**Методы и функции по работе со списками и кортежами**

Для управления элементами списки имеют целый ряд методов. Некоторые из них:

* **append(item)**: добавляет элемент item в конец списка
* **insert(index, item)**: добавляет элемент item в список по индексу index
* **extend(items)**: добавляет набор элементов items в конец списка
* **remove(item)**: удаляет элемент item. Удаляется только первое вхождение элемента. Если элемент не найден, генерирует исключение ValueError
* **clear()**: удаление всех элементов из списка
* **index(item)**: возвращает индекс элемента item. Если элемент не найден, генерирует исключение ValueError
* **pop([index])**: удаляет и возвращает элемент по индексу index. Если индекс не передан, то просто удаляет последний элемент.
* **count(item)**: возвращает количество вхождений элемента item в список
* **sort([key])**: сортирует элементы. По умолчанию сортирует по возрастанию. Но с помощью параметра key мы можем передать функцию сортировки.
* **reverse()**: расставляет все элементы в списке в обратном порядке
* **copy()**: копирует список

Кроме того, Python предоставляет ряд встроенных функций для работы со списками:

* **len(list)**: возвращает длину списка
* **sorted(list, [key])**: возвращает отсортированный список
* **min(list)**: возвращает наименьший элемент списка
* **max(list)**: возвращает наибольший элемент списка
* **del variable** удаление переменной **del obj.attr** удаление атрибута **del data[k]** удаление элемента по индексу **del data[i:j]** удаление элементов по срезу

**С множествами можно выполнять множество операций: находить объединение, пересечение...**

* len(s) - число элементов в множестве (размер множества).
* x in s - принадлежит ли x множеству s.
* **set.isdisjoint**(other) - истина, если set и other не имеют общих элементов.
* **set == other** - все элементы set принадлежат other, все элементы other принадлежат set.
* **set.issubset**(other) или **set <= other** - все элементы set принадлежат other.
* **set.issuperset**(other) или **set >= other** - аналогично.
* **set.union**(other, ...) или **set | other | ...** - объединение нескольких множеств.
* **set.intersection**(other, ...) или **set & other & ...** - пересечение.
* **set.difference**(other, ...) или **set - other - ...** - множество из всех элементов set, не принадлежащие ни одному из other.
* **set.symmetric\_difference**(other); **set ^ other** - множество из элементов, встречающихся в одном множестве, но не встречающиеся в обоих.
* **set.copy**() - копия множества.

И операции, непосредственно изменяющие множество:

* **set.update**(other, ...); set |= other | ... - объединение.
* **set.intersection\_update**(other, ...); set &= other & ... - пересечение.
* **set.difference\_update**(other, ...); set -= other | ... - вычитание.
* **set.symmetric\_difference\_update**(other); set ^= other - множество из элементов, встречающихся в одном множестве, но не встречающиеся в обоих.
* **set.add**(elem) - добавляет элемент в множество.
* **set.remove**(elem) - удаляет элемент из множества. KeyError, если такого элемента не существует.
* **set.discard**(elem) - удаляет элемент, если он находится в множестве.
* **set.pop**() - удаляет первый элемент из множества. Так как множества не упорядочены, нельзя точно сказать, какой элемент будет первым.
* **set.clear**() - очистка множества

### Экранированные последовательности - служебные символы

Экранированные последовательности позволяют вставить символы, которые сложно ввести с клавиатуры

| **Экранированная последовательность** | **Назначение** |
| --- | --- |
| \n | Перевод строки |
| \a | Звонок |
| \b | Забой |
| \f | Перевод страницы |
| \r | Возврат каретки |
| \t | Горизонтальная табуляция |
| \v | Вертикальная табуляция |
| \N{id} | Идентификатор ID базы данных Юникода |
| \uhhhh | 16-битовый символ Юникода в 16-ричном представлении |
| \Uhhhh… | 32-битовый символ Юникода в 32-ричном представлении |
| \xhh | 16-ричное значение символа |
| \ooo | 8-ричное значение символа |
| \0 | Символ Null (не является признаком конца строки) |