Лабораторная работа 8

Декораторы. Два уровня заданий на выбор.

Задание попроще

1. Напишите декоратор **@uppercase**, который преобразует возвращаемое функцией строковое значение в верхний регистр.
2. Создайте декоратор **@count_calls**, который подсчитывает, сколько раз была вызвана функция.

```
@count_calls
def greet(name):
    print(f"Hello, {name}!")

greet("Tom")
greet("Denis")
print(f"Функция greet вызвана {greet.call_count} раз(а).").
```

3. Создайте декоратор **@html_tag(tag)**, который оборачивает возвращаемую строку в HTML-тег, заданный в параметре, см. пример:

```
@html_tag("div")
def get_text():
    return "Hello, World!"
print(get_text())  # Вывод: <div>Hello, World!</div>
```

Задание поинтереснее

1. Напишите декоратор @**timer**, который замеряет и выводит время выполнения функции. Пример вызова:

```
@timer
def some_function(n):
    return sum(i * i for i in range(n))

some_function(10**6)
```

2. Создайте декоратор **@log**, который записывает в файл информацию о вызове функции (имя, аргументы, результат).
3. Напишите декоратор **@memory_usage**, который замеряет объем памяти, используемый функцией во время выполнения. Используйте memory_profiler или sys.getsizeof().