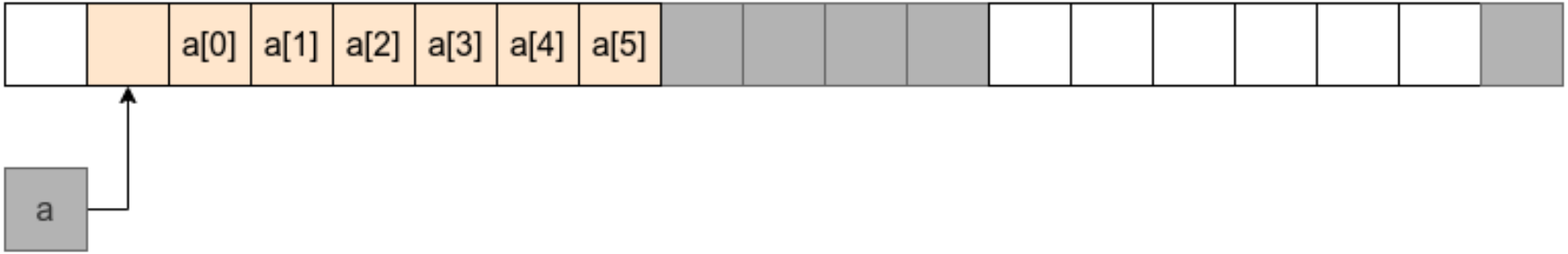
Массивы
(класс array)

Массив — это структура данных, предназначенная для хранения множества элементов одного типа.

Все элементы идентифицируются по индексу.

Гарантируется, что все элементы в оперативной памяти располагаются последовательно без пропусков.



В стандартной библиотеке Python массив представлен классом ***`array`*** в модуле ***`array`***.

https://docs.python.org/3/library/array.html#module-array

Конструктор класса array

```
array(typecode[, initializer])
```

Type code	C Type	Python Type	Minimum size in bytes
`'b'`	signed char	int	1
`'B'`	unsigned char	int	1
~~`'u'`~~	~~wchar_t~~	~~Unicode character~~	~~2~~ устарело
`'h'`	signed short	int	2
`'H'`	unsigned short	int	2
`'i'`	signed int	int	2
`'I'`	unsigned int	int	2
`'l'`	signed long	int	4
`'L'`	unsigned long	int	4
`'q'`	signed long long	int	8
`'Q'`	unsigned long long	int	8
`'f'`	float	float	4
`'d'`	double	float	8

```
import array

foo = array.array("i")
bar = array.array("i", [10, 20, 30, 40])

print("type(foo):", type(foo))
print("foo:", foo)
print()

print("type(bar):", type(bar))
print("bar:", bar)
```

```
type(foo): <class 'array.array'>
foo: array('i')

type(bar): <class 'array.array'>
bar: array('i', [10, 20, 30, 40])
```

```
from array import array

foo = array("i")
bar = array("i", [10, 20, 30, 40])

print("type(foo):", type(foo))
print("foo:", foo)
print()

print("type(bar):", type(bar))
print("bar:", bar)
```

```
type(foo): <class 'array.array'>
foo: array('i')

type(bar): <class 'array.array'>
bar: array('i', [10, 20, 30, 40])
```

```
from array import array

print(dir(array))
```

['__add__', '__class__', '__contains__', '__copy__', '__deepcopy__', '__delattr__', '__delitem__', '__dir__', '__doc__', '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', '__getstate__', '__gt__', '__hash__', '__iadd__', '__imul__', '__init__', '__init_subclass__', '__iter__', '__le__', '__len__', '__lt__', '__module__', '__mul__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__rmul__', '__setattr__', '__setitem__', '__sizeof__', '__str__', '__subclasshook__', 'append', 'buffer_info', 'byteswap', 'count', 'extend', 'frombytes', 'fromfile', 'fromlist', 'fromunicode', 'index', 'insert', 'itemsize', 'pop', 'remove', 'reverse', 'tobytes', 'tofile', 'tolist', 'tounicode', 'typecode']

Метод array.extend()

https://docs.python.org/3/library/array.html#array.array.extend

extend(*iterable*)

Append items from *iterable* to the end of the array. If *iterable* is another array, it must have *exactly* the same type code; if not, `TypeError` will be raised. If *iterable* is not an array, it must be iterable and its elements must be the right type to be appended to the array.

```
from array import array

foo = array("i", [1, 2, 3])
bar = array("i", [10, 20, 30, 40])

foo.extend(bar)

print (foo)
```

```
array('i', [1, 2, 3, 10, 20, 30, 40])
```

```
from array import array

foo = array("i", [1, 2, 3])
bar = [10, 20, 30, 40]

foo.extend(bar)

print (foo)
```

```
array('i', [1, 2, 3, 10, 20, 30, 40])
```

```
from array import array

foo = array("i", [1, 2, 3])
bar = [10.0, 20.0, 30.0, 40.0]

foo.extend(bar)
```

```
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'float' object cannot be interpreted as an integer
```

Сравнение времени добавления элементов в конец списка и массива (с использованием JupyterLab)


```
from array import array

foo = []
bar = array("i")

%timeit foo.append(100)
%timeit bar.append(100)
```

```
19 ns ± 0.772 ns per loop (mean ± std. dev. of 7 runs, 10,000,000 loops each)
71.8 ns ± 1.95 ns per loop (mean ± std. dev. of 7 runs, 10,000,000 loops each)
```