

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ ФГАОУ ВО «СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ
УНИВЕРСИТЕТ»

Институт цифрового развития

Кафедра инфокоммуникаций

Дисциплина: «языки программирования»

ОТЧЕТ

по лабораторной работе №1

Выполнил: студент 2 курса группы ИТС-21-1

Снадный Михаил Сергеевич

Проверил: к.ф.-м.н., доцент кафедры инфокоммуникаций

Воронкин Роман Александрович

Работа защищена с оценкой: _____

Ставрополь, 2021

Тема:

Замыкания в языке Python

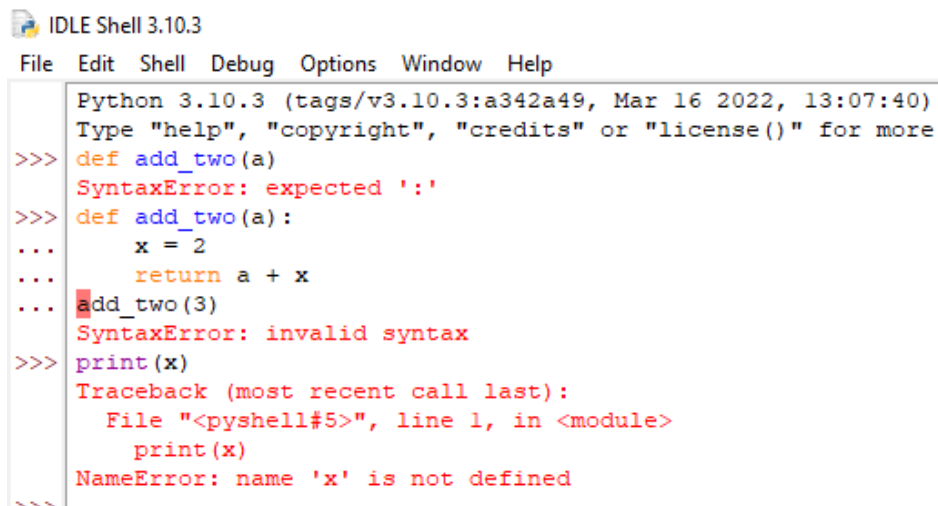
Цель работы:

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

Порядок выполнения работы:

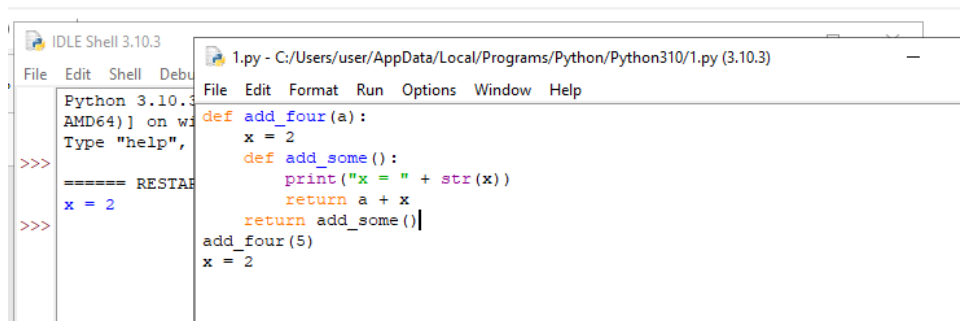
- 1) Создадим общедоступный репозиторий на GitHub (<https://github.com/peach909/11.git>)
- 2) Решим задачи с помощью языка программирования Python3. И отправим их на GitHub.

Проработал пример 1



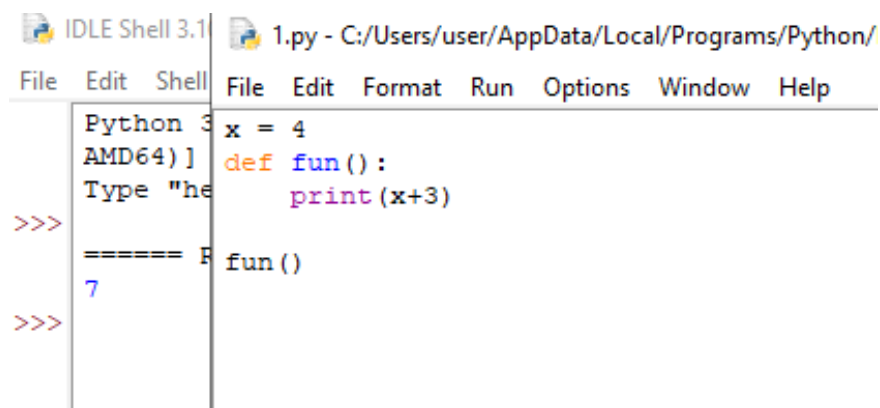
```
Python 3.10.3 (tags/v3.10.3:a342a49, Mar 16 2022, 13:07:40)
Type "help", "copyright", "credits" or "license()" for more
>>> def add_two(a)
SyntaxError: expected ':'
>>> def add_two(a):
...     x = 2
...     return a + x
... add_two(3)
SyntaxError: invalid syntax
>>> print(x)
Traceback (most recent call last):
  File "<pyshell#5>", line 1, in <module>
    print(x)
NameError: name 'x' is not defined
>>>
```

Проработал пример 2



```
Python 3.10.3 (tags/v3.10.3:a342a49, Mar 16 2022, 13:07:40)
Type "help", "copyright", "credits" or "license()" for more
>>>
===== RESTART: This prompt will execute from the top of the input.
>>> x = 2
>>>
def add_four(a):
    x = 2
    def add_some():
        print("x = " + str(x))
    return a + x
    return add_some()
add_four(5)
x = 2
```

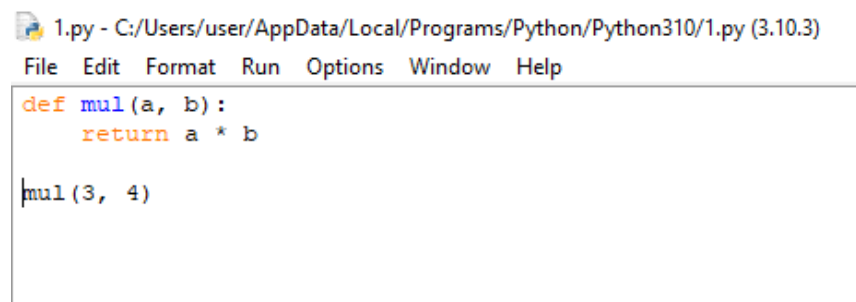
Проработал пример 3



```
Python 3.10.3 Shell
AMD64)
Type "help()" for help
>>>
=====
7
>>>
```

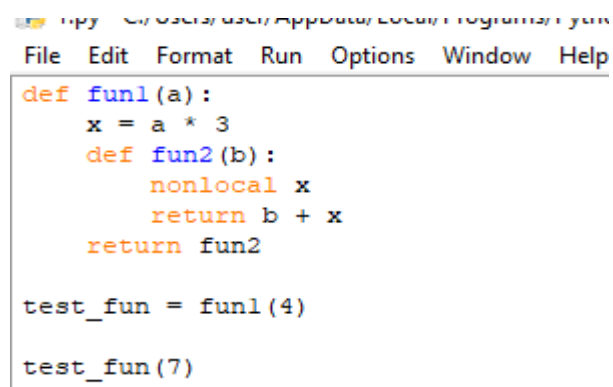
```
1.py - C:/Users/user/AppData/Local/Programs/Python/Python310/1.py (3.10.3)
File Edit Format Run Options Window Help
x = 4
def fun():
    print(x+3)
fun()
```

Проработал пример 4



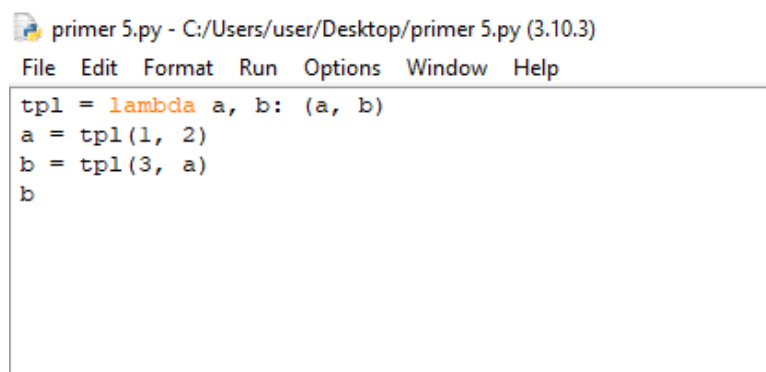
```
1.py - C:/Users/user/AppData/Local/Programs/Python/Python310/1.py (3.10.3)
File Edit Format Run Options Window Help
def mul(a, b):
    return a * b
mul(3, 4)
```

Проработал пример 5



```
1.py - C:/Users/user/AppData/Local/Programs/Python/Python310/1.py (3.10.3)
File Edit Format Run Options Window Help
def fun1(a):
    x = a * 3
    def fun2(b):
        nonlocal x
        return b + x
    return fun2
test_fun = fun1(4)
test_fun(7)
```

Проработал пример 6



```
primer 5.py - C:/Users/user/Desktop/primer 5.py (3.10.3)
File Edit Format Run Options Window Help
tpl = lambda a, b: (a, b)
a = tpl(1, 2)
b = tpl(3, a)
b
```

Индивидуальное задание

```
1.py - C:/Users/user/AppData/Local/Programs/Python/Python310/1.py (3.10.3)
File Edit Format Run Options Window Help

import sys
def fun1(to_replace, replacer):
    def fun2(string):
        nonlocal to_replace, replacer
        result = string.replace(replacer, to_replace)
        return result
    return fun2
if __name__ == "__main__":
    x = input(" llllll ")
    c = input(" ll ")
    h = input(" 2 ")
    rep = fun1(h, c)
    print(rep(x))
```

Контрольные 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?

В Python, выделяют четыре области видимости для переменных

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

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?

Функция `mul()` умножает два числа и возвращает полученный результат. Если мы ходим на базе нее решить задачу: “умножить число на пять”, то в самом простом случае, можно вызывать `mul()`, передавая в качестве первого аргумента пятерку.

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

множество натуральных чисел замкнуто относительно операции сложения – какие бы натуральные числа мы не складывали, получим натуральное число, но это множество не замкнуто относительно операции вычитания. Это свойство позволяет строить иерархические структуры данных.

Вывод:

приобрел навыки по работе с замыканиями при написании программ с помощью языка программирования Python версии 3.x