

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

**Кафедра инфокоммуникаций**

**Языки программирования  
Отчет по лабораторной работе №8**

**Замыкания 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)

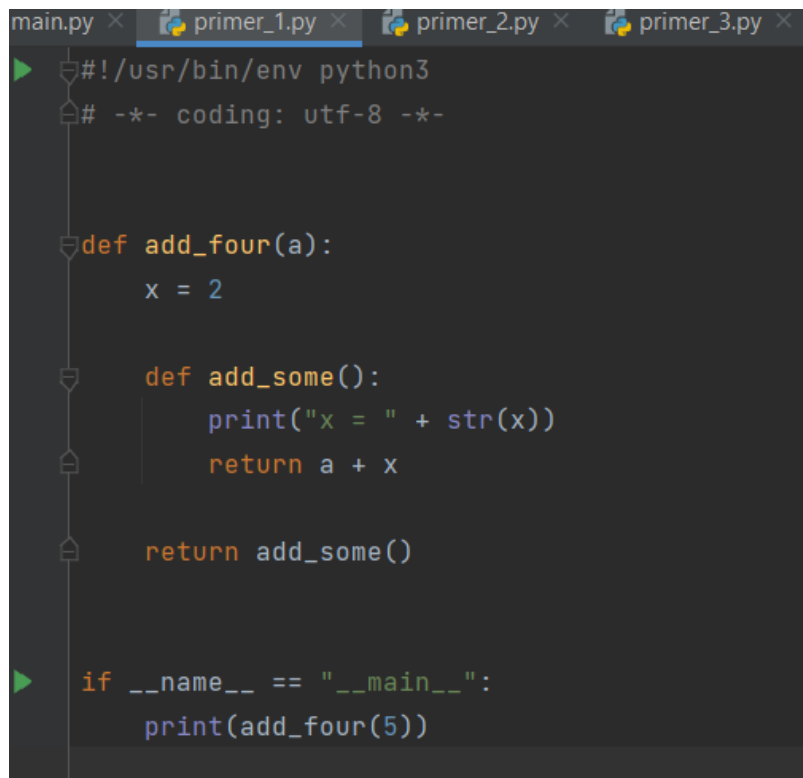
г.Ставрополь 2021

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

Ссылка на репозиторий: [https://github.com/Any3002/Lab\\_8](https://github.com/Any3002/Lab_8)

Порядок выполнения работы:

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



```
main.py x primer_1.py x primer_2.py x primer_3.py x
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

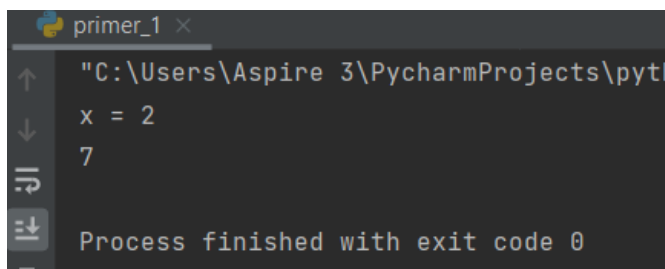
def add_four(a):
    x = 2

    def add_some():
        print("x = " + str(x))
        return a + x

    return add_some()

if __name__ == "__main__":
    print(add_four(5))
```

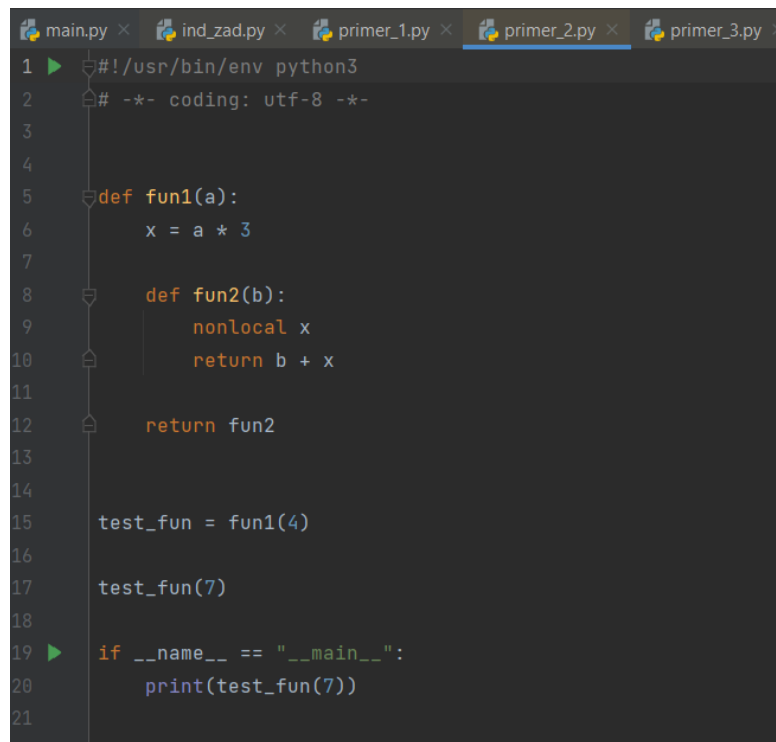
Рисунок 1 – Проработанный пример №1



```
primer_1 x
"C:\Users\Aspire 3\PycharmProjects\pyth
x = 2
7
Process finished with exit code 0
```

Рисунок 2 – Результат примера №1

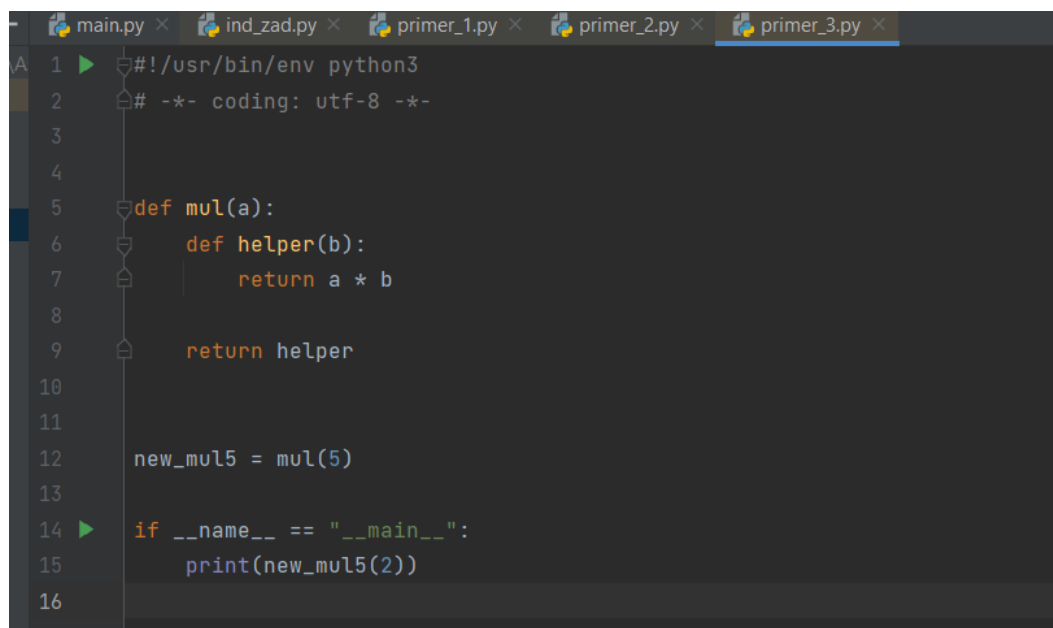
В данном примере переменная x имеет область видимости enclosing для функции add\_some().



```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4
5 def fun1(a):
6     x = a * 3
7
8     def fun2(b):
9         nonlocal x
10        return b + x
11
12    return fun2
13
14
15 test_fun = fun1(4)
16
17 test_fun(7)
18
19 if __name__ == "__main__":
20     print(test_fun(7))
21
```

Рисунок 3 – Проработанный пример №2

В функции `fun1()` объявлена локальная переменная `x`, значение которой определяется аргументом `a`. В функции `fun2()` используются эта же переменная `x`, `nonlocal` указывает на то, что эта переменная не является локальной, следовательно, ее значение будет взято из ближайшей области видимости, в которой существует переменная с таким же именем.

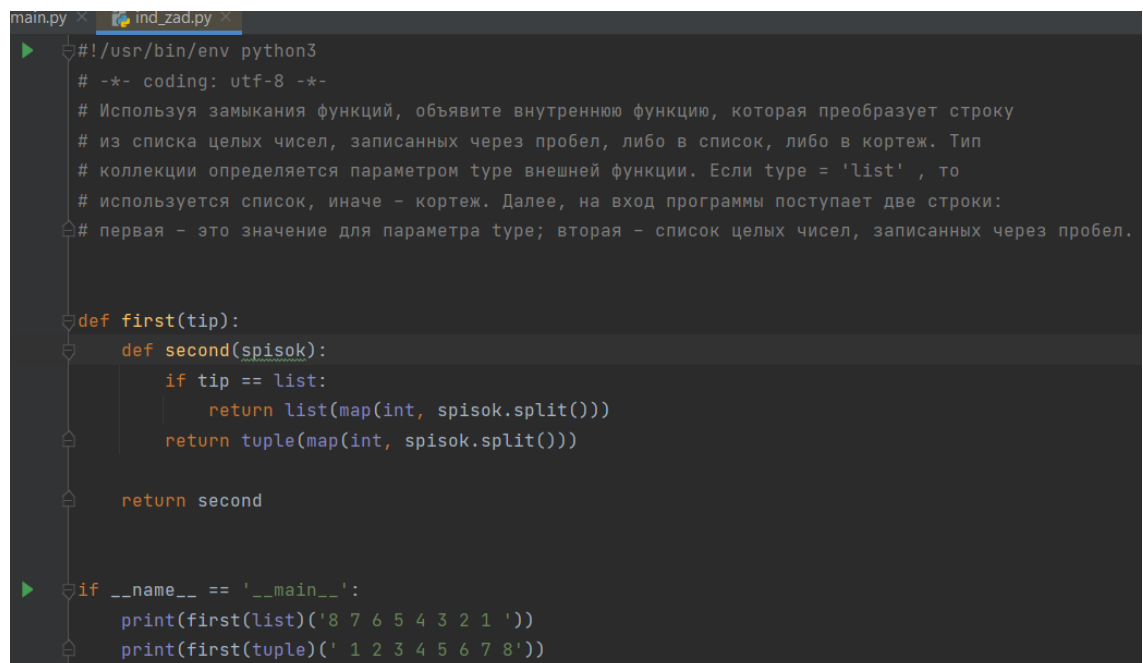


```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4
5 def mul(a):
6     def helper(b):
7         return a * b
8
9     return helper
10
11
12 new_mul5 = mul(5)
13
14 if __name__ == "__main__":
15     print(new_mul5(2))
16
```

Рисунок 4 – Проработанный пример №3

4. Выполнила индивидуальное задание:

Используя замыкания функций, объявите внутреннюю функцию, которая преобразует строку из списка целых чисел, записанных через пробел, либо в список, либо в кортеж. Тип коллекции определяется параметром `type` внешней функции. Если `type = 'list'`, то используется список, иначе – кортеж. Далее, на вход программы поступает две строки: первая – это значение для параметра `type`; вторая – список целых чисел, записанных через пробел. С помощью реализованного замыкания преобразовать эту строку в соответствующую коллекцию. Результат работы замыкания выведите на экран.



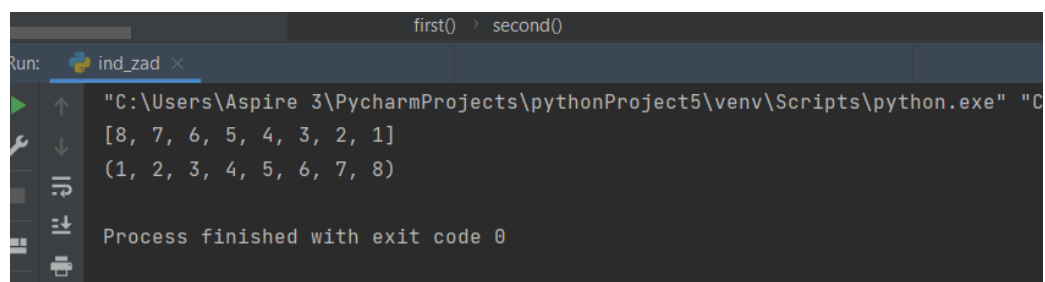
```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
# Используя замыкания функций, объявите внутреннюю функцию, которая преобразует строку
# из списка целых чисел, записанных через пробел, либо в список, либо в кортеж. Тип
# коллекции определяется параметром type внешней функции. Если type = 'list', то
# используется список, иначе – кортеж. Далее, на вход программы поступает две строки:
# первая – это значение для параметра type; вторая – список целых чисел, записанных через пробел.

def first(type):
    def second(spisok):
        if type == list:
            return list(map(int, spisok.split()))
        return tuple(map(int, spisok.split()))

    return second

if __name__ == '__main__':
    print(first(list)('8 7 6 5 4 3 2 1 '))
    print(first(tuple)(' 1 2 3 4 5 6 7 8'))
```

Рисунок 5 – Код индивидуального задания



```
Run: ind_zad x
first() > second()
"C:\Users\Aspire 3\PycharmProjects\pythonProject5\venv\Scripts\python.exe" "C
[8, 7, 6, 5, 4, 3, 2, 1]
(1, 2, 3, 4, 5, 6, 7, 8)
Process finished with exit code 0
```

Рисунок 6 – Результат индивидуального задания

5. Зафиксировала изменения в репозиторий.
6. Добавила отчет по лабораторной работе в формате PDF в репозиторий.
7. Отправила сделанные изменения на сервер репозитория.

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: по области видимости, переменные делят на глобальные и локальные. Глобальные существуют в течении всего времени выполнения программы, а локальные создаются внутри методов, функций и прочих блоках кода, при этом, после выхода из такого блока переменная удаляется из памяти.

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 Built-in?

Ответ: уровень Python интерпретатора. 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: использование замыкания представлено в примерах

8. Как замыкания могут быть использованы для построения иерархических данных

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

Вывод по работе: в ходе лабораторной работы были приобретены навыки по работе с замыканиями при написании программ с помощью языка программирования. Также были изучены видимости Local, Enclosing, Global, Build-in.