Списки в Python

Что такое списки?

Список (list) представляет тип данных, который хранит набор или последовательность элементов. Во многих языках программирования есть аналогичная структура данных, которая называется массив.

```
numbers = [1, 2, 3, 4, 5] # Целые значения
people = ["Tom", "Sam", "Bob"] # Строковые
objects = [1, 2.6, "Hello", True] # Смешанные
```

Функция List()

```
letters = list("Hello")
print(letters)      # ['H', 'e', 'l', 'l', 'o']
```

Повторение значений:

```
numbers = [5] * 6   # 6 раз повторяем 5
people = ["Tom"] * 3    # 3 раза повторяем "Tom"
students = ["Bob", "Sam"] * 2   # 2 раза повторяем "Bob", "Sam"
```

Обращение к элементам списка

```
people = ["Tom", "Sam", "Bob"]
```

Одиночные элементы

```
S = people[0] # Выводит первый элемент “Tom”
S = people[-1] # Выводит последний элемент “Bob”
```

Диапазоны элементов

```
list[:end]: -                   people[:3]   # с 0 по 3
list[start:end]:                people[1:3]   # с 1 по 3
list[start:end:step]:           people[1:6:2]   # с 1 по 6 с шагом 2
```

Разложение списка - Python позволяет разложить список на отдельные элементы:

```
tom, bob, sam = people
```

Методы и функции по работе со списками

1. Добавление элементов списка

добавляем в конец списка
```
people.append("Alice")  # ["Tom", "Bob", "Alice"]
```

добавляем на определенную позицию
```
people.insert(1, "Bill")  # ["Tom", "Bill", "Bob", "Alice"]
```

Добавляем список значений в список
```
people.extend(["Mike", "Sam"])      # ["Tom", "Bill", "Bob",
                                       "Alice", "Mike", "Sam"]
```

2. Проверка наличия и перебор элементов

```
if "Alice" in people: # Проверка наличия элемента

for person in people: # Перебор элементов с for

while i < len(people):
    print(people[i])    # применяем индекс элемента
    i += 1
```

3. Удаление элементов

удаляем по этому индексу
```
people.pop(3)
```

удаляем последний элемент
```
people.pop()
```

удаляем элемент по значению
```
people.remove("Alice")
```

удаляем все элементы
```
people.clear()
```

4. Удаление с помощью del

```
del people[1]   # удаляем второй элемент

del people[:3]   # удаляем  по четвертый элемент не
                                                включая
del people[1:]   # удаляем  со второго элемента
```

Методы и функции по работе со списками

5. Подсчет вхождений count

```
people = ["Tom", "Bob", "Alice", "Tom", "Bill", "Tom"]
people_count = people.count("Tom") # 3
```

6. Сортировка

```
# По возрастанию
people = ["Tom", "Bob", "Alice", "Sam", "Bill"]
people.sort() # ["Alice", "Bill", "Bob", "Sam", "Tom"]

# По убыванию
people = ["Tom", "Bob", "Alice", "Sam", "Bill"]
people.reverse() # ["Tom", "Sam", "Bob", "Bill", "Alice"]
```

функция sorted

sorted(list): сортирует список list
sorted(list, key): сортирует список list, применяя к элементам функцию key

```
my_list = ["apple", "banana", "cherry", "date"]
sor_l= sorted(my_list, key=len) # ['date', 'apple', 'cherry', 'banana']
```

7. Минимальное и максимальное значения min() и max()

```
numbers = [9, 21, 12, 1, 3, 15, 18]
print(min(numbers))    # 1
print(max(numbers))    # 21
```

8. Копирование списков copy():

```
people1 = ["Tom", "Bob", "Alice"]
people2 = people1.copy()
```

9. Соединение списков

```
people1 = ["Tom", "Bob", "Alice"]
people2 = ["Tom", "Sam", "Tim", "Bill"]
people3 = people1 + people2
```

10. Длина списка

```
people1 = ["Tom", "Bob", "Alice"]
l_list = len(people1) # 3
```

Списки в списках (двумерные списки)

Списки кроме стандартных данных типа строк, чисел, также могут содержать другие списки. Подобные списки можно ассоциировать с таблицами, где вложенные списки выполняют роль строк.

Название списка [Индекс списка][Индекс подсписка]

```
people = [
   ["Tom", 29],
   ["Alice", 33],
   ["Bob", 27]
]

people[0]       # ["Tom", 29]
people[0][0]    # Tom
people[0][1]    # 29
```

Пример перебора значений:

```
for person in people:
   for item in person:
      print(item, end=" | ")
```

Добавление и удаление элементов

```
nested_list = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]

new_list = [10, 11, 12]
nested_list.append(new_list)

nested_list[1].append(3)

nested_list[0].remove(2)
```

Самостоятельные задания

1. Напишите программу, которая создает пустой список и заполняет его элементами с 1 до 10 с помощью цикла.
2. Напишите программу, которая находит сумму всех элементов в списке.
3. Напишите программу, которая находит наибольший элемент в списке.
4. Напишите программу, которая находит наименьший элемент в списке.
5. Напишите программу, которая создает новый список, содержащий только четные элементы из исходного списка.
6. Напишите программу, которая удаляет все дубликаты из списка.
7. Напишите программу, которая проверяет, содержит ли список заданный элемент.
8. Напишите программу, которая инвертирует порядок элементов в списке (переворачивает его).
9. Напишите программу, которая объединяет два списка в один новый список.
10. Напишите программу, которая находит среднее арифметическое всех элементов в списке.

Самостоятельные задания

1. Дан список размера 5 x 10. Удалить столбец, содержащий максимальный элемент матрицы.
2. Дана целочисленная квадратный список порядка N. Получить новую матрицу, исключив из исходного списка столбцы, состоящие из нулей.
3. Дан список размера 5 x 10. В каждой строке найти количество элементов, больших среднего арифметического всех элементов этой строки.
4. Дан двумерный список R(4,3). Увеличьте значение каждого элемента в 5 раз, а затем найдите сумму значений элементов первой строки. Выведите на экран новый список в виде таблицы.