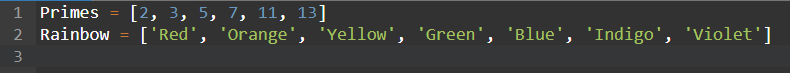
**Списки (массивы)**

Раньше мы сталкивались с задачей обработки элементов последовательности, например, вычисляя наибольший элемент последовательности. Но при этом мы не сохраняли всю последовательность в памяти компьютера. Однако, во многих задачах нужно именно сохранять всю последовательность, например, если бы нам требовалось вывести все элементы последовательности в возрастающем порядке (“отсортировать последовательность”).

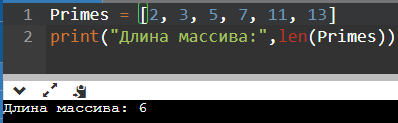
Для хранения таких данных можно использовать структуру данных, называемую в Питоне список (в большинстве же языков программирования используется другой термин “массив”). Список представляет собой последовательность элементов, пронумерованных от 0, как символы в строке.

Например, список можно задать так:

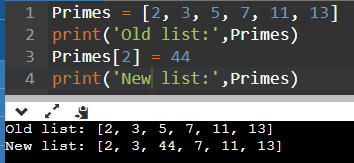


В списке **Primes — 6** элементов, а именно: **Primes[0] == 2, Primes[1] == 3, Primes[2] == 5, Primes[3] == 7, Primes[4] == 11, Primes[5] == 13**. Список **Rainbow** состоит из **7** элементов, каждый из которых является строкой.

Длину списка, то есть количество элементов в нем, можно узнать при помощи функции **len**, например, **len(Primes)** == 6.



Элементы массива, в отличие от строк, можно изменять, присваивая им новые значения. Изменение конкретного значения происходит через обращение к немо по его номеру:



**Способы создания и считывания списков**

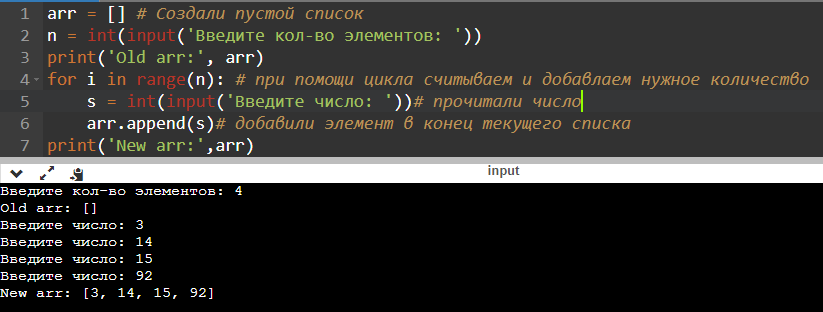
**Способ 0**

Самый простой способ мы свами рассмотрели выше, когда элементы массива перечисляются при его объявлении в квадратных скобках через запятую

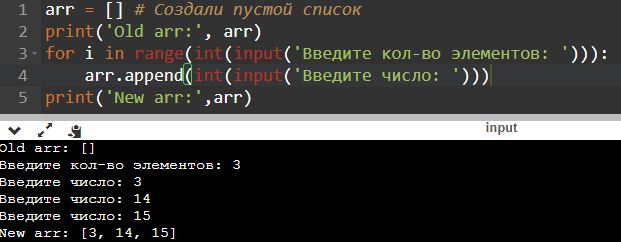
**Способ 1**

Можем создать пустой список длины 0, а затем добавлять в него новые эле менты при помощи метода **append**

Например, пусть нам необходимо заполнить массив n элементами, введенными с клавиатуры:

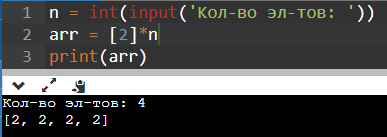


Или это можно сделать так (сокращенный вариант)



**Способ 2. Генераторы списков**

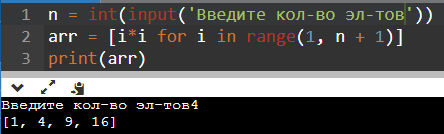
Для создания списка из одинаковых элементов можно использовать оператор повторения списка, например



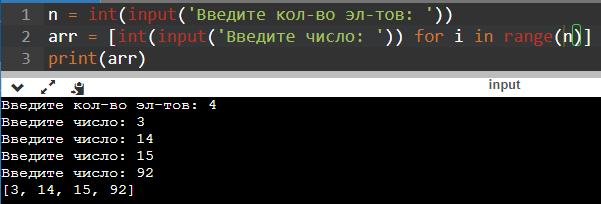
Для создания более сложных списков по каким-либо формулам, можно использовать генераторы: выражения, позволяющие заполнить список некоторой формулой. Общий вид генератора следующий:



где **переменная** — идентификатор некоторой переменной, **последовательность** — последовательность значений, который принимает данная переменная (это может быть список, строка или объект, полученный при помощи функции range), **выражение** — некоторое выражение, как правило, зависящее от использованной в генераторе переменной, которым будут заполнены элементы списка. Например, если мs хотим заполнить список квадратами целых чисел в интервале **от 1 до n**

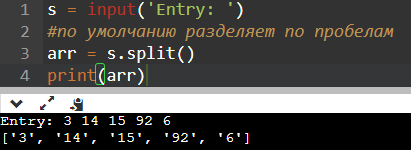


А в этом примере список будет состоять из строк, считанных со стандартного ввода: сначала нужно ввести число элементов списка потом — заданное количество строк.

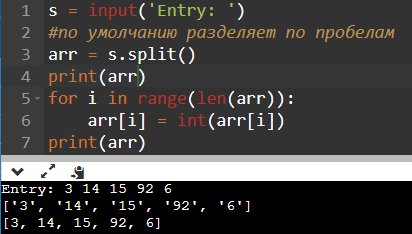


**Способ 3. Метод split**

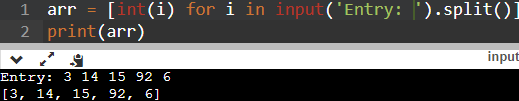
Элементы списка могут вводиться в одну строку, разделенную, например, пробелами. Для получения массива из введенной строки можно воспользоваться методом split(), который возвращает список строк, которые получатся, если разделить исходную строку на части по указанному символу (по умолчанию разделяет по пробелам)



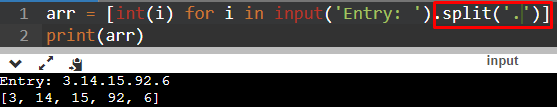
Обратите внимание, что список будет состоять из строк, а не из чисел. Если хочется получить список именно из чисел, то можно затем элементы списка по одному преобразовать в числа:



Используя генераторы, приведенный выше код можно записать в одну строку:



У метода **split()** есть необязательный параметр, который определяет, какая строка будет использоваться в качестве разделителя между элементами списка. Например, вызов метода **split('.')** вернет список, полученный разрезанием исходной строки по символам **'.'**



**Срезы**

Со списками, так же как и со строками, можно делать срезы. А именно:

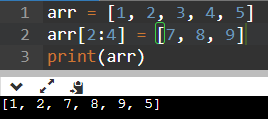
**A[i:j]** срез из **j-i** элементов **A[i], A[i+1], ..., A[j-1].**

**A[i:j:-1]** срез из **i-j** элементов **A[i], A[i-1], ..., A[j+1]** (то есть **меняется порядок** элементов).

**A[i:j:k]** срез с шагом **k: A[i], A[i+k], A[i+2\*k]**,... . Если значение **k<0**, то элементы идут в противоположном порядке.

Каждое из чисел **i** или **j** может отсутствовать, что означает “начало строки” или “конец строки”

Списки, в отличии от строк, являются изменяемыми объектами: можно отдельному элементу списка присвоить новое значение. Но можно менять и целиком срезы. Например:



Операции со списками

* **x in A** Проверить, содержится ли элемент в списке. Возвращает True или False
* **x not in A** То же самое, что **not(x in A)**
* **min(A)** Наименьший элемент списка
* **max(A)** Наибольший элемент списка
* **A.index(x)** Индекс первого вхождения элемента x в список, при его отсутствии генерирует исключение
* **A.count(x)** Количество вхождений элемента x в список

**Задачи**

**ПРИМЕЧАНИЕ:** количество элементов массива, сами элементы массива вводятся с КЛАВИАТУРЫ.

**№1**

Выведите все элементы списка с четными индексами (то есть **A[0], A[2], A[4]**, ...).

**Примечание**: особенно одаренные могут попытаться решить эту задачу в одну строку

**№2**

Выведите все четные элементы списка

**№3**

Дан список чисел. Определите, сколько в этом списке элементов, которые больше двух своих соседей, и выведите количество таких элементов. Крайние элементы списка никогда не учитываются, поскольку у них недостаточно соседей.

**№4**

Дан целочисленный массив **A[n],** среди элементов есть одинаковые. Создать **НОВЫЙ** массив из различных элементов **A[n].**

**№5**

Дан массив, состоящий из n натуральных чисел. Образовать **НОВЫЙ** массив, элементами которого будут элементы исходного массива, оканчивающиеся на цифру **k**, введенную с клавиатуры.

**№6**

Дан массив целых чисел. Найти в этом массиве минимальный элемент **m** и максимальный элемент **M**. Получить все целые числа из интервала (**m; M**) которые не входят в данный массив.

**№7**

В массиве с четным количеством элементов (**2N**) находятся координаты **N** точек плоскости. Они располагаются в следующем порядке:.Определить минимальный радиус круга с центром в начале координат, который содержит все точки.