Python: Необязательные параметры функций

В программировании у многих функций и методов есть параметры, которые редко меняются. В таких случаях этим параметрам задают значения по умолчанию, которые можно поменять по необходимости. С помощью этого сокращается количество одинакового кода. Рассмотрим, как это выглядит на практике.

Разберем пример:

```
# Функция возведения в степень
# Второй параметр имеет значение по умолчанию два
def pow(x, base=2):
    return x ** base

# Три во второй степени (двойка задана по умолчанию)
pow(3)  # 3 * 3 = 9
# Три в третьей степени
pow(3, 3)  # 3 * 3 * 3 = 27
```

https://replit.com/@hexlet/python-basics-define-functions-default-parameters

Значение по умолчанию выглядит как обычное присваивание в определении. Оно срабатывает только в том случае, если параметр не передали.

Представьте, что вы не привезли с собой в автосервис запчасти для вашего автомобиля. Тогда автомеханик предложит вам поставить те, которые есть у него — по умолчанию.

Значение по умолчанию может быть даже в том случае, когда параметр один:

```
def my_print(text='nothing'):
    print(text)

my_print()  # => "nothing"
my_print("Hexlet")  # => "Hexlet"
```

Параметров со значениями по умолчанию может быть любое количество:

```
def f(a=5, b=10, c=100):
```

У значений по умолчанию есть одно ограничение. Они должны идти в самом конце списка параметров. С точки зрения синтаксиса, невозможно создать функцию, у которой после необязательного параметра идет обязательный:

```
# Такой код завершится с ошибкой
def f(a=5, b=10, c=100, x):
# И такой
def f(a=5, b=10, x, c=100):
```

```
# Этот код сработает
def f(x, a=5, b=10, c=100):

# Этот тоже сработает
def f(x, y, a=5, b=10, c=100):
```

Теперь вы умеете работать со значениями параметров по умолчанию. Они могут быть как у нескольких параметров, так и у одного. И помните, что значения по умолчанию должны быть в самом конце списка параметров. Эти знания помогут сократить количество одинакового кода.

Задание

Реализуйте функцию `get_hidden_card()`, который принимает на вход номер кредитки (состоящий из 16 цифр) в виде строки и возвращает его скрытую версию, которая может использоваться на сайте для отображения. Если исходная карта имела номер *2034399002125581*, то скрытая версия выглядит так *****5581*. Другими словами, функция заменяет первые 12 символов, на звездочки. Количество звездочек регулируется вторым, необязательным, параметром. Значение по умолчанию — 4.

```
# Кредитка передается внутрь как строка
# Второй параметр не передается, значит звездочек будет 4
get_hidden_card('1234567812345678') # ****5678
```

```
get_hidden_card('1234567812345678', 2) # **5678
get_hidden_card('1234567812345678', 3) # ***5678

# Или используя переменные

card_number = '2034399002121100'
get_hidden_card(card_number) # ****1100
get_hidden_card(card_number, 1) # *1100
```

Для выполнения задания вам понадобится механизм повторения строк, который повторяет строку указанное количество раз. Для этого достаточно умножить строку на число повторений:

```
'+' * 5 # +++++
'o' * 3 # ooo
```

- ▶ 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 🙄

Полезное

- Параметры по умолчанию

Определения

- Параметр по умолчанию — необязательный

параметр функции