**Списки**

**Cписки** (в большинстве же языков программирования используется другой термин – "массив") — это такая структура данных, которая позволяет хранить в себе несколько однотипных значений, таких как числа или строки. Например, если мы хотим проверить число на простоту, а потом построить его разложение на простые множители, нам удобно создать список простых чисел. Тогда мы сможем проверять делимость числа на все ранее построенные элементы списка.

Список в Python представляет собой последовательность элементов, пронумерованных от 00, как символы в строке. Тогда список, состоящий из nn элементов, будет заканчиваться элементом с индексом n−1n−1.

**Пример:**

Список можно задать перечислением элементов списка в квадратных скобках, например, список из первых 66 простых чисел можно задать так:

p = [2, 3, 5, 7, 11, 13]

В списке p— 66 элементов, а именно, p[0] == 2, p[1] == 3, p[2] == 5, p[3] == 7, p[4] == 11, p[5] == 13.

Также как и символы строки, элементы списка в Python можно индексировать отрицательными числами с конца, например, p[-1] == 13. Длину списка, то есть количество элементов в нём, можно узнать при помощи функции len(), например, len(p) == 6.

Если мы попробуем обраться к элементу списка по некорректному индексу, например, обратимся к элементу p[6], то произойдёт ошибка **IndexError: list index out of range**.

**Пример:**

Пусть теперь мы хотим перебрать все элементы списка, при этом выводя их на экран, а потом возводя в квадрат. Организуем цикл следующим образом:

**for** i **in** range(len(p)):

**print**(p[i])

p[i] = p[i] \*\* 2

Таким образом, при использовании списков можно присваивать элементам новые значения, в отличие от работы со строками.

**Пример:**

Пусть нам необходимо хранить базу данных учащихся некоторой школы, содержащую данные об учащихся и их оценках для ведения электронного дневника. Заведем списки для хранения имен учащихся names и для хранения их оценок scores, причём элементы с одинаковыми индексами будут соответствовать один и тем же ученикам. Выведем данные, пронумеровав записи, с помощью следующей программы:

names = ['Иванов', 'Петров', 'Королев']

scores = [5, 4, 5]

**for** i **in** range(len(scores)):

**print**(i+1, '.', sep = '', end = ' ')

**print**(names[i], scores[i])

На выходе получим:

1. Иванов 5

2. Петров 4

3. Королев 5

**Операции со списками**

Для списков целиком определены следующие операции: конкатенация списков (добавление одного списка в конец другого) и повторение списков (умножение списка на число). Например:

a = [1, 2, 3]

b = [4, 5]

c = a + b

d = a \* 10

В результате приведенных выше операций список c будет равен [1, 2, 3, 4, 5], а список d будет состоять из 30 элементов, из элементов 1, 2, 3, повторенных 10 раз.

**Создание списков**

Операция повторения списка позволяет конструировать большие списки. Например, списки с 1000 элементов, заполненных нулями.

a = [0] \* 1000

Рассмотрим другой способ создания списков — при помощи метода append, позволяющего помещать элементы в конец списка. Прежде всего можно создать пустой список (не содержащий элементов, имеющий длину 0), в конец списка можно последовательно добавлять элементы.

a = []

**for** i **in** range(1000):

a.append(0)

Ноль можно заменить любым другим выражением.

a = []

**for** i **in** range(1000):

a.append(i)

В полученном списке будет 1000 элементов — от 0 до 999.