

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

Кафедра инфокоммуникаций

**Основы кроссплатформенного программирования
Отчет по лабораторной работе №6**

Декораторы функций в языке Python

Выполнила студентка группы
ИТС-б-о-20-1 (2)

Скачедубова А.В « » _____ 20__ г.

Подпись студента _____

Работа защищена « » _____ 20__ г.

Проверил к.т.н., доцент

Кафедры инфокоммуникаций

Воронкин Р.А.

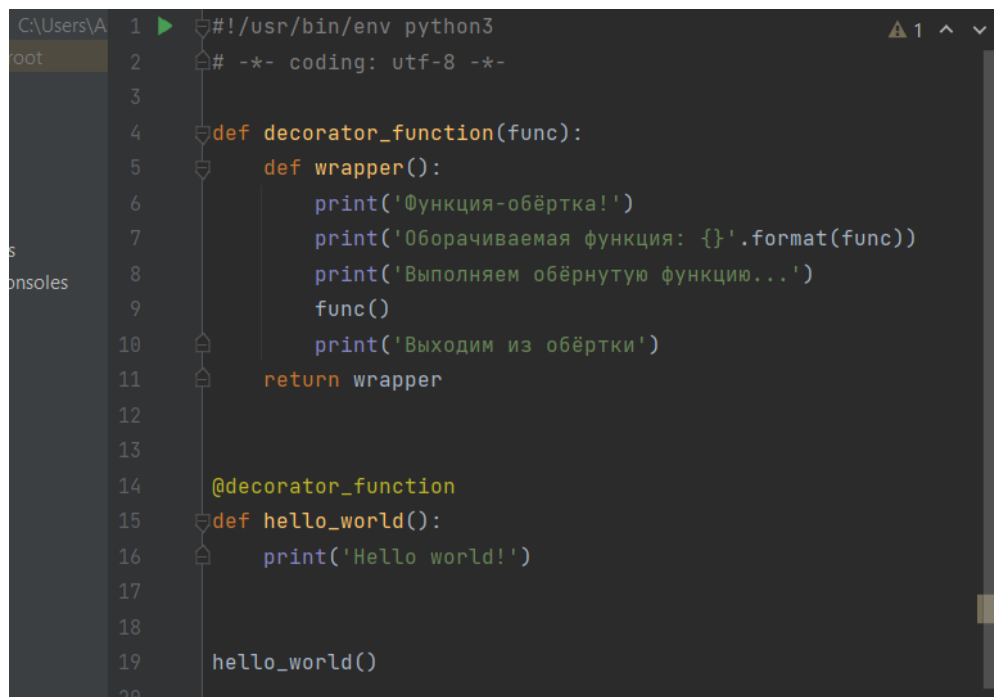
(подпись)

Цель работы: приобрести навыки по работе с декораторами функций при написании программ с помощью языка программирования Python.

Ссылка на репозиторий: https://github.com/Any3002/Lab_6

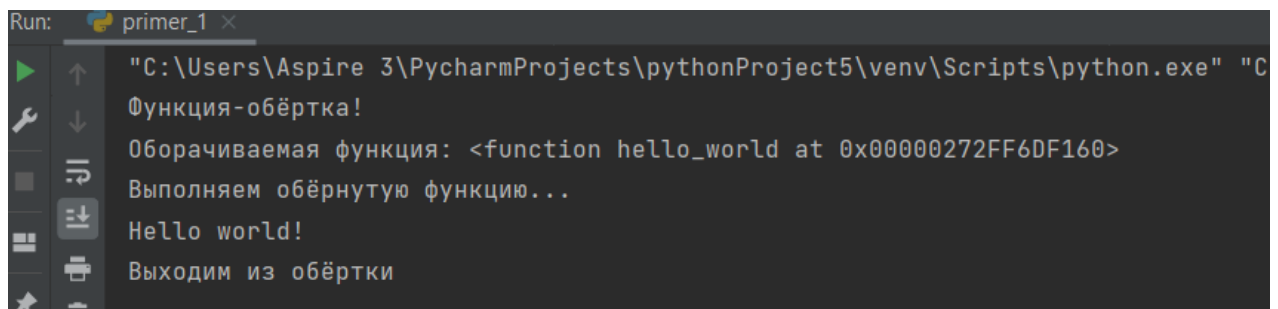
Порядок выполнения работы:

1. Изучила теоретический материал.
2. Создала общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.
3. Проработала примеры лабораторной работы:



```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  def decorator_function(func):
5      def wrapper():
6          print('Функция-обёртка!')
7          print('Оборачиваемая функция: {}'.format(func))
8          print('Выполняем обёрнутую функцию...')
9          func()
10         print('Выходим из обёртки')
11     return wrapper
12
13
14  @decorator_function
15  def hello_world():
16      print('Hello world!')
17
18
19  hello_world()
20
```

Рисунок 1 – Отработанный пример №1



```
Run: primer_1 x
"C:\Users\Aspire 3\PycharmProjects\pythonProject5\venv\Scripts\python.exe" "C:\Users\Aspire 3\PycharmProjects\pythonProject5\venv\Scripts\python.exe"
Функция-обёртка!
Оборачиваемая функция: <function hello_world at 0x00000272FF60DF160>
Выполняем обёрнутую функцию...
Hello world!
Выходим из обёртки
```

Рисунок 2 – Результат примера №1

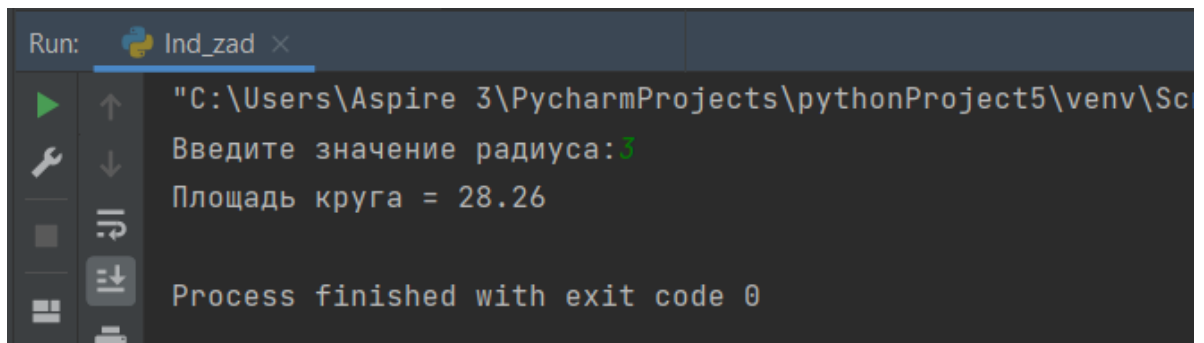
4. Выполнила индивидуальное задание. Вариант 18(8):

Объявите функцию, которая вычисляет площадь круга и возвращает вычисленное значение. В качестве аргумента ей передается значение радиуса. Определите декоратор для этой функции, который выводит на экран

сообщение: «Площадь круга равна =». В строке выведите числовое значение с точностью до сотых. Примените декоратор к функции и вызовите декорированную функцию.

```
1  #!/usr/bin/venv python3
2  # -*- coding: utf-8 -*-
3  # Объявите функцию, которая вычисляет площадь круга и возвращает вычисленное значение.
4  # В качестве аргумента ей передается значение радиуса.
5  # Определите декоратор для этой функции, который выводит на экран сообщение:
6  # «Площадь круга равна = <число>». В строке выведите числовое значение с точностью до сотых.
7  # Примените декоратор к функции и вызовите декорированную функцию. Вариант 8
8
9
10 def decorator(func):
11     def decorator1(x):
12         c = "Площадь круга"
13         ret = func(x)
14         print(f'{c} = {ret}')
15         return ret
16     return decorator1
17
18
19 @decorator
20 def add(x):
21     pi = 3.14
22     return x * x * pi
23
24
25 if __name__ == "__main__":
26     r = float(input("Введите значение радиуса:"))
27     add(r)
```

Рисунок 3 – Код задания №1



```
Run: Ind_zad x
"C:\Users\Aspire 3\PycharmProjects\pythonProject5\venv\Sc
Введите значение радиуса: 3
Площадь круга = 28.26
Process finished with exit code 0
```

Рисунок 4 – Результат задания №1

5. Зафиксировала изменения в репозитории.
6. Добавила отчет по лабораторной работе в формате PDF в репозиторий.

Ответы на вопросы:

1. Что такое декоратор?

Ответ: декоратор – это функция, которая позволяет обернуть другую функцию для расширения ее функциональности без непосредственного изменения ее кода.

2. Почему функции являются объектами первого класса?

Ответ: объектами первого класса в контексте конкретного языка программирования называются элементы, с которыми можно делать все то же, что и с любыми другими объектами: передавать как параметр, возвращать из функции и присваивать переменной

3. Каково назначение функций высших порядков?

Ответ: функции высших порядков– это функции, которые могут принимать в качестве аргументов и возвращать другие функции.

4. Как работают декораторы?

Ответ: они позволяют обворачивать другие функции при помощи символа @.

5. Какова структура декоратора функций?

Ответ: для начала записывается декоратор. Далее идет вызов с помощью @, и потом основная функция, которую обворачивает декоратор.

6. Самостоятельно изучить как можно передать параметры декоратору, а не декорируемой функции?

Ответ: если декоратор вызван без скобок, то единственным его параметром будет декорируемая функция (что подпадает под определение чистого декоратора), которая попадет в переменную `_func`, а иначе она будет равна `None`. Если же, мы передаем только именованный параметр, например, `n=10`, то остается `_func = None` и благодаря этому декоратор понимает вызвали его с параметром или нет.

Вывод по работе: в ходе лабораторной работы были приобретены навыки по работе с декораторами функций при написании программ с помощью языка программирования Python.