoT) technology. and so on. In this article, we well learn about datasets in the python programming language and how to work with them.*

Keywords: *list, append, remove, pop, reverse, index.*

KIRISH

Boshqa dasturlash tillaridan farqli ravishda python dasturlash tilida List yordamida bir o‘zgaruvchida ko‘pgina qiymatlarni yoki o‘zgaruvchilarni saqlashni o‘rganamiz. Ro‘yxat o‘z nomi bilan, bitta o‘zgaruvchida bir nechta qiymatlarni saqlash imkonini beradi. Bu qiymatlar List(ro‘yxat) elementlari deyiladi. Ro‘yxatda son, matn yoki aralash turdagi elementlarni saqlash mumkin. Ushbu maqolada biz list va tuple ma’lumotlar to‘plami va ular ustida turli a‘mallarni bajarishni ko‘rib chiqamiz.

List quyidagicha yaratiladi:

```
talabalar= ['ali', 'nozim', 'kozim', "asqar"] # talabalar ro‘yxati (matnlar)
sonlar = [700, 36, 85, 97] # sonlar ro‘yxati (sonlar)
raqamlar = ['bir', 'besh', 8, 7, 3] # raqamlar va matnlar aralash ro'yxat
ismlar = [] # bo'sh ro‘yxat
```

Ro‘yxatni(list) saqlaydigan o‘zgaruvchilarni nomlashda -lar dan foydalanish maqsadga muvofiq bo‘ladi.

Misol uchun: sonlar, uylar, talabalar, kitoblar, bolalar.

Ro‘yxatdagi har bir element tartib bilan joylashgani sababli, biz istalgan elementga uning tartib raqami (indeksi) bo‘yicha murojaat qilishimiz mumkin.

Dasturlash tillarida indeks 0 dan boshlanadi. Yani Listning birinchi elementining tartib raqami (indeksi) 0 ga, ikkinchi elementining indeksi esa 1 ga teng va hokazo.

```
sonlar = [700, 36, 85, 97] # sonlar ro‘yxati.
print("Birinchi son: ", sonlar[0])
print("Ikkinchi son: ", sonlar[1])
```

Natija:

Birinchi son: 700

Ikkinchi son: 36

List elementlari ustida arifmetik amallar bajarish:

```
sonlar = [700, 36, 85, 97]
print(sonlar[2] + sonlar[3])
```

Natija:182

Pythonda Listning eng oxirgi elementiga -1 indeksi orqali ham murojat qilish mumkin. Bu usul Listning uzunligini bilmaganda qulay xisoblanadi.

```
raqamlar = ['bir', 'besh', 8, 7, 3]
print(raqamlar[-1]) # Listning eng oxirgi elementiga -1 bilan murojat qilamiz
```

Natija: 3

Dastur davomida listning tarkibi o‘zgarishi, yangi elementlar qo‘shilishi, bazi elementlar o‘chirilishi oddiy hol.

Elementlarni o'zgartirish. Ro‘yxatdagi biror elementning qiymatini o‘zgartirish uchun, o‘sha elementga indeks bo‘yicha murojat qilamiz va yangi qiymat yuklaymiz.

```
narhlar = [5600, 4400, 18700, 26200]
narhlar[0] = 13000 # 1-qiymatni 13000 ga o'zgartiramiz
narhlar[2] = 11000 # 3-qiymatni 11000 ga o'zgartiramiz
narhlar[3] = narhlar[3]+2000 # 4-qiymatga 2000 qo'shamiz
print(narhlar)
```

Natija: [13000, 4400, 11000, 28200]

.append() metodi yordamidan yangi element qo‘shish. Ro‘yxatga yangi element qo‘shishning oson usuli bu .append() metodi yordamida ro‘yxatning oxiriga qiymat qo‘shishdir:

```
mevalar = ['olma', 'anjir', 'shaftoli', "o'rik"]
mevalar.append("kivi") # mevalarga kivi elementini qo'shamiz
print(mevalar)
```

Natija: ['olma', 'anjir', 'shaftoli', "o'rik", 'kivi']

.append() metodi bo‘sh ro‘yxatni to‘ldirishda juda qulay usul. Odatda dastur boshida bo‘sh ro‘yxat yaratilib, dastur davomida ro'yxat foydalanuvchi tomonidan to‘ldirib borilishi odatiy hol.

```
mevalar = []  # bo‘sh ro‘yxat append bilan uni to‘ldiramiz
mevalar.append("uzum")
mevalar.append("anor")
mevalar.append("nok")
mevalar.append("olma")
mevalar.append("qovun")
mevalar.append("shaftoli")
print(mevalar)
```

Natija : ['uzum', 'anor', 'nok', 'olma', 'qovun', 'shaftoli']

Ro‘yxatning istalgan joyiga yangi element qo‘shish uchun .insert() metodidan foydalanamiz. .insert() metodi ichida yangi elementning indeksi va qiymati beriladi:

```
mevalar = ['uzum', 'anor', 'nok', 'olma', 'qovun', 'shaftoli']
mevalar.insert(0, 'avakado') # 1-o'ringa yangi qiymat qo'shamiz
print(mevalar)
```

Natija: ['avakado', 'uzum', 'anor', 'nok', 'olma', 'qovun', 'shaftoli']

```
mevalar.insert(2, 'banan') # 3-o‘ringa yangi qiymat qo‘shamiz
print(cars)
```

Natija: ['avakado', 'uzum', 'banan', 'anor', 'nok', 'olma', 'qovun', 'shaftoli']

del yordamida elementni o‘chirish uchun yoki ro‘yxatdan biror elementni olib tashlash uchun uning indeksini yoki qiymatini bilish kerak bo‘ladi. Indeks yordamida olib tashlash uchun del operatoridan quyidagicha foydalanamiz:

```
mevalar = ['olma', 'anjir', 'shaftoli', "o'rik", 'anor']
del mevalar[1] # 2-element (anjir) ni o‘chirib tashlaymiz
```

print(mevalar)

Natija: ['olma', 'shaftoli', "o'rik", 'anor']

Elementlarni qiymati bo‘yicha o‘chirish uchun esa .remove(qiymat) metodidan foydalanamiz. Buning uchun qavs ichida o‘chirib tashlash kerak bo‘lgan qiymatni yozamiz:

mevalar = ['olma', 'anjir', 'shaftoli', "o'rik", 'anor']

mevalar.remove('shaftoli') # Ro'yxatdan shaftolini o‘chirdik

print(mevalar)

Natija: ['olma', 'anjir', "o'rik", 'anor']

.remove(qiymat) metodi ro‘yxatda uchragan birinchi mos keluvchi qiymatni o‘chiradi. Agar ro‘yxatning ichida 2 va undan ko‘p bir hil qiymatli elementlar bo‘lsa, ulardan eng birinchisi o‘chadi.

hayvonlar = ['ot', 'mushuk', 'it', tuya', 'quyon', 'mushuk']

hayvonlar.remove("mushuk") # Ro‘yxatda 2 ta mushuk bor, ulardan birinchisi o‘chadi

print(hayvonlar)

Natija: ['it', 'sigir', "qo'y", 'quyon', 'mushuk']

Bazida biror elementni butunlay o‘chirib tashlash emas, balki uni ro‘yxatdan chiqarib olish va undan foydalanish talab qilinishi mumkin. Buning uchun Pythonda .pop(indeks) metodidan foydalanmiz.

mevalar = ["kivi'", 'shaftoli', 'ananas', 'banan', "mandarin"]

mahsulot = mevalar(3) # Ro‘yxatdan bananni sug’urib olamiz

print("Men " + mahsulot + " sotib oldim")

print("Olinmagan mevalar: ", mevalar)

Natija:

Men banan sotib oldim

Olinmagan mahsulotlar: ["kivi'", 'shaftoli', 'ananas', "mandarin"]

Agar .pop() metodida indeks berilmasa, ro‘yxatdan oxirgi element chiqarib olinadi.

Ro‘yxatni tartiblash. Ayrim hollarda list ichidagi elementlarni alifbo ketma-ketligida tartiblash talab qilinishi mumkin. Bunda ro‘yxat uchun maxsus .sort() metodidan foydalanamiz.

moshinalar = ['cobalt','nexia', 'malibu', 'byd', 'tesla', 'audi','toyota']

moshinalar.sort()

print(moshinalar)

Natija: ['audi', 'byd', 'cobalt', 'malibu', 'nexia', 'tesla', 'toyota']

Ko‘rib turganizdek, yuqoridagi ro‘yxat alifbo bo‘yicha tartiblandi.

E’tibor bering! Tartiblashda katta harflar kichik harflardan avval kelishini hisobga oling. Agar matndagi so‘zlarning bosh harfi katta-kichik aralash yozilgan bo‘lsa, ularni bir ko‘rinishga keltirib olish maqsadga muvofiq bo'ladi.

Bazida ro‘yxatni aylantirish yani boshini oxiriga, oxirini esa boshiga qilib chiqarish talab qilinishi mumkin. Buning uchun .reverse() metodidan foydalanamiz.

moshinalar = ['cobalt','nexia', 'malibu', 'byd', 'tesla', 'audi','toyota']

moshinalar.reverse()

print(moshinalar)

Natija: ['toyota', 'audi', 'tesla', 'byd', 'malibu', 'nexia', 'cobalt']

Ko‘rganingizdek haqiqiy list bilan natijani solishtirishingiz mumkin.

Ro‘yxatning uzunligini topish. Ayrim o‘rinlarda ro‘yxatning uzunligini, yani uning elementlar sonini aniqlash uchun len() funksiyasidan foydalanamiz:

```
moshinalar = ['cobalt','nexia', 'malibu', 'byd', 'tesla', 'audi','toyota']
print("Elementlar soni:",len(moshinalar)) # len(moshinalar) ro‘yxat uzunligini qaytaradi.
```

Natija: Elementlar soni: 7

range() funksiyasi ushbu funksiya yordamida biz ma‘lum oraliqdagi sonlar ketma-ketligini hosil qilishimiz mumkin. list() funktsiyasi yordamida esa bu oraliqni ro‘yxat shaklida saqlab olamiz:

```
raqamlar = list(range(0,10))
print(raqamlar)
```

Natija: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

Yuqoridagi misolda range(0,10) funksiyasi 0 dan 9 gacha raqamlarlar ketma-ketligini shakllantirdi, list(range(0,9)) esa bu ketma-ketlikni ro‘yxatga aylantirdi.

E’tibor qiling range() funksiyasi ikkinchi indeksdan bitta avval to‘xtaydi.

range() yordamida qadamni ham berish mumkin:

```
juft_sonlar = list(range(0,20,2)) # 0 dan 20 gacha 2 qadam bilan
toq_sonlar = list(range(1,20,2))  # 1 dan 20 gacha 2 qadam bilan
print("Juft sonlar: ", juft_sonlar)
print("Toq sonlar: ", toq_sonlar)
```

Natija:

Juft sonlar: [0, 2, 4, 6, 8, 10, 12, 14, 16, 18]

Toq sonlar: [1, 3, 5, 7, 9, 11, 13, 15, 17, 19]

Agar sonlar ketma-ketligi 0 dan boshlansa, range() funktsiyasida yakuniy indeksni ko‘rsatish kifoya. Misol uchun range(0,10) emas range(10) deb yozish kifoya bo‘ladi.

Bazida ro‘yxatning ma‘lum bir bo‘lagini ajratib olish talab qilinishi mumkin, biz quyidagi cars degan ro'yxatdan birinchi 3 ta elementni ajratib olmoqchimiz, buning uchun biz bo shlang'ich va oxirgi indekslarni beramiz:

```
moshinalar = ['cobalt','nexia', 'malibu', 'byd', 'tesla', 'audi','toyota']
mening_moshinalarim= moshinalar[0:3] # 0-indeksdan boshlab 3 ta element ajratib olamiz
print(mening_moshinalarim
```

Natija: ['cobalt', 'nexia', 'malibu']

Diqqat! Python 2-indeksdan bitta avval to'xtaydi. Yuqoridagi misolda ham 0,1,2-elementlar ajratib olindi.

Bu usul bilan ro‘yxatning istalgan joyidan bo‘lishimiz mumkin:

```
print(moshinalar[2:5]) # 2-3-4-elementlarni ajratib olamiz (5 kirmaydi)
```

Natija: ['malibu', 'byd', 'tesla']

Agar boshlang’ch indeksni bermasangiz, Python avtomat ravishda 0 indeksdan boshlab kesadi. Agar 2-indeksni kiritmasangiz, ro‘yxat oxirigacha kesadi:

```
print(moshinalar[:4]) # Ro‘yxat boshidan 4-gacha kesadi (0,1,2,3)
print(moshinalar[2:]) # 2-elementdan boshlab ro‘yxat oxirigacha kesib oladi
```

Natija:

['cobalt', 'nexia', 'malibu', 'byd']

['malibu', 'byd', 'tesla', 'audi', 'toyota']

Tuples – o‘zgarmas ro‘yxat. Dastur yozish davomida o‘zgarmas ro‘yxat tuzish talab qilinishi mumkin. Pythonda bunday ro‘yxatlar tuples deb yuritiladi. Tuple ichidagi qiymatlarni bir marta, dastur boshida beriladi va keyin o‘zgartirib bo‘lmaydi. List dan farqli ravishda, Tuple e’lon qilishda kvadrat qavslar [] o‘rniga oddiy qavslar () ishlatiladi:

sonlar = (18,19,20, 30, 55,2)

print(tomonlar)

Natija: (20, 30, 55.2)

Tuple ichidagi elementlarga huddi ro‘yxat elementlariga murojaat qilingani kabi murojat qilinaveradi:

moshinalar = ('cobalt','nexia', 'malibu', 'byd', 'tesla', 'audi','toyota')

print(moshinalar[0])

print(moshinalar[-1])

print(moshinalar[2:5])

Natija:

cobalt

toyota

('malibu', 'byd', 'tesla')

Tuple ichidagi elementlarni qiymatini o‘zgartirib bo‘lmaydi:agar, bu operatsiyalarni bajarsangiz yani element qo‘shish, o‘chirish va hokazo shu kabi amallarni bajarmoqchi bo‘lsangiz dastur natijasi xatolikka olib keldi. Tuple ro‘yxatdan biror elementini o‘chirish yoki yangi element qo‘shish mumkin emas.

Agar Tuple ga o‘zgartirish kiritish talab qilinsa, yagona yo‘li o‘zgarmas ro‘yxatni list() funksiyasi yordamida List (oddiy ro‘yxat) ko'rinishiga keltirib olish, keyin esa o‘zgartirishlarni bajarsih va qaytarib tuple() funksiyasi yordamida o‘zgarmas ro‘yxatga o‘tkazish mumkin:

moshinalar = ('cobalt','nexia', 'malibu', 'byd', 'tesla', 'audi','toyota') # o'zgarmas ro‘yxat

moshinalar = list(moshinalar) # o‘zgarmas ro‘yxatni oddiy ro‘yxatga (List) aylantiramiz va ro‘yxatga o‘zgartirishlar kiritamiz

moshinalar.append('cadilac')

moshinalar.remove('nexia')

moshinalar[1] = 'ferrari'

moshinalar = tuple(moshinalar) # Ro'yxatni qaytadan o'zgarmas ro'yxatga (Tuple) aylantiramiz

print(moshinalar)

Natija: ('cobalt', 'ferrari', 'byd', 'tesla', 'audi', 'toyota', 'cadilac')

Xulosa. Umuman olganda python dasturlash tilida ma’lumot to‘plamalari bilan ishlash boshqa dasturlash tillariga qaraganda osonroq va soddaroq hisoblanadi hamda tilning foydalanish imkoniyatlarini oshiradi. Ushbu maqola orqali biz faqat qisman ma’lumotlarni keltirb o‘tdik.

FOYDALANILGAN ADABIYOTLAR:

1. www.python.org.
2. www.w3schools.com.
3. www.geeksforgeeks.org.