

**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 №2.11**

**Замыкания в языке Python**

Выполнил студент группы  
ИТС-б-о-21-1

Крамаренко И.В. « » \_\_\_\_\_ 20\_\_ г.

Подпись студента \_\_\_\_\_

Работа защищена « » \_\_\_\_\_ 20\_\_ г.

Проверил к.т.н., доцент

Кафедры инфокоммуникаций

Воронкин Р.А.

---

(подпись)

Ставрополь 2022

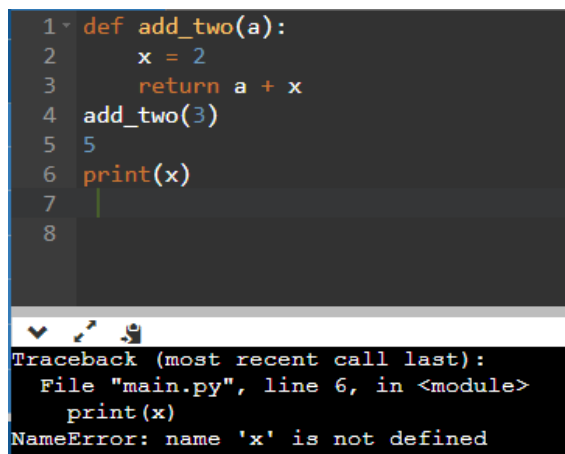
## Лабораторная работа 2.11 Замыкания в языке Python.

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

**Теоретический материал: Замыкание** – это 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.

### Ход работы:

1. Изучил теоретический материал работы.
2. Создал общедоступный репозиторий на GitHub, в котором использовал лицензию MIT и язык программирования Python.
3. Выполнил клонирование нового репозитория.
4. Дополнил файлы .gitignore необходимыми файлами для работы с IDE PyCharm/
5. Организовал свой репозиторий с моделью ветвления git-flow.
6. Создал проект PyCharm в папке репозитория.
7. Проработал примеры лабораторной работы.



```
1 def add_two(a):
2     x = 2
3     return a + x
4 add_two(3)
5 5
6 print(x)
7
8
```

Traceback (most recent call last):  
File "main.py", line 6, in <module>  
 print(x)  
NameError: name 'x' is not defined

Рисунок 1. Пример 1(С ошибкой)

```
1 def add_four(a):
2     x = 2
3     def add_some():
4         print("x = " + str(x))
5         return a + x
6     return add_some()
7 print(add_four(5))
```

x = 2  
7

Рисунок 6. Пример 2

```
1 x = 4
2 def fun():
3     print(x+3)
4 fun()
```

7

Рисунок 5. Пример 3

```
1 def fun1(a):
2     x = a * 3
3     def fun2(b):
4         nonlocal x
5         return b + x
6     return fun2
7 test_fun = fun1(4)
8 test_fun(7)
9
```

19

...Program finished with exit code 0  
Press ENTER to exit console.

Рисунок 4. Пример 4

8. Выполнил индивидуальное задание.

Вариант 2.

Используя замыкания функций, объявите внутреннюю функцию, которая заключает строку s (s – строка, параметр внутренней функции) в произвольный тег, содержащийся в переменной tag – параметре внешней функции. Далее, на вход программы поступает две строки: первая с тегом,

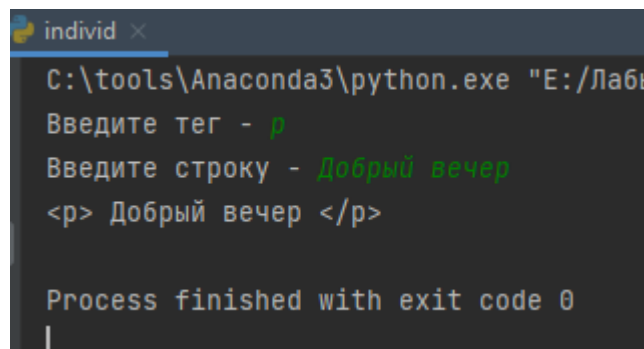
вторая с некоторым содержимым. Вторую строку нужно поместить в тег из первой строки с помощью реализованного замыкания. Результат выведите на экран.

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
# Используя замыкания функций, объявите
# функцию, которая заключает строку s (s – строка)
# в произвольный тег, содержащийся в переменной tag.
# Далее, на вход программы поступает две строки:
# первая – тег, вторая – строка.
# Вторую строку нужно поместить в тег из первой строки.
# Результат выведите на экран

def get_func(tag):
    def func(s):
        return f"<{tag}> {s} </{tag}>"
    return func

if __name__ == '__main__':
    A = str(input("Введите тег - "))
    B = str(input("Введите строку - "))
    print(get_func(A)(B))
```

Рисунок 5. Индивид. Задание



```
individ x
C:\tools\Anaconda3\python.exe "E:/Лабы/individ.py"
Введите тег - p
Введите строку - Добрый вечер
<p> Добрый вечер </p>

Process finished with exit code 0
```

Рисунок 6. Индивид задание (Исходное решение)

9. Зафиксировал изменения в репозитории.
10. Добавил отчёт в формат PDF в папу созданного репозитория.
11. Выполнил слияние ветки для разработки с веткой master/main.

12. Отправил сделанные изменения на сервер GitHub.
13. Отправил ссылку на репозиторий с выполненной работой на адрес преподавателя.

### **Ответы на контрольные вопросы:**

1. Что такое замыкание?

Ответ: 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. Как реализованы замыкания в языке программирования Python?

Ответ: замыкания реализованы с помощью областей видимости (Local, Enclosing, Global, Build-in).

3. Что подразумевает под собой область видимости Local?

Ответ: эту область видимости имеют переменные, которые создаются и используются внутри функций.

4. Что подразумевает под собой область видимости Enclosing?

Ответ: суть данной области видимости в том, что внутри функции могут быть вложенные функции и локальные переменные, так вот локальная переменная функции для ее вложенной функции находится в enclosing области видимости.

5. Что подразумевает под собой область видимости Global?

Ответ: переменные области видимости global – это глобальные переменные уровня модуля (модуль – это файл с расширением .py).

6. Что подразумевает под собой область видимости Build-in?

Ответ: в рамках этой области видимости находятся функции open, len и т. п., также туда входят исключения. Эти сущности доступны в любом

модуле Python и не требуют предварительного импорта. Built-in – это максимально широкая область видимости.

7. Как использовать замыкания в языке программирования Python?

Ответ: можно использовать функции, позволяющие гибко и быстро решить текущую проблему, например функции (`mul()`, `new_mul()` и `fun()`)

8. Как замыкания могут быть использованы для построения иерархических данных?

Ответ: в общем случае, операция комбинирования объектов данных обладает свойством замыкания в том случае, если результаты соединения объектов с помощью этой операции сами могут соединяться этой же операцией

**Вывод:** я приобрёл навыки по работе с замыканиями при написании программ с помощью языка программирования Python версии 3.x.