**Модуль array. Массивы в python**

Модуль array определяет массивы в python. Массивы очень похожи на [списки](https://pythonworld.ru/tipy-dannyx-v-python/spiski-list-funkcii-i-metody-spiskov.html), но с ограничением на тип данных и размер каждого элемента.

Размер и тип элемента в массиве определяется при его создании и может принимать следующие значения:

| **Код типа** | **Тип в C** | **Тип в python** | **Минимальный размер в байтах** |
| --- | --- | --- | --- |
| 'b' | signed char | int | 1 |
| 'B' | unsigned char | int | 1 |
| '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 |

Класс **array.array**(TypeCode [, инициализатор]) - новый массив, элементы которого ограничены TypeCode, и инициализатор, который должен быть списком, объектом, поддерживающий интерфейс буфера, или итерируемый объект.

**array.typecodes** - строка, содержащая все возможные типы в массиве.

**Массивы изменяемы**. Массивы поддерживают все списковые методы (индексация, срезы, умножения, итерации), и другие методы.

**Методы массивов (array) в python**

**array.typecode** - TypeCode символ, использованный при создании массива.

**array.itemsize** - размер в байтах одного элемента в массиве.

**array.append**(х) - добавление элемента в конец массива.

**array.buffer\_info**() - кортеж (ячейка памяти, длина). Полезно для низкоуровневых операций.

**array.byteswap**() - изменить порядок следования байтов в каждом элементе массива. Полезно при чтении данных из файла, написанного на машине с другим порядком байтов.

**array.count**(х) - возвращает количество вхождений х в массив.

**array.extend**(iter) - добавление элементов из объекта в массив.

**array.frombytes**(b) - делает массив array из массива байт. Количество байт должно быть кратно размеру одного элемента в массиве.

**array.fromfile**(F, N) - читает N элементов из [файла](https://pythonworld.ru/tipy-dannyx-v-python/fajly-rabota-s-fajlami.html) и добавляет их в конец массива. Файл должен быть открыт на бинарное чтение. Если доступно меньше N элементов, генерируется [исключение](https://pythonworld.ru/tipy-dannyx-v-python/isklyucheniya-v-python-konstrukciya-try-except-dlya-obrabotki-isklyuchenij.html) EOFError , но элементы, которые были доступны, добавляются в массив.

**array.fromlist**(список) - добавление элементов из списка.

**array.index**(х) - номер первого вхождения x в массив.

**array.insert**(n, х) - включить новый пункт со значением х в массиве перед номером n. Отрицательные значения рассматриваются относительно конца массива.

**array.pop**(i) - удаляет i-ый элемент из массива и возвращает его. По умолчанию удаляется последний элемент.

**array.remove**(х) - удалить первое вхождение х из массива.

**array.reverse**() - обратный порядок элементов в массиве.

**array.tobytes**() - преобразование к байтам.

**array.tofile**(f) - запись массива в открытый файл.

**array.tolist**() - преобразование массива в список.

Вот и всё, что можно было рассказать про массивы. Они используются редко, когда нужно достичь высокой скорости работы. В остальных случаях массивы можно заменить другими типами данных: [списками](https://pythonworld.ru/tipy-dannyx-v-python/spiski-list-funkcii-i-metody-spiskov.html), [кортежами](https://pythonworld.ru/tipy-dannyx-v-python/kortezhi-tuple.html), [строками](https://pythonworld.ru/tipy-dannyx-v-python/stroki-funkcii-i-metody-strok.html).