

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 №1

**дисциплины «Основы кроссплатформенного
программирования»**

Выполнил:

Боженко Александр Иванович

1 курс, группа ИТС-б-о-21-1,

11.03.02 «Инфокоммуникационные
технологии и системы связи»,

направленность (профиль)

«Инфокоммуникационные системы и сети»,

очная форма обучения

(подпись)

Руководитель практики:

Воронкин Р.А, канд. техн. наук, доцент
кафедры инфокоммуникаций

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2022 г.

Тема: Замыкания в языке Python

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

Ход работы:

Задание 6

Используя замыкания функций, объявите внутреннюю функцию, которая бы все повторяющиеся символы заменяла одним другим указанным символом. Какие повторяющиеся символы искать и на что заменять, определяются параметрами внешней функции. Внутренней функции передается только строка для преобразования. Преобразованная (сформированная) строка должна возвращаться внутренней функцией. Вызовите внутреннюю функцию замыкания и отобразите на экране результат ее работы.

```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sys
5
6  def fun1(to_replace, replacer):
7
8      def fun2(string):
9          nonlocal to_replace, replacer
10         result = string.replace(replacer, to_replace)
11         return result
12
13     return fun2
14
15
16 if __name__ == "__main__":
17     x = input("Введите строку: ")
18     c = input("Введите символ, который нужно заменить: ")
19     h = input("Введите символ, на который заменить: ")
20     rep = fun1(h, c)
21     print(rep(x))
```

Рис 1. Код программы

```
Введите строку: 1234*6789
Введите символ, который нужно заменить: *
Введите символ, на который заменить: 5
123456789
```

Рис 2. Результат программы

Вопросы:

1. Что такое замыкание?

Замыкание (closure) в программировании – это функция, в теле которого присутствуют ссылки на переменные, объявленные вне тела этой функции в окружающем коде и не являющейся ее параметрами. Обычно, по области видимости, переменные делят на глобальные и локальные.

2. Как реализованы замыкания в языке программирования Python?

```
def outer():  
    x = 1  
    def inner():  
        print(f'x in outer function: {x}')    return inner
```

Функция `outer` определяется с функцией `inner` внутри, а функция `outer` возвращает функцию `inner`; именно она – возвращаемое значение `outer`. Здесь вложенная функция – это и есть замыкание.

3. Что подразумевает под собой область видимости Local?

Область видимости `Local` имеют переменные, которые создаются и используются внутри функции.

4. Что подразумевает под собой область видимости Enclosing?

Суть области видимости `Enclosing` в том, что внутри функции могут быть вложенные функции и локальные переменные, так вот локальная

5. Что подразумевает под собой область видимости Global?

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*?

Область видимости *Built-in* 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*?

```
>>> new_mul5 = mul(5)
>>> new_mul5
<function mul.<locals>.helper at 0x000001A7548C1158>
>>> new_mul5(2)
10
>>> new_mul5(7)
35
```

Вызывая *new_mul5(2)*, мы фактически обращаемся к функции *helper()*, которая находится внутри *mul()*. Переменная *a*, является локальной для *mul()*, и имеет область *enclosing* в *helper()*. Несмотря на то, что *mul()* завершила свою работу, переменная *a* 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?

Перейдем с уровня математики на уровень функционального программирования. Вот как определяется “свойство замыкания” в книге “Структура и интерпретация компьютерных программ” Айбельсона Х., Сассмана Д. Д.: “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 было изучены Замыкания в пайтоне, а также мы приобрели навыки по работе с замыканиями при написании программ с помощью пайтон.