

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

Кафедра инфокоммуникаций

Отчет по лабораторной работе №2.11

Замыкания в языке Python

по дисциплине «Технологии программирования и алгоритмизация»

Выполнил

студент группы ИВТ-б-о-20-1

Дыбов Д.В. « » _____ 20__ г.

Работа защищена « » _____ 20__ г.

Проверил Воронкин Р.А. _____

(подпись)

Ставрополь 2021

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

Ход работы

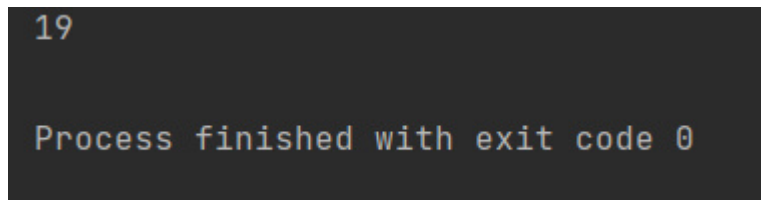
1. Создал новый репозиторий для лабораторной работы №2.11;
2. Клонировал созданный репозиторий на компьютер;
3. Создал новый PyCharm проект в папке репозитория;
4. Проработал первый пример;
5. Проверил пример на работоспособность:



```
x = 2
7
```

Рисунок 1 – Результат выполнения первого примера

6. Проработал второй пример;
7. Проверил пример на работоспособность:

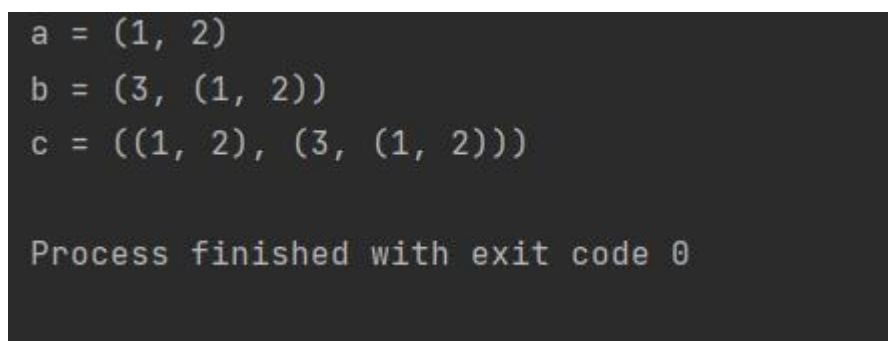


```
19

Process finished with exit code 0
```

Рисунок 2 – Результат выполнения второго примера

8. Проработал третий пример;
9. Проверил на работоспособность:



```
a = (1, 2)
b = (3, (1, 2))
c = ((1, 2), (3, (1, 2)))

Process finished with exit code 0
```

Рисунок 3 – Результат выполнения третьего примера

10. Выполнил индивидуальное задание;
11. Проверил индивидуальное задание на работоспособность:

```
Напишите строки через пробел: 1 2 3 4 5 6 7
<ol>

<li> строка 1</li>

<li> строка 2</li>

<li> строка 3</li>

<li> строка 4</li>

<li> строка 5</li>

<li> строка 6</li>

<li> строка 7</li>

</ol>
```

Рисунок 4 – Результат выполнения индивидуального задания

12. С помощью сайта проверил первый пример на наличие ошибок;
13. Результат не выдал ошибок;

Python code

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

No syntax errors detected :)

Рисунок 5 – Проверка кода на наличие ошибок

14. С помощью сайта проверил второй пример на наличие ошибок;
15. Результат не выдал ошибок;

Python code

```
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    return fun2
12
13
```

No syntax errors detected :)

Рисунок 6 – Проверка второго примера на наличие ошибок

16. С помощью сайта проверил третий пример на наличие ошибок;
17. Результат не выдал ошибок;

Python code

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

No syntax errors detected :)

Рисунок 7 – Проверка третий пример на наличие ошибок

18. С помощью сайта проверил индивидуальное задание на наличие ошибок;

19. Результат не выдал ошибок;

Python code

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4
5 def string():
6     def sophia():
7         a = list(map(int, input("Напишите строки через пробел: ").split()))
8         print("<ol>\n")
9
10        for i in a:
11            print(f"<li> строка {i}</li>\n")
12
13        print("</ol>")
```

No syntax errors detected :)

Рисунок 8 – Проверка индивидуального задания на наличие ошибок

20. Отправил все изменения на репозиторий.

Контрольные вопросы

1. Что такое замыкание?

Замыкание 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?

Каждый раз, когда мы вызываем функцию, у нее создаются локальные переменные, а после завершения – уничтожаются. Можно сделать так, чтобы после завершения работы функции, часть локальных переменных не уничтожалась, а сохраняла свои значение до следующего запуска.

Локальная переменная не будет уничтожена, если на нее где-то останется “живая” ссылка, после завершения работы функции. Эту ссылку может сохранять вложенная функция. Функции, построенные по такому

принципу могут использоваться для построения специализированных функций, т.е. являются фабриками функций. Замыкания как раз и используют эту идею.

3. Что подразумевает под собой область видимости Local?

Локальная область видимости – идентификатор доступен только внутри определённой функции (процедуры). Видимость в пределах модуля может существовать в модульных программах, состоящих из нескольких отдельных фрагментов кода, обычно находящихся в разных файлах.

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 её 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 Build-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?

Вызывая функцию, мы фактически обращаемся к вложенной функции. Переменная внешней функции, является в ней локальной и имеет область enclosing во вложенной функции. Несмотря на то, что внешняя функция завершает свою работу, переменная не уничтожается, т.к. на нее сохраняется ссылка во внутренней функции, которая была возвращена в качестве результата

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.

Вывод: в ходе выполнения лабораторной работы были приобретены навыки по работе с замыканиями при написании программ с помощью языка программирования Python версии 3.x.