

Списки

План урока

- Массивы в программировании
- Динамические массивы
- Списки в Python
- Методы работы со списками
- Примеры

МАССИВ

Это коллекция элементов **одного** типа,
расположенных в памяти **последовательно**
и имеющих общее имя

Каждый элемент в массиве имеет
уникальный номер (индекс)

МАССИВ

Это коллекция элементов **одного** типа,
расположенных в памяти **последовательно**
и имеющих общее имя

i →	0	1	2	3	4	5	6	7	8	9
A[i]	-5	-2	-6	-1	0	4	2	3	-1	-3

A - имя массива

i - индекс элемента

Поиск адреса переменной

ооо	массив
[10, 20, 30]	

Допустим, один элемент занимает в памяти

Предположим, что адрес первого элемента в массиве [10] = 1000

Тогда адрес третьего элемента в массиве [30] с индексом [2], будет равен:

$$1000 + 4 \times 2 = 1008$$

ДИНАМИЧЕСКИЕ МАССИВЫ

Списки в Python – это **динамические массивы**

Динамические массивы – это массивы,
размер которых может **изменяться**
во время выполнения программы

Почему в Python списки называются списками?

СПИСОК

это упорядоченный набор элементов, каждый из которых имеет свой номер (индекс), относящийся к абстрактным структурам данных

Абстрактные структуры данных **не имеют** строго определенного представления в **памяти** и определяются набором свойств и методов

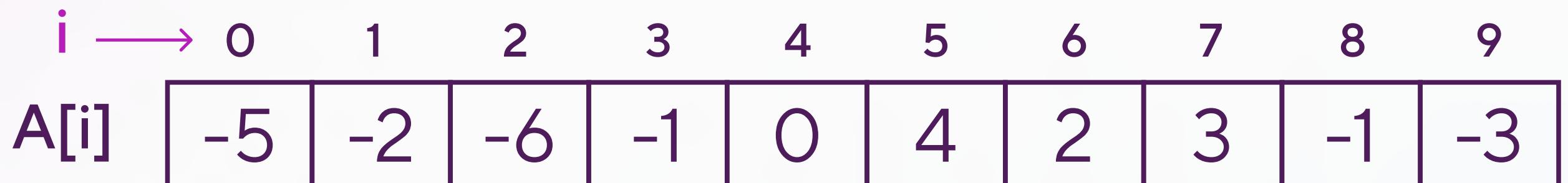
Атрибуты списка

- Размер (длина)
- Методы чтения
- Методы записи
- Добавление
- Удаление

СПИСКИ В РУТНОН

Динамический массив

- это конкретная структура данных



A - имя массива

i - индекс элемента

Списки в Python реализуют все свойства и методы абстрактной структуры данных «**список**», но реализация построена на **динамических массивах**

Длина списка

ооо

код

```
numbers = [3, 8, 11, -4, 9, 2]  
          ↑  
print(len(numbers))
```

ооо

вывод

```
>>> 6
```

Создаем массив
целых чисел

Длина списка

ооо

код

```
numbers = [3, 8, 11, -4, 9, 2]  
print(len(numbers))
```

ооо

вывод

```
>>> 6
```

Вызываем метод **len**,
который возвращает длину

Обращение к элементу массива

ооо

код

```
numbers = [3, 8, 11, -4, 9, 2]
```

```
print(numbers[0])
```

```
print(numbers[1])
```

ооо

вывод

```
>>> 3
```

```
>>> 8
```

Обращаемся к элементам
по индексу с помощью []

Обращение к элементу массива

ooo | код

```
numbers = [3, 8, 11, -4, 9, 2]
```

```
print(numbers[0]) ←
```

```
print(numbers[1]) ←
```

ooo | вывод

```
>>> 3
```

```
>>> 8
```

Индексация начинается с 0

Обращение к элементу массива

ооо

код

```
numbers = [3, 8, 11, -4, 9, 2]
```

```
print(numbers[-1])
```

```
print(numbers[-2])
```

ооо

вывод

```
>>> 2
```

```
>>> 9
```

Обращаемся к элементам
с помощью отрицательных
индексов

Обращение к элементу массива

ооо

код

```
numbers = [3, 8, 11, -4, 9, 2]  
print(numbers[6])
```

ооо

вывод

```
>>> IndexError
```

Обращаемся к элементу
с несуществующим индексом

Пример

ooo

код

```
coins = [1, 2, 5]
it = iter(coins)
print(next(it))
print(next(it))
print(next(it))
```

ooo

вывод

```
>>> 1
>>> 2
>>> 5
```

Пример

ooo

код

```
numbers = [3, 8, 100, -4, 9, 2]  
  
for number in numbers:  
    print(number)
```

ooo

вывод

```
>>> 3  
>>> 8  
>>> 100  
>>> -4  
>>> 9  
>>> 2
```

Пример

ooo

код

```
numbers = [3, 8, 100, -4, 9, 2]

for i in range(len(numbers)):
    numbers[i] += 1
print(numbers)
```

ooo

вывод

```
>>> [4, 9, 101, -3, 10, 3]
```

Пример

ооо | код

```
list1 = [1, 2, 3] ←  
list2 = [1, 2, 3] ←  
  
print(id(list1), id(list2))
```

Создаем два списка:
list1 и list2

Пример

ooo | код

```
list1 = [1, 2, 3]
list2 = [1, 2, 3]

print(id(list1), id(list2))
```

ooo | вывод

```
>>> 2397818524480 2397818525376
```

Разные id

Добавление элемента

ооо	код
<pre>list1 = [1, 2, 3] list1.append(4) ← print(list1)</pre>	

Добавляем
элемент 4

ооо	вывод
<pre>>>> [1, 2, 3, 4]</pre>	

Удаление элемента

ооо

код

```
list1 = [1, 2, 3, 4]  
list1.pop(1)  
print(list1)
```

Удаление
элемента
с индексом 1

ооо

вывод

```
>>> [1, 2, 4]
```

Объединение списков

ооо | код

```
list1 = [1, 3, 4]  
list2 = [1, 2, 3]
```

```
list1.extend(list2)  
print(list1)
```

Объединение
списков

ооо | вывод

```
>>> [1, 3, 4, 1, 2, 3]
```

Оператор in

ооо

код

```
numbers = [2, 7, 9]  
  
print(2 in numbers)  
print(3 in numbers)
```

ооо

вывод

```
>>> True  
>>> False
```

Срезы

ооо

```
some_list[start:end:step]
```

ооо

```
some_list[start:end]
```

Значения параметров по умолчанию
такие же как и в случае с **range**

Также добавляются значения параметров
по умолчанию для **end** и для **start**

python.org