Функции и методы списков

Примеры списков:

- `family = ['Мама', 'Папа', 'Я', 'Кот']`
- `luckyNumbers = [2, 4, 15]`

Методы для списков:

- `.append()` – метод для добавления элементов в список
- `extend()` – добавляет несколько элементов в конец списка
- `insert()` – добавляет один элемент в произвольное место списка, необязательно в конец.
- `luckyNumbers [x]` – обращение к элементу списка номер X
- `luckyNumbers[x:y]` – срез списка с элемента X до элемента Y-1
- `remove()` – удаляет из списка выбранный вами элемент.
- `del()` – удаляет элемент по его индексу
- `pop()` – извлекает из списка последний элемент и возвращает его вам.
- `if 'a' in letters:` – часть логического выражения. Возвращает значение «True», если в списке присутствует буква a, и значение «False» в противном случае.
- `index()` – местоположение элемента в списке определяет метод

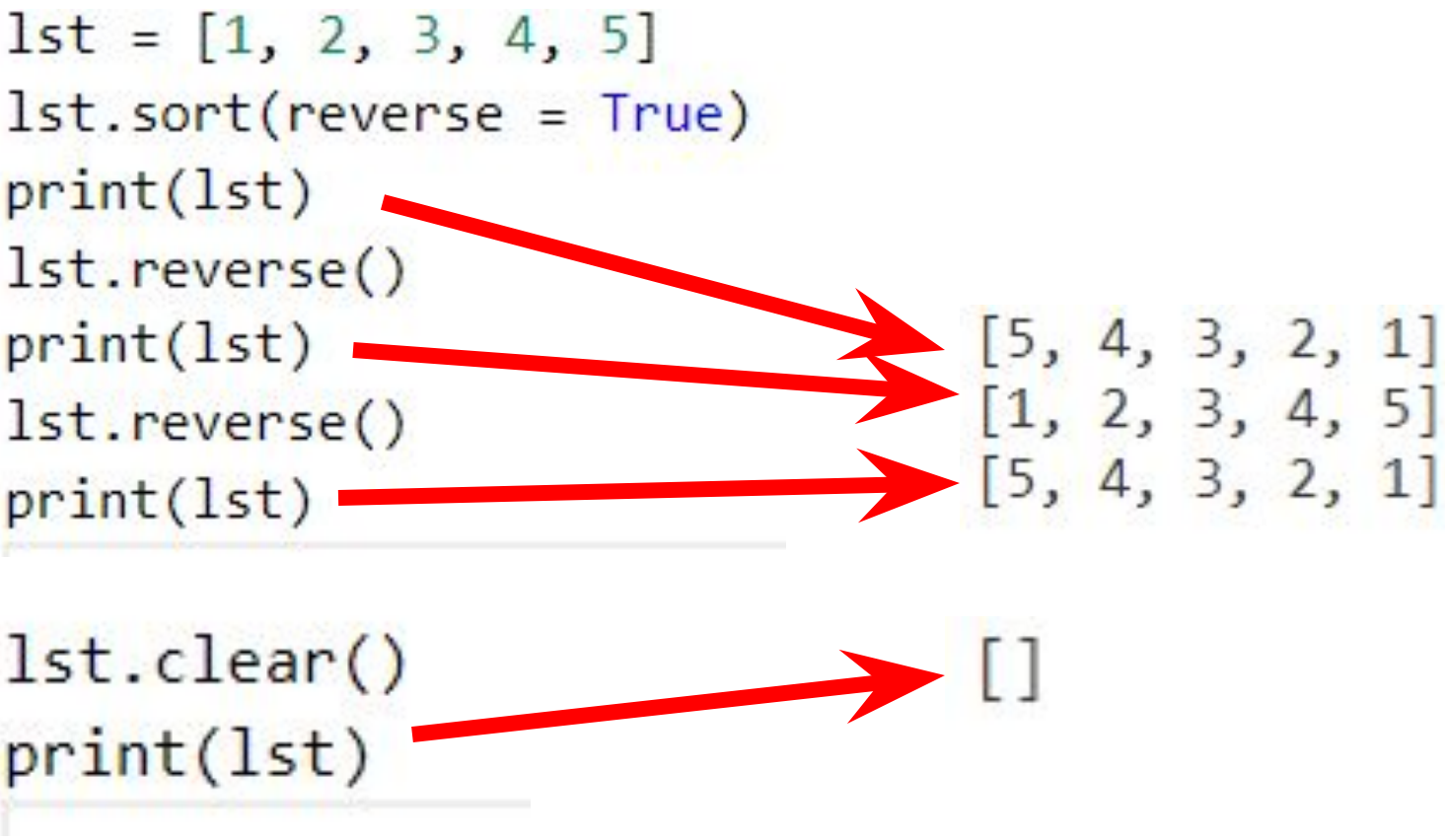
1. Метод **sort()** автоматически располагает строки в алфавитном порядке, а числа упорядочивает по возрастанию.

```
letters = ["d", "a", "e", "c", "b"]
print(letters)
letters.sort()
print(letters)
```

2. Метод **sort()** сам редактирует список. То есть он вносит изменения в оригинал, а не создает для вас новую отсортированную копию.

5. Существует еще один способ получения отсортированной копии без нарушения порядка размещения элементов в исходном списке. В Python для этой цели используется функция **sorted()**, которая дает вам отсортированную копию исходного списка.

```
original = [5, 2, 3, 1, 4]
newer = sorted(original)
print(original)
print(newer)
```


lst = [1, 2, 3, 4, 5]
lst.sort(reverse = True)
print(lst)
lst.reverse()
print(lst)
lst.reverse()
print(lst)
[5, 4, 3, 2, 1]
[1, 2, 3, 4, 5]
[5, 4, 3, 2, 1]
lst.clear()
print(lst)
[]

Домашнее задание

Ваша программа получает имена учеников класса, пока не получит пустую строку. Нужно посчитать, сколько одинаковых имен в классе, и вывести это имя и число его повторений.