1. Напишите декоратор, который считает сколько раз была вызвана декорируемая функция. Декоратор может быть использован для функций с различным количеством аргументов.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Напишите декоратор, который сохраняет в log файл название декорируемой функции, ее аргументы и key-word аргументы, а также считает сколько времени выполнялась декорируемая функция. Декоратор может быть использован для функций с различным количеством аргументов.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Напишите декоратор, который принимает 2 аргумента – **html tag** и число **N**.

В результате декорирования функции, которая принимает аргументом **строку** вы должны получить список строк длиной **N** в формате <**html tag**> **строка** *с заглавной буквы + порядковый номер элемента в списке, начиная с 1*</**html tag** >.

Если строка, передаваемая в качестве аргумента функции, содержит спецсимволы @, #, %, &, $, ^, \*, \_ нужно каждый из них заменить на пробел, а символы <, >, / удалить из строки.

**Пример:**

@html\_decorator("li", 3)

def html\_element(text):

…

print(html\_element('>list\_item<'))

**Результат:**

['<li>List item 1</li>', '<li>List item 2</li>', '<li>List item 3</li>']

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_