

Семинар 7

Списки

Списки в языке Python

Список (list) – это изменяемая последовательность данных. Списки как и кортежи могут хранить любые ссылки на объекты, для задания списка используются квадратные скобки. Для преобразования к списку функция `list()`. Список в Python может использоваться аналогично массивам в других языках программирования, но может содержать разнотипные данные.

Списки в языке Python

```
myList = ['a', 10]
myList[-1] = 20
print(myList)
print("В обратном порядке:", myList[::-1])
myList = [1, 2]
print("MyList size:", myList.__sizeof__())
print("MyList elements count:", myList.__len__()) # или функция len()
myTuple = (1, 2)
print("MyTuple size:", myTuple.__sizeof__())
print("MyTuple elements count:", myTuple.__len__()) # или функция len()
```

Вывод:

['a', 20]

В обратном порядке: [20, 'a']

MyList size: **28**

MyList elements count: **2**

MyTuple size: **20**

MyTuple elements count: **2**

Задача 1

Определить индексы элементов списка, значение которых не меньше заданного минимума и не больше заданного максимума. Список заполняется случайными числами в диапазоне от 0 до 99 (включительно) и состоит из 100 элементов.

```
import random # использование псевдослучайных чисел

arr = []
for i in range(100):
    x = int(random.random() * 100) # 0 <= x <= 99
    arr.append(x)
    print("%3d" % x, end=' ') # выравнивание при выводе
    if (i + 1) % 10 == 0: # каждый 10й элемент с новой строки
        print()
minimum = int(input('Min: '))
maximum = int(input('Max: '))
index = []
for i in arr:
    # if minimum <= i <= maximum: # 2 альтернативы, рекомендуется вторая
    if (minimum <= i) and (i <= maximum):
        index.append(arr.index(i))
print("Всего в интервале: ", len(index))
print("Индексы: ", index)
```

Копирование списков

*# Копирование списков - shallow copy - ТОЛЬКО
ССЫЛКИ!*

```
myList1 = [10, 12]
```

```
myList2 = myList1
```

```
myList1[0] = 11
```

```
print(myList2) # что будет выведено?
```

```
myList2 = myList1.copy() # copy
```

```
# deep copy - copy.deepcopy(myList1)
```

```
myList1[0] = 100500
```

```
print(myList2) # что будет выведено?
```

```
myList3 = [myList1, myList2]
```

```
print(myList3)
```

```
myList4 = myList3.copy() #что скопируется?
```

```
myList1[0] = 0; myList2[0] = 0
```

```
print(myList4)
```

```
# но если вот так, то будут новые объекты:
```

```
myList3 = [myList1, myList2]
```

```
print(myList3)
```

```
myList4 = myList3.copy() #что скопируется??
```

```
СПИСКИ? ССЫЛКИ?
```

```
myList1 = [1, 1]; myList2 = [1, 1]
```

```
print(myList4)
```

Задача 2

В списке, состоящем из 20 положительных и отрицательных целых чисел $[-50; 50]$, найти первый, третий и шестой положительные элементы и вычислить их произведение.

На заметку: третий по счету положительный элемент может быть далеко не третьим в списке.

```
import random
```

```
# заполнение исходного списка
```

```
arr = list()
```

```
for i in range(20):
```

```
    arr.append(int(random.random() * 101) - 50)
```

```
print(arr)
```

Задача 2

```
i = 0
j = 1
ind1 = ind3 = ind6 = 0
while i < 20:
    if arr[i] > 0:
        if j == 1:
            ind1 = i
        elif j == 3:
            ind3 = i
        elif j == 6:
            ind6 = i
        break
    j += 1
    i += 1
if ind6 > 0: # если всё нашлось
    mult = arr[ind1] * arr[ind3] * arr[ind6]
    print("Индексы элементов:", ind1, ind3, ind6)
    print("%d * %d * %d = %d" % (arr[ind1], arr[ind3], arr[ind6], mult))
else:
    print("В списке не нашлось 6-ти положительных элементов")
```

Задача 3

*# В матрице (n*n) заменить последний элемент каждой строки на сумму предыдущих элементов той же строки.*

Матрица - список списков (массив массивов)

```
import random
```

```
# размеры матрицы
```

```
n = 4
```

```
matrix = []
```

```
for i in range(n): # формируем исходную матрицу
```

```
    line = []
```

```
    for j in range(n):
```

```
        line.append(int(random.random() * 11) - 5) # от -5 до 5
```

```
    matrix.append(line)
```


Задача 3

```
for line in matrix: # вывод исходной матрицы
    for i in line:
        print('%4d' % i, end='')
    print()
```

```
print()
```

```
for line in matrix: # изменение матрицы
    summa = 0
    i = 0
    while i < n - 1:
        summa += line[i]
        i += 1
    else:
        line[n - 1] = summa
```

```
for line in matrix: # вывод измененной матрицы
    for i in line:
        print('%4d' % i, end='')
    print()
```

Метод split

`str.split(sep[, maxsplit]) -> list`

sep=None - строка-разделитель, может содержать один или несколько символов. Если разделитель не указан, то разделителем считается последовательность пробельных символов.

maxsplit=-1 - максимальное число разбиений. Если не указано, то не ограничено.

Выделение части информации из строки

```
def getSurnameAge(info):  
    parts = info.split(",") # Возвращает список  
    return parts[0].split()[0], parts[1].split()[0]
```

```
personInfo = "Иванов Иван Иванович, 23 года, 8-999-12345678"  
print(getSurnameAge(personInfo))
```

Пример 2. Ввод последовательности чисел

```
numbers = [int(s) for s in input().split()]
```

Функции map и zip

```
map(func, iterator1, [iterator2, ...])
```

func - функция, которую нужно применить к элементам последовательности(тям)

iterator - последовательность (или iterable-объект)

```
words = ["dog", "frog", "cat", "tiger"]  
print(list(map(len, words))) # [3, 4, 3, 5]
```

```
print(list(map(lambda x, y: x * y, [1, 2], [3, 4, 5]))) # [3, 8]
```

```
zip(iterator1, [iterator2, ...])
```

```
a = [1, 2]  
b = [3, 4]  
print(list(zip(a, b))) # [(1, 3), (2, 4)]
```

Метод join

`st.join(sequence)`

`sequence` – iterable-объект строк

```
words = ["cat", "dog", "frog", "tiger"]  
st = ", ".join(words) # "cat, dog, frog, tiger"
```

```
numbers = [1, 2, 3, 4, 5, 3, 1]  
st1 = ' '.join(numbers) # Почему так нельзя?  
st1 = ' '.join(map(str, numbers)) # 1 2 3 4 5 3 1
```

```
st1 = ' '.join(map(str, numbers))
```

```
numbersList = list(map(str, numbers))  
st2 = '- '.join(numbersList) # 1 - 2 - 3 - 4 - 5 - 3 - 1
```

Методы для работы со списками

count(x) - количество элементов в списке

index(x) - поиск первого вхождения элемента в список

index(x, from) - поиск первого вхождения элемента в список, начиная с позиции from

count(x) - количество вхождения элемента

append(x) - добавление элемента в конец списка

extend(list) - добавление другого списка в конец текущего

remove(x) - удаление первого вхождения элемента в список

insert(index, x) - вставка элемента в список в позицию index

reverse() - переворот списка

pop() - возвращает последний элемент списка и удаляет его

pop(index) - возвращает элемент списка в позиции index и удаляет его

copy() - поверхностное копирование списка

clear() - очистка списка

sort() - сортировка списка

in/not in - содержится/не содержится элемент в списке

max/min - максимальный/минимальный элемент списка