

1. Напишите функцию `make_sentence`, которая принимает один именованный аргумент `words`, который должен быть списком строк, и возвращает строку, составленную из элементов списка, разделенных пробелами и заканчивающуюся точкой. Если аргумент `words` не указан, то по умолчанию используется список `["This", "is", "a", "sentence"]`.

Пример вызова:

```
make_sentence(words=["Python", "is", "fun"])  
# Python is fun.
```

2. Напишите функцию `sum_of_squares`, которая принимает произвольное количество позиционных аргументов, которые должны быть числами, и возвращает сумму их квадратов. Если функции не передано ни одного аргумента, она должна вернуть 0.
3. Напишите функцию `greet`, которая принимает два именованных аргумента: `name` и `language`. Аргумент `name` должен быть строкой, а аргумент `language` должен быть одним из трех возможных значений: `"en"`, `"ru"` или `"fr"`. Функция должна возвращать приветствие на выбранном языке. Если аргумент `language` не указан, то по умолчанию используется `"en"`.

Пример вызова:

```
greet(name="Anna", language="en")  
# Hello, Anna!
```

4. Напишите функцию `print_info`, которая принимает произвольное количество именованных аргументов (`**kwargs`) и выводит их в формате `"key: value"` по одной паре на строку. Напоминаю, что `kwargs` в функции будет словарем. Если функции не передано ни одного аргумента, она должна вывести `"No info given."`.

Пример вызова:

```
print_info(name="Alex", age=25, city="Amsterdam")
```

```
name: Alex  
age: 25  
city: Amsterdam
```

5. Напишите функцию `print_table`, которая принимает произвольное количество именованных аргументов в виде пар ключ-значение и выводит их в виде таблицы с двумя столбцами: "Key" и "Value". Если функции не передано ни одного аргумента, она должна вывести "No data given."

Пример вызова:

```
print_table(name="Bob", age=30, city="Amsterdam")
```

```
| Key | Value |
|-----|-----|
| name | Bob   |
| age  | 30    |
| city | Amsterdam |
```

6. Напишите функцию `calculate`, которая принимает произвольное количество позиционных аргументов, которые должны быть числами, и один именованный аргумент `operation`, который должен быть одним из четырех возможных значений: "+", "-", "*" или "/".

Функция должна возвращать результат выполнения указанной операции над всеми числами в порядке их передачи.

Если функции не передано ни одного позиционного аргумента, она должна вернуть 0.

Если аргумент `operation` не указан, то по умолчанию используется "+".

Пример вызова: `calculate(1, 2, 3, operation="*")`

7. Напишите функцию `print_triangle`, которая принимает один именованный аргумент `height`, который должен быть положительным целым числом, и выводит равнобедренный треугольник из символов "*" с заданной высотой. Если аргумент `height` не указан, то по умолчанию используется число 5.

Пример вызова: `print_triangle(height=4)`

```
  *
 ***
*****
*****
```

8. Напишите функцию `create_post`, которая создает пост для блога, основываясь на переданных параметрах. Обязательными параметрами являются: заголовок, содержимое и автор. Необязательным параметром является категория. Если она не была передана, то по умолчанию будет текущее значение “general”. Функция должна возвращать словарь поста.
9. Напишите функцию `create_product`, которая создает товар для интернет-магазина, основываясь на переданных параметрах. Обязательными параметрами являются: имя, цена и категория. Необязательным параметром является рейтинг. Если он не был передан параметр, то по умолчанию будет 0. Функция должна возвращать словарь товара.
10. Напишите функцию `create_student`, которая создает словарь студента для учебного заведения, основываясь на переданных параметрах. Обязательными параметрами являются: имя, фамилия, отчество и группа. Также дополнительными параметрами могут быть: дата поступления в виде строки «19.10.2023», средний бал, семестр обучения, номер телефона, адрес. Функция должна возвращать словарь студента только с переданными данными, если некоторые данные не были переданы, то их не должно быть в словаре.