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:
Шальнев Владимир Сергеевич,
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:

```
F:\pythonProject>git clone https://github.com/HAXF13D/laboratory-14
Cloning into 'laboratory-14'...
remote: Enumerating objects: 11, done.
remote: Counting objects: 100% (11/11), done.
remote: Compressing objects: 100% (10/10), done.
remote: Total 11 (delta 2), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (11/11), done.
```

Клонирование репозитория

```
>>> def add_two(a):
...     x = 2
...     return a + x
...
>>> add_two(3)
5
>>> x
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'x' is not defined
>>>
```

Пример 1

```
>>> def add_four(a):
...     x = 2
...     def add_some():
...             print("x = " + str(x))
...             return a + x
...     return add_some()
...
>>> add_four(5)
x = 2
7
>>>
```

Пример 2

```
>>> x = 4
>>> def fun():
...     print(x+3)
...
>>> fun()
7
>>>
```

Пример 3

```
>>> def mul(a, b):
...     return a * b
...
>>> mul(3, 4)
12
```

Пример 4

```
>>> def mul(a, b):
...     return a * b
...
>>> mul(3, 4)
12
>>> def mul5(a):
...     return mul(5, a)
...
>>> mul5(2)
10
>>> def mul(a):
...     def helper(b):
...             return a * b
...     return helper
...
>>> mul(5)(2)
10
>>> new_mul5 = mul(5)
>>> new_mul5
<function mul.<locals>.helper at 0x000001B4143654C0>
>>> new_mul5(2)
10
>>>
```

Пример 5

```
>>> def fun1(a):
...     x = a * 3
...     def fun2(b):
...             nonlocal x
...             return b + x
...     return fun2
...
>>> test_fun = fun1(4)
>>> test_fun(7)
19
>>>
```

Пример 6

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-


def individual_func(string):

    def replace_string(name, surname):
        nonlocal string
        answer = string.replace("%N%", name)
        answer = answer.replace("%F%", surname)
        return answer
    return replace_string


if __name__ == "__main__":
    replacer = individual_func("Уважаемый %F%, %N%! Вы делаете работу по замыканиям функций.")
    print(replacer('Vladimir', 'Shalnev'))
    print(replacer('Dinis', 'Proverkovich'))
    replacer = individual_func("Уважаемый %F%, %N%!")
    print(replacer('Nestikus', 'Dimov'))

```

Индивидуальное задание 1

```
Уважаемый Shalnev, Vladimir! Вы делаете работу по замыканиям функций.
Уважаемый Proverkovich, Dinis! Вы делаете работу по замыканиям функций.
Уважаемый Dimov, Nestikus!
```

Значение №1

```
F:\pythonProject\laboratory-14>git add .

F:\pythonProject\laboratory-14>git commit -m "Add indiv task"
[main 96d6582] Add indiv task
 1 file changed, 21 insertions(+)
 create mode 100644 tasks/indiv.py

F:\pythonProject\laboratory-14>
```

Коммит изменений

ОТВЕТЫ НА ВОПРОСЫ:

1. Что такое замыкание?

Замыкание — 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?

- У нас должна быть вложенная функция (функция внутри функции).
- Вложенная функция должна ссылаться на значение, определенное в объемлющей функции.

- Объемлющая функция должна возвращать вложенную функцию.

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 Build-in?

Уровень Python интерпретатора. В рамках этой области видимости находятся функции open, len и т. п., также туда входят исключения.

7. Как использовать замыкания в языке программирования Python?

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

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”