

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 №14
дисциплины «Программирование на Python»

Выполнил:
Середа Кирилл Витальевич
1 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и
вычислительная техника», очная
форма обучения

(подпись)

Руководитель практики:
Воронкин Роман Александрович

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

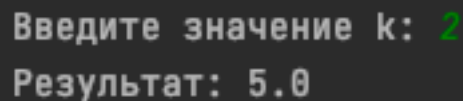
Ставрополь, 2023 г.

Тема: Замыкания в языке Python

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

Ход выполнения:

1) Выполнил индивидуальное задание – Вариант 1: Используя замыкания функций, определите вложенную функцию, которая бы увеличивала значение переданного параметра на 3 и возвращала бы вычисленный результат. Вызовите внешнюю функцию для получения ссылки на внутреннюю функцию и присвойте ее переменной с именем `cnt`. Затем, вызовите внутреннюю функцию через переменную `cnt` со значением `k`, введенным с клавиатуры.



Введите значение k: 2
Результат: 5.0

Рисунок 1 – Результат выполнения программы

Код программы:

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

def outer_function():
    def inner_function(x):
        return x + 3
    return inner_function

if __name__ == "__main__":
    cnt = outer_function()

    k = float(input("Введите значение k: "))

    result = cnt(k)

    print(f"Результат: {result}")
```

Ответы на вопросы:

1. Определение замыкания: Замыкание (closure) в программировании — это функция, в теле которой присутствуют ссылки на переменные, объявленные вне тела этой функции в окружающем коде и не являющиеся ее параметрами.

2. Реализация замыканий в Python: Замыкания (closures) в языке Python реализованы так, что вложенные функции имеют доступ к переменным внешней функции даже после завершения работы внешней функции.

3. Область видимости Local: Область видимости Local относится к переменным, созданным и используемым внутри функций. Эти переменные недоступны вне функции.

4. Область видимости Enclosing: Enclosing область видимости относится к локальным переменным внутри функции и ее вложенных функций.

5. Область видимости Global: Global относится к глобальным переменным уровня модуля. Доступны из любой функции в данном модуле, но не из других модулей при импорте.

6. Область видимости Built-in: Built-in относится к уровню интерпретатора Python, включая встроенные функции, исключения и прочие сущности доступные в любом модуле без импорта.

7. Использование замыканий в Python: Пример использования замыканий:

```
def mul(a):  
    def helper(b):  
        return a * b  
    return helper
```

```
new_mul5 = mul(5)  
result = new_mul5(2)
```

В данном примере переменная `a` сохраняет свое значение внутри возвращаемой функции `helper`, создавая замыкание.

8. Замыкания для построения иерархических данных: Пример использования замыканий для построения иерархических данных:

9. `tpl = lambda a, b: (a, b)`

10.a = tpl(1, 2)

11.b = tpl(3, a)

12.c = tpl(a, b)

В этом примере функция `tpl` порождает структуру данных `((1, 2), (3, (1, 2)))`, используя замыкания для сохранения значений.

Вывод: В результате выполнения лабораторной работы были получены навыки по работе с замыканиями при написании программ с использованием языка программирования Python версии 3.x.