Лабораторная работа 8.

Декораторы.

Платформа:

<https://tio.run/>

Цели:

* Знакомство с декораторами.
* Понять области применения декораторов

Описание:

Декоратор – функция, принимающая в качестве аргумента, декорируемую функцию и, как правило, возвращающая ее в изменённом виде, и/или обогащая её дополнительным функционалом.

Пример:

Есть функция, выполняющая некоторые расчёты вида f(a,b,c). Необходимо как-то измерять время её выполнения без массового изменения кода.

function myFunc(a,b) {

result = 0

for(let i = 1; i < a; i++)

for(let j = 1; j < b; j++)

result = (result + i \* j + 1) % j

return result

}

console.log(myFunc(10000,10000))

function checkExecTime(fn) {

return function(...args) {

let timeStart = Date.now()

let result = fn(...args)

let timeEnd = Date.now()

console.log("Execution time = " + (timeEnd - timeStart) + " ms")

return result

}

}

myDecoratedFunc = checkExecTime(myFunc)

console.log(myDecoratedFunc(10000,10000))

Таким образом, в частности, можно тестировать функции перед их непосредственным запуском, добавлять отладочную информацию в лог при каждом вызове, изменять или добавлять параметры (если нужно).  
Также этот пример наглядно иллюстрирует, работу spread-оператора, который позволяет декорировать одинаковым функционалом совершенно любые функции, принимающие произвольное количество аргументов.

Задание:

1. Напишите функцию-декоратор для вашей функции div(a,b), возвращающей текстовую строку «a поделить на b равно <результат>», который перед запуском декорируемой функции защищает её от вызовов с b=0
2. Напишите функцию-декоратор, пишущий в лог все вызовы декорируемой функции, ее аргументы, ее результат и дату вызова.  
   - Используйте внешний массив (переменную) для хранения данных. В него элементами должны помещаться ассоциативные массивы вида {“params”:[значения аргументов], “result”: <что вернула функция>, “date”:”строка со временем”}- время можно получить вызовом Date() - для получения массива входящих значений можно использовать конструкцию […args]

Варианты:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | PHP | 4 | Ruby | 7 | JavaScript |
| 2 | Java | 5 | Haskell | 8 | Python |
| 3 | Go | 6 | Lua | 9 | C(любой) |