

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 ПРАКТИЧЕСКОЙ РАБОТЕ №2.12
дисциплины «Программирование на Python»

Выполнил:
Степанов Леонид Викторович
2 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и
вычислительная техника»,
направленность (профиль)
«Программное обеспечение
средств вычислительной техники
и автоматизирование систем»,
очная форма обучения

(подпись)

Руководитель практики:
Воронкин Р.А., канд. техн. наук,
доцент, доцент кафедры
инфокоммуникаций

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

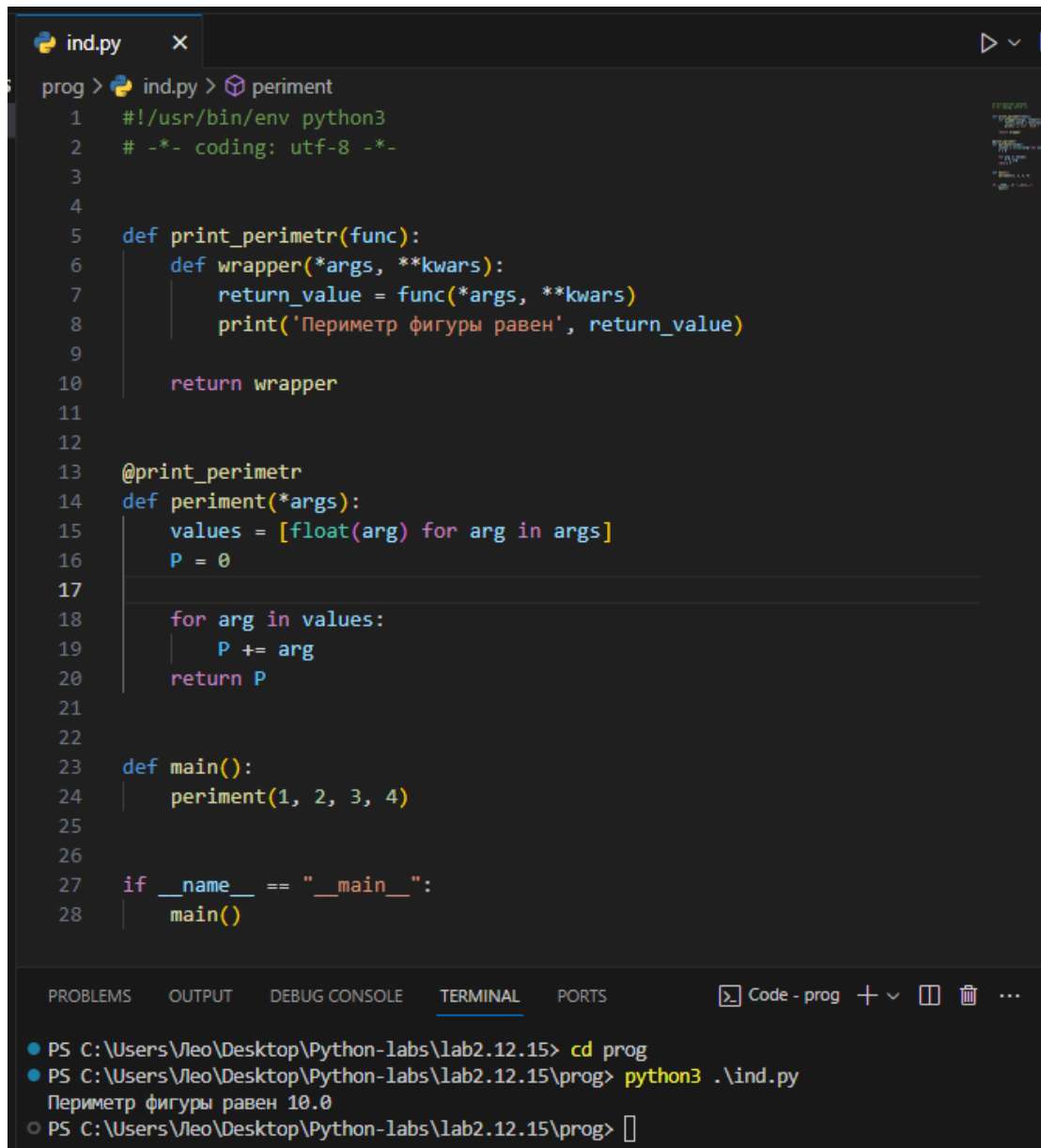
Ставрополь, 2023 г.

Тема: Декораторы функций в языке Python

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

Порядок выполнения работы:

Индивидуальное задание (Вариант 7): объявите функцию, которая вычисляет периметр многоугольника и возвращает вычисленное значение. Длины сторон многоугольника передаются в виде коллекции (списка или кортежа). Определите декоратор для этой функции, который выводит на экран сообщение: «Периметр фигуры равен = <число>». Примените декоратор к функции и вызовите декорированную функцию.



```
ind.py x
prog > ind.py > periment
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4
5  def print_perimetr(func):
6      def wrapper(*args, **kwargs):
7          return_value = func(*args, **kwargs)
8          print('Периметр фигуры равен', return_value)
9
10         return wrapper
11
12
13  @print_perimetr
14  def periment(*args):
15      values = [float(arg) for arg in args]
16      P = 0
17
18      for arg in values:
19          P += arg
20      return P
21
22
23  def main():
24      periment(1, 2, 3, 4)
25
26
27  if __name__ == "__main__":
28      main()
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Code - prog + - [] [] ...

- PS C:\Users\Ileo\Desktop\Python-labs\lab2.12.15> cd prog
- PS C:\Users\Ileo\Desktop\Python-labs\lab2.12.15\prog> python3 .\ind.py
Периметр фигуры равен 10.0
- PS C:\Users\Ileo\Desktop\Python-labs\lab2.12.15\prog> []

Рисунок 1 – Результат выполнения программ ind.py

Вывод: в ходе выполнения работы были приобретены навыки по работе с декораторами функций при написании программ с помощью языка программирования Python версии 3.x.

Контрольные вопросы

1. Что такое декоратор?

Декоратор — это функция, которая позволяет обернуть другую функцию для расширения её функциональности без непосредственного изменения её кода.

2. Почему функции являются объектами первого класса?

потому что они могут быть присвоены переменной, переданы в качестве аргумента другой функции, возвращены из функции и сохранены в структурах данных.

3. Каково назначение функций высших порядков?

могут принимать другие функции в качестве аргументов или возвращать их в качестве результатов. Они позволяют абстрагировать действия и оперировать функциями, как данными.

4. Как работают декораторы?

Работают путем обертывания другой функции, позволяя добавлять новое поведение к этой функции без изменения ее кода. Декораторы принимают функцию в качестве аргумента, возвращают другую функцию и обычно используются с символом @.

5. Какова структура декоратора функций?

Структура декоратора функций обычно выглядит следующим образом:

```
def decorator_function(original_function):  
    def wrapper_function(*args, **kwargs):  
        # Добавление нового поведения  
        return original_function(*args, **kwargs)  
    return wrapper_function
```

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

Параметры могут быть переданы декоратору, а не декорируемой функции, путем добавления еще одного уровня вложенной функции в декораторе. Этот уровень может принимать параметры и передавать их в обернутую функцию.