JAVDAL MIQDORLARGA OID MASALALARGA DASTUR TUZISHDA PYTHON DASTURLASH TILINING IMKONIYATLARIDAN FOYDALANISH

Djumoboyeva Yanglish Egamnazarovna - Guliston davlat universiteti tayanch doktorant
E-mail: Yanglish8392@gmail.com

Annotatsiya: Ushbu maqolada ko‘plab dasturlash masalalarini yechishda Python dasturlash tilining List ma’lumotlar tipidan foydalaniladi, Listlar boshqa dasturlash tillarida mavjud bo‘lgan array(massiv) ga o‘xshash ma’lumot tipi bo‘lib, lekin ba’si imkoniyatlari bilan ulardan farq qiladi. Ushbu maqolada Listlar haqida batafsil ma’lumot berish bilan birga bir nechta dasturlash masalalarini yechiimini keltirdik.

Kalit so‘zlar: Ro‘yxat (list), o‘zgaruvchi, elementlar, list elementlarini o‘zgartirish, nusxalash, birlashtirish, list metodlar, append, insert, clear.

Bo‘lajak informatika o‘qituvchilarning dasturlashga oid kasbiy kompetenligini rivojiantirish metodikasini takomillashtirish kommunikativ kompetentligini shakllantirish muvaffaqiyati, kasbini san’at darajasiga olib chiqib, o‘nlab, yuzlab shogirdlar tayyorlaganligi uchun berilgan unvondir. Yoki ma’lum bir dunyoga yangi ko‘z bilan qaraydigan, uddaburon, ishning ko‘zini biluvchi, buyuk kelajagimiz poydevorini quruvchi va yuksaltiruvchi mutaxassis kadrlarni tayyorlash Respublikamiz pedagoglari oldida turgan eng muhim va mas’uliyatli vazifadir. Pedagog o‘z ixtisosligi bo‘yicha ta’lim va tarbiya ishlari yuzasidan jamiyat oldida mas’uldir. U muntazam ravishda o‘z kasbiy tayyorgarligini takomillashtirib borish bilan birgalikda o‘qituvchi, muallim, ustoz, tarbiyachi, pedagog huquqiga ega bo‘lishi, aniqrog‘i, kommunikativ kompetensiya egasi bo‘lishi lozim. Endilikda o‘qituvchi uchun o‘zi tanlagan kasb, fan, mutaxassislik borasidagi amaliy va nazariy bilimlarni puxta egallash kamlik qilib qoldi. U har qanday vaziyatda ta’limtarbiyaning yangi-yangi usul va metodlarini o,,z faoliyatida qo‘llay olishi, bir so‘z bilan aytganda, kompetensiya egasi bo‘lishi lozim.

Programmalash asoslarini mukammal o‘rganish bugungi kunda jamiyatni axborotlashtirish jarayonini samarali tashkil qilishda, turli ishlab chiqarish korxonalarida va tashkilotlarda vujudga kelgan texnik va iqtisodiy masalalarni echishda muhim o‘rinni egallaydi. Programma yaratish dasturlashtirishning turli rivojlanish bosqichlarida o‘ziga xos o‘ringa ega bo‘lib kelgan. Kompyuterlarning imkoniyatlari oshishi, dasturlashtirishning vosita va uslublari o‘sishi bilan kompyuterlarda murakkab masalalar echila boshlandi. Hozirgi kunda ishlab chiqarish sohalarining xar birida zamonaviy texnologiyalarni qo‘llash, ularning dasturiy

ta'minotini ishlab chiqish eng dolzarb masalalardan hisoblanadi. Deyarli barcha korxonalar, tashkilotlar va boshqarmalarda boshqaruvni avtomatlashtirish eng muhim muammolardan hisoblanib uning negizini albatta dasturlash tashkil qiladi. Ushbu tadqiqot ishining maqsadi algoritm tuzish usullari va dastur ishlab chiqish jarayonlarini batafsil o'rganish, bevosita dastur ishlab chiqishda yuzaga keladigan muammolarni innovatsion texnologiyalar asosida hal qilishni o'rgatishdan iborat.

Python dasturlash tilida ma'lumot to'plamlarini saqlashda bir qancha ma'lumot turlar ishladiladi. Shulardan bir bu ro'yxatlar(list)dir. Listlar – elementlari kvadrat qavslarga olib yoziluvchi ma'lumot tip bo'lib bunda elementlar tiplari har hil yoki bir hil bo'lishi mumkin [1]. Masalan:

```
list1 = [1, -45, 76, -10, -45, 100000, -999954]
list2 = ["salom ", True, -15.6, 200]
```

List elementlari belgilangan tartibda, element qiymatlarini o'zgartirish va yana element qo'shish mumkin bo'ladi (Tuple da o'zgartirish imkoniyati mavjud emas). List elementlar 0,1,2,3… tartibda indekslanadi.

list1 = [1, -45, 76, -10, -45, 100000, -999954]								
List index	0	1	2	3	4	5	6	
list2 = ["salom", True, -15.6, 200]								
List index	0		1	2	3			

List indexlarini aniqlash va **type()** funksiyasini Listga nisbatan ishlatsak:

```
list1 = [1, -45, 76, -10, -45]
list2 = ["salom", True, -15.6, 200]
print(list1)
print(list2)
print(list1[2])
print(list2[0])
print(type(list1))
```

Quyidagi natijaga ega bo'lishimiz mumkin:

```
[1, -45, 76, -10, -45]
['salom', True, -15.6, 200]
76
Salom
<class 'list'>
```

Listlar **for** sikli bilan ishlatish juda samarali. Bu orqali listdagi elementlar sonini toppish mumkin, lekin Python tilida **len** funksiyasi orqali ham listdagi elementlar sonini toppish mumkin [2].

```
list2 = ["salom", True, -15.6, 200]
t = 0
for i in list2:
    print(i)
    t += 1
print('elementlar soni ',t)
print('len funksiyasi orqali ',len(list2))
```

Bu yerda **i** o‘zgaruvchisi har sikl takrorlanganda listni har bir elementiga teng bo‘ladi. **t** o‘zgaruvchisi sikl takrorlanganda bittaga oshadi [3], [4].

```
salom
True
-15.6
200
elementlar soni  4
len funksiyasi orqali  4
```

List metodlari.

append() bu metod listga yangi element qo‘shish uchun ishlatiladi.

count() bu metod listdagi aniq bir elementning sonini toppish mumkin.

extend() bu metod bir listga boshqa bir listni birlashtirish amalini bajaradi

insert() bu metod listga biror elementni aniq bir pozitsiyaga joylashtirish uchun ishlatiladi

pop() bu metod listdagi aniq bir indexdagi elementni o‘chiradi

remove() bu metod listdagi bir elementni o‘chiradi

reverse() bu metod list elementlarini teskarisiga joylashtiradi

sort() bu metod list elementlarini tartiblaydi

list metodlariga doir misollar

```
list1 = [14, 45, -89, 6, 74, 65, 6, 78, 6]
list2 = [15, 48, -9, 11, 54, 25, 61, 703258, 678]
list3 = [-77777,-88888,-99999]
list4 = [-1111,-3333,-4444]
list1.append(10000)
print('append ',list1)
```

```
print('count ',list1.count(6))
list3.extend(list1)
print('extend ',list3)
list4.insert(1,-2222)
print('insert ', list4)
list4.pop(1)
print('pop ', list4)
list4.remove(-4444)
print('remove ', list4)
list3.reverse()
print('reverse ', list3)
list2.sort()
print('sort ', list2)
```

Dastur natijasi:

append [14, 45, -89, 6, 74, 65, 6, 78, 6, **10000**]
count 3
extend [-77777, -88888, -99999, 14, 45, -89, 6, 74, 65, 6, 78, 6, **10000**]
insert [-1111, **-2222**, -3333, -4444]
pop [-1111, ____ -3333, -4444]
remove [-1111, -3333] **-4444 elementi o'chib ketdi**
reverse [10000, 6, 78, 6, 65, 74, 6, -89, 45, 14, -99999, -88888, -77777]
sort [-9, 11, 15, 25, 48, 54, 61, 678, 703258]

Masala: A[1..n] sonlar to'plami berilgan. Shu to'plam elementlarining eng kattasini, eng kichchigini, sonlar yig'indisini toping.

```
a = input()
a = a.split(' ')
maxl = int(a[0])
minl = int(a[0])
suml = 0
for i in a:
    i = int(i)
    if maxl < i:
        maxl = i
    if minl > i:
        minl = i
    suml += i
print('Maximum ',maxl)
```

```
print('Minimum ',minl)
print('Summa ',suml)
```

Dastur natijasi:

```
5454 87878 315218 454
Maximum  315218
Minimum  454
Summa  409004
```

Masalani yechish algoritmi for sikli yordamida listning har bir elementini tekshirib chiqishdan iborat, lekin Python dasturlash tilida **max(), min(), sum()** kabi standart funksiyalar ham mavjud, lekin **input**() funksiyasi **str** tip orqali listni olganligi uchun qo'shimcha **b** listidan foydalanamiz.

```
a = input()
a = a.split(' ')
b = []
for i in a:
    b.append(int(i))
print('Maximum ',max(b))
print('Minimum ',min(b))
print('Summa ',sum(b))
```

Dastur natijasi:

```
454 5454 548
Maximum  5454
Minimum  454
Summa  6456
```

Foydalanilgan adabiyotlar:

1. Дмитрий Мусин. Самоучитель Python. 2015. – 136 с.
2. К.Ю. Поляков, В.М. Гуровиц. Язык Python в школьном курсе информатики – М.: Издательский дом МЭИ, 2011. – 424 с.
3. Марк Лутц. Программирование на Python. 1995.
4. Н.А. Прохоренок. Python.Самое необходимое. – Санкт-Петербург:
5. БХВ-Петербург, 2011, – 416 с.