МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ ЦИФРОВОГО РАЗВИТИЯ

Отчет о лабораторной работе №15 по дисциплине «Основы программной инженерии»

Выполнил:

Мамонтов Д.В.,

2 курс, группа ПИЖ-б-о-20-1,

Проверил:

Доцент кафедры инфокоммуникаций,

Воронкин Р.А.

Ставрополь, 2022 г

**ХОД РАБОТЫ**

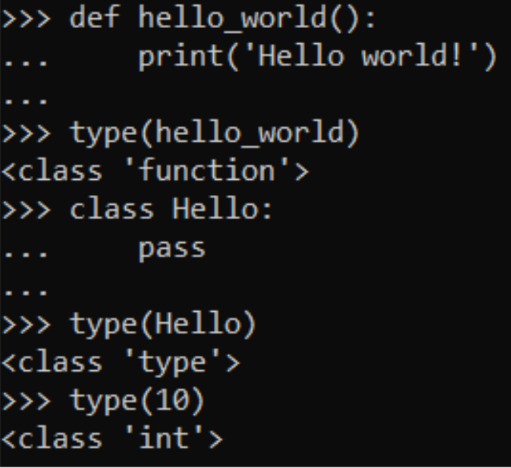


Рисунок 1 – изменение типа переменной

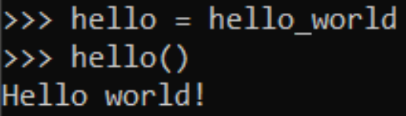


Рисунок 2 – присвоение

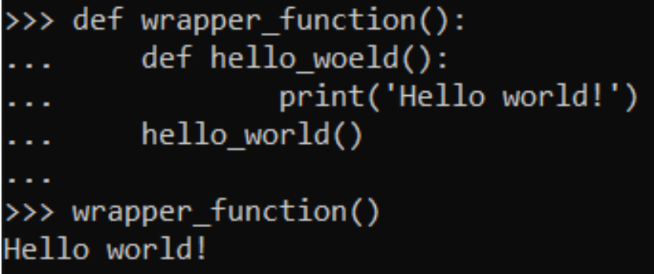


Рисунок 3 – применение декоратора

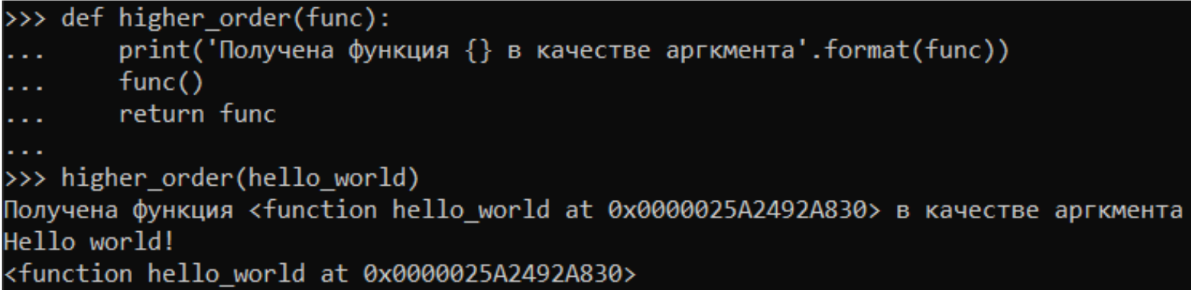


Рисунок 4 – функция как значение аргумента

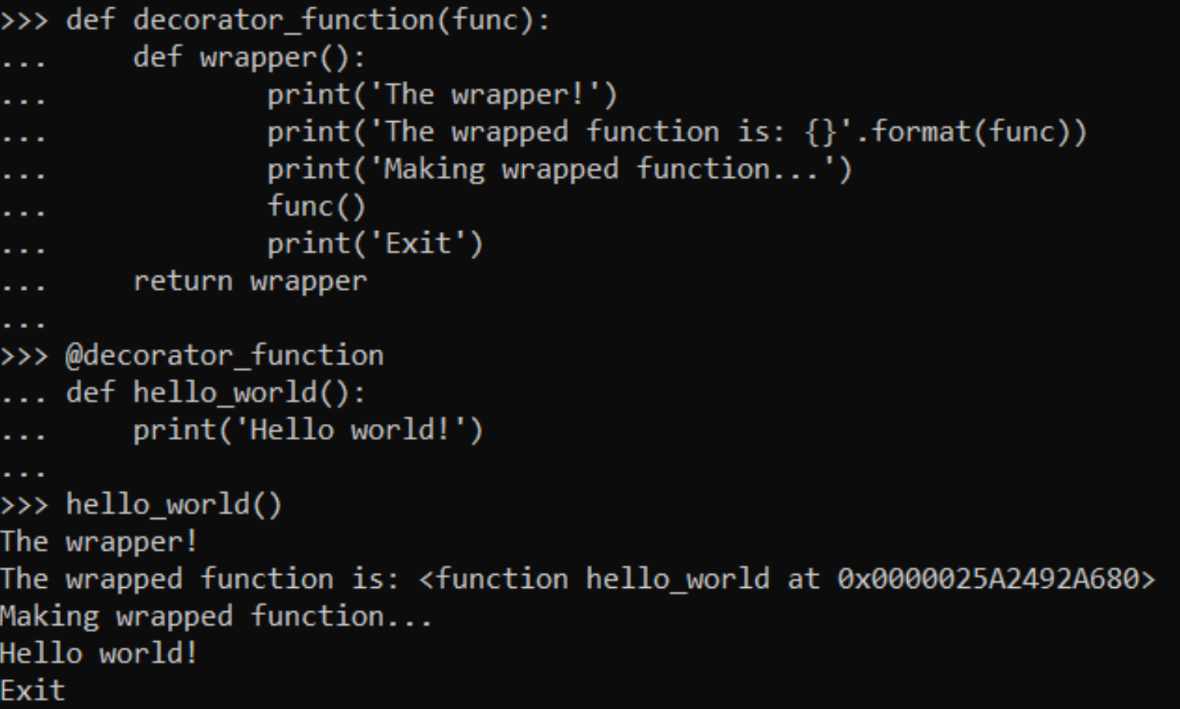


Рисунок 5 – применение декоратора

**ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ**

Задание:

Вводятся два списка (каждый с новой строки) из слов, записанных через пробел. Имеется функция, которая преобразовывает эти две строки в два списка слов и возвращает эти списки. Определите декоратор для этой функции, который из этих двух списков формирует словарь, в котором ключами являются слова из первого списка, а значениями – соответствующие элементы из второго списка. Полученный словарь должен возвращаться при вызове декоратора. Примените декоратор к первой функции и вызовите ее. Результат (словарь) отобразите на экране.

Код:

def start\_func(text1, text2):  
 return text1.split(), text2.split()  
  
  
def decorated(func):  
 def decorated\_m(text1, text2):  
 data = func(text1, text2)  
 return dict(zip(\*data))  
 return decorated\_m  
  
  
start\_func = decorated(start\_func)  
x = input()  
y = input()  
print(start\_func(x, y))

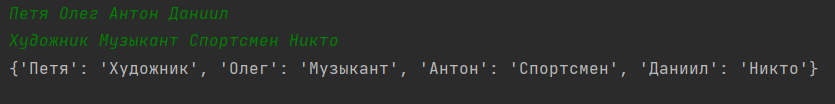


Рисунок 1 – результат работы программы

**КОНТРОЛЬНЫЕ ВОПРОСЫ**

1. Что такое декоратор?

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

1. Почему функции являются объектами первого класса? Объектами первого класса в контексте конкретного языка

программирования называются элементы, с которыми можно делать всё то же, что и с любым другим объектом: передавать как параметр, возвращать из функции и присваивать переменной.

1. Каково назначение функций высших порядков?

Функции высших порядков — это такие функции, которые могут принимать в качестве аргументов и возвращать другие функции.

1. Как работают декораторы?

Используя конструкцию @decorator def function(), мы делаем конструкцию вида function=decorator(function), а это значит, что значению нашей функции будет соответствовать значение функции, которую вернул декоратор.

1. Какова структура декоратора функций?

def decorator\_function(func):

def wrapper(): print('Функция-обёртка!')

print('Оборачиваемая функция: {}'.format(func)) print('Выполняем обёрнутую функцию...') func()

print('Выходим из обёртки') return wrapper

1. Как можно передать параметры декоратору, а не декорируемой функции?

def decorator\_setup(start=0): def decorator\_function(func):

def wrapper(args): result = func(args) return result + start

return wrapper

return decorator\_function