

**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 №2.11 по дисциплине основы  
программной инженерии**

Выполнил:

Кожухов Филипп Денисович,  
2 курс, группа ПИЖ-б-о-20-1,

Проверил:

Доцент кафедры  
прикладной математики и  
компьютерной  
безопасности, Воронкин Р.А.

Отчет защищен с оценкой \_\_\_\_\_ Дата защиты \_\_\_\_\_

Ставрополь, 2021 г.

## ВЫПОЛНЕНИЕ:

```
1 ▶  #!/usr/bin/env python3
2    # -*- coding: utf-8 -*-
3
4
5    def add_two(a):
6        x = 2
7        return a + x
8
9
10 ▶  if __name__ == "__main__":
11      print(add_two(3))
12      print(x)
13
```

Рисунок 1 - Код примера №1

5

Process finished with exit code 1

Рисунок 2 - Вывод программы

```
1 ▶ #!/usr/bin/env python3
2   # -*- coding: utf-8 -*-
3
4
5   def add_four(a):
6       x = 4
7
8       def add_some():
9           print("x = " + str(x))
10          return a + x
11
12          return add_some()
13
14
15 ▶ if __name__ == "__main__":
16     print(add_four(5))
17
```

Рисунок 3 - Код примера №2

```
C:\Python37\python.exe C:/Users/student-09-525/PycharmProjects/lab_14/ex2.py
x = 4
9

Process finished with exit code 0
```

Рисунок 4 - Вывод программы

```
1 ▶  #!/usr/bin/env python3
2    # -*- coding: utf-8 -*-
3
4    x = 4
5
6
7    def fun():
8        print(x + 3)
9
10
11 ▶  if __name__ == "__main__":
12    fun()
13
```

Рисунок 5 - Код примера №3

```
C:\Python37\python.exe C:/Users/student-09-525/PycharmProjects/lab_14/ex3.py
7

Process finished with exit code 0
```

Рисунок 6 - Вывод программы

```
1  ▶  #!/usr/bin/env python3
2      # -*- coding: utf-8 -*-
3
4
5      def mul(a):
6          def helper(b):
7              return a * b
8          return helper
9
10
11  ▶  if __name__ == "__main__":
12      print(mul(5)(2))
13      new_mul5 = mul(5)
14      print(new_mul5)
15      print(new_mul5(2))
16      print(new_mul5(7))
17
```

Рисунок 7 - Код примера №4

```
C:\Python37\python.exe C:/Users/student-09-525/PycharmProjects/lab_14/ex4.py
10
<function mul.<locals>.helper at 0x02D8D7C8>
10
35

Process finished with exit code 0
```

Рисунок 8 - Вывод программы

```
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 ▶ if __name__ == "__main__":
16     test_fun = fun1(4)
17     print(test_fun(7))
18
```

Рисунок 9 - Код примера №5

```
C:\Python37\python.exe C:/Users/student-09-525/PycharmProjects/lab_14/ex5.py
19

Process finished with exit code 0
```

Рисунок 10 - Вывод программы

```
1 ▶ #!/usr/bin/env python3  
2 # -*- coding: utf-8 -*-  
3  
4 ▶ if __name__ == "__main__":  
5     tpl = lambda a, b: (a, b)  
6     a = tpl(1, 2)  
7     print(a)  
8     b = tpl(3, a)  
9     print(b)  
10     c = tpl(a, b)  
11     print(c)  
12
```

Рисунок 11 - Код примера №6

```
C:\Python37\python.exe C:/Users/student-09-525/PycharmProjects/lab_14/ex6.py  
(1, 2)  
(3, (1, 2))  
((1, 2), (3, (1, 2)))  
  
Process finished with exit code 0
```

Рисунок 12 - Вывод программы

# ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ

## Вариант 5

```
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4
5 def sample(string):
6     def name_surname(n, s):
7         sample_data = string.replace("%N%", n)
8         sample_data = sample_data.replace("%F%", s)
9         return sample_data
10
11     return name_surname
12
13
14 ▶ if __name__ == "__main__":
15     sample_string = (
16         "Greetings %F% %N%! You are doing the function closure."
17     )
18     name, surname = input("Enter your name and surname: ").split()
19     print(sample(sample_string)(name, surname))
20
```

Рисунок 13 - Код индивидуального задания

```
C:\Python37\python.exe C:/Users/student-09-525/PycharmProjects/lab_14/i1.py
Enter your name and surname: Kozhuhov Philip
Greetings Philip Kozhuhov! You are doing the function closure.

Process finished with exit code 0
|
```

Рисунок 14 - Вывод программы



## ОТВЕТЫ НА ВОПРОСЫ

### 1. Что такое замыкание?

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?

Необходимо объявить вложенную функцию в объемлющей функции. Эта вложенная функция должна ссылаться на значение переменных, объявленных в объемлющей функции, необходимо, чтобы объемлющая функция возвращала значение вложенной функции.

### 3. Что подразумевает под собой область видимости Local?

Эту область видимости имеют переменные, которые создаются и используются внутри функций.

### 4. Что подразумевает под собой область видимости Enclosing?

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?

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

### 6. Что подразумевает под собой область видимости Built-in?

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?

Пример:

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

Использование: mul(5)(2) или new\_mul5 = mul(5), new\_mul5(2)

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

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. Это свойство позволяет строить иерархические структуры данных. Покажем это на примере кортежей в Python:

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

```
a = tpl(1, 2)
```

```
(1, 2)
```

```
b = tpl(3, a)
```

```
(3, (1, 2))
```

```
c = tpl(a, b)
```

```
((1, 2), (3, (1, 2)))
```