Python: Цикл For

С помощью цикла `while` решают любую задачу перебора элементов, но его отличает многословность. Для `while` нужно задавать условие остановки и вводить счетчик. Когда циклов немного, то это нормально, но в реальном коде циклы встречаются на каждом шагу. Поэтому управлять условиями вручную утомительно, особенно когда условие остановки очевидно.

Например, если мы хотим перебрать символы в строке, то компьютер сам может понять, когда строка заканчивается. Для таких ситуаций в Python ввели цикл `for`. Он сам знает, когда нужно остановиться, так как работает только с коллекциями — наборами элементов, которые нужно перебрать.

Строка — это коллекция, так как состоит из набора символов. Остальные виды коллекций подробно изучаются в другом курсе.

Пример:

```
text = 'code'
for symbol in text:
    print(symbol)

# => c
# => o
```

```
# => d
# => e
```

В коде выше `for` проходит по каждому символу в строке, записывает его в переменную `symbol` и вызывает внутренний блок кода, где эта переменная используется. Имя этой переменной может быть любым. Общая структура цикла `for` выглядит так: `for <переменная> in <коллекция>`.

Посмотрим, как реализовать функцию переворота строки через цикл `for`:

```
def reverse_string(text):
    # Начальное значение
    result = ''
    # char - переменная, в которую записывается текущий сим
    for char in text:
        # Соединяем в обратном порядке
        result = char + result
    # Цикл заканчивается, когда пройдена вся строка
    return result


reverse_string('go!')  # => '!og'
```

Теперь посчитаем количество упоминаний символа в строке без учета регистра:

```
# text - произвольный текст
```

```
# char - символ, который нужно учитывать
def chars_count(text, char):
    # Так как ищем сумму, то начальное значение 0
    result = 0
    for current_char in text:
        # приводим все к нижнему регистру,
        # чтобы не зависеть от текущего регистра
        if current_char.lower() == char.lower():
            result += 1
    return result


chars_count('hexlet!', 'e')  # 2
chars_count('hExlet!', 'e')  # 2
chars_count('hExlet!', 'E')  # 2

chars_count('hexlet!', 'a')  # 0
```

Советуем поэкспериментировать с примерами выше в интерактивном Replit.

Задание

В одном из предыдущих уроков мы уже написали функцию `filter_string()`. Напомним, она принимает на вход строку и символ и возвращает новую строку, в которой удалён переданный символ во всех его позициях. На этот раз реализуйте эту функцию с помощью цикла `for`. Дополнительное условие: регистр исключаемого символа не имеет значения.

Пример вызова:

```
text = 'If I look forward I win'
filter_string(text, 'i')  # 'f  look forward  wn'
filter_string(text, 'O')  # 'If I lk frward I win'
```

- ▶ Упражнение не проходит проверку — что делать? 😶
- ▶ В моей среде код работает, а здесь нет 😐
- ▶ Мой код отличается от решения учителя 🤔
- ▶ Прочитал урок — ничего не понятно 🙄

Определения

- Агрегация — Накопление результата во время итераций и работа с ним после цикла.