

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №15
дисциплины «Основы программной инженерии»

Выполнил:
Плугатырев Владислав Алексеевич
2 курс, группа ПИЖ-б-о-22-1,
09.03.04 «Программная инженерия»,
направленность (профиль) «Разработка и
сопровождение программного
обеспечения», очная форма обучения

(подпись)

Доцент кафедры инфокоммуникаций
Воронкин Роман Александрович

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2023 г.

Тема: декораторы функций в языке Python.

Цель работы: приобретение навыков по работе с декораторами функций при написании программ с помощью языка программирования Python версии 3.x.

Порядок выполнения работы

1. Создал репозиторий.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk ().*

Owner * **Repository name ***

BigLofanbl4 / lab15

✔ lab15 is available.

Great repository names are short and memorable. Need inspiration? How about [laughing-octo-giggle](#) ?

Description (optional)

☒ **Public**
Anyone on the internet can see this repository. You choose who can commit.

☐ **Private**
You choose who can see and commit to this repository.

Initialize this repository with:

☒ **Add a README file**
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

This will set `main` as the default branch. Change the default name in your [settings](#).

You are creating a public repository in your personal account.

Рисунок 1.1 – Создание репозитория

2. Проработал примеры из лабораторной работы.

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

def benchmark(func):
    import time
    def wrapper(*args, **kwargs):
        start = time.time()
        return_value = func(*args, **kwargs)
        end = time.time()
        print('[*] Время выполнения: {} секунд.'.format(end - start))
        return return_value

    return wrapper

@benchmark
def fetch_webpage(url):
    import requests
    webpage = requests.get(url)
    return webpage.text

if __name__ == "__main__":
    print(fetch_webpage("https://google.com"))
```

Рисунок 2.1 – Код примера из лабораторной работы

```
вы программной инженерии/13/ tab13/example.py
[*] Время выполнения: 1.061335802078247 секунд.
<!doctype html><html itemscope="" itemtype="http://schema.org/WebPage"
```

Рисунок 2.2 – Вывод программы

3. Индивидуальное задание. Вариант 14.

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

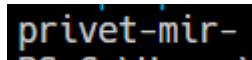
def func_decorator(func):
    def wrapper(*args, **kwargs):
        return_value = func(*args, **kwargs)
        while "---" in return_value:
            return_value = return_value.replace("---", "-")
        return return_value
    return wrapper

@func_decorator
def to_lat(string):
    t = {'ë': 'yo', 'а': 'a', 'б': 'b', 'в': 'v', 'г': 'g', 'д': 'd', 'е': 'e', 'ж': 'zh',
        'з': 'z', 'и': 'i', 'й': 'y', 'к': 'k', 'л': 'l', 'м': 'm', 'н': 'n', 'о': 'o', 'п': 'p',
        'р': 'r', 'с': 's', 'т': 't', 'у': 'u', 'ф': 'f', 'х': 'h', 'ц': 'c', 'ч': 'ch', 'ш': 'sh',
        'щ': 'shch', 'ъ': ' ', 'ы': 'y', 'ь': ' ', 'э': 'e', 'ю': 'yu', 'я': 'ya',
        '!': '-', '?': '-', ':': '-', ';': '-', '.': '-', ',': '-', '_': '-'}

    string = string.lower()
    for key, val in t.items():
        string = string.replace(key, val)
    return string

if __name__ == "__main__":
    print(to_lat("Привет!!!---Мир??.."))
```

Рисунок 3.1 – Код программы



privet-mir-
PS C:\Users\...

Рисунок 3.2 – Вывод программы

Ответы на контрольные вопросы

1. Декоратор - это функция, которая позволяет обернуть другую функцию для расширения её функциональности без непосредственного изменения её кода.
2. В Python функции являются объектами первого класса, что означает, что они могут быть присвоены переменным, переданы в качестве аргументов другим функциям и возвращены из функций.
3. Функции высших порядков - это функции, которые принимают другие функции в качестве аргументов или возвращают функции в качестве результата.
4. Декораторы работают путем обертывания функции, которую они декорируют, в другую функцию, которая может изменять её поведение.
5. Структура декоратора функций обычно состоит из двух функций: декоратора и декорируемой функции. Декоратор обычно определяется как функция, которая принимает декорируемую функцию в качестве аргумента и возвращает новую функцию, которая оборачивает декорируемую функцию 1.
6. Чтобы передать параметры декоратору, а не декорируемой функции, можно определить декоратор как функцию, которая принимает параметры и возвращает другую функцию, которая, в свою очередь, принимает декорируемую функцию в качестве аргумента.