МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития

Кафедра информационных систем и технологий

Отчет по лабораторной работе №14.

Дисциплина: «Основы программной инженерии»

Выполнил:

Студент группы ПИЖ-б-о-22-1,

направление подготовки: 09.03.04

«Программная инженерия»

ФИО: Джараян Арег Александрович

Проверил:

Воронкин Р. А.

Ставрополь 2022

Тема: Лабораторная работа 2.11 Замыкания в языке Python.

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

Выполнение работы:

1. Изучил теоретический материал работы.

2. Создал репозиторий на git.hub.

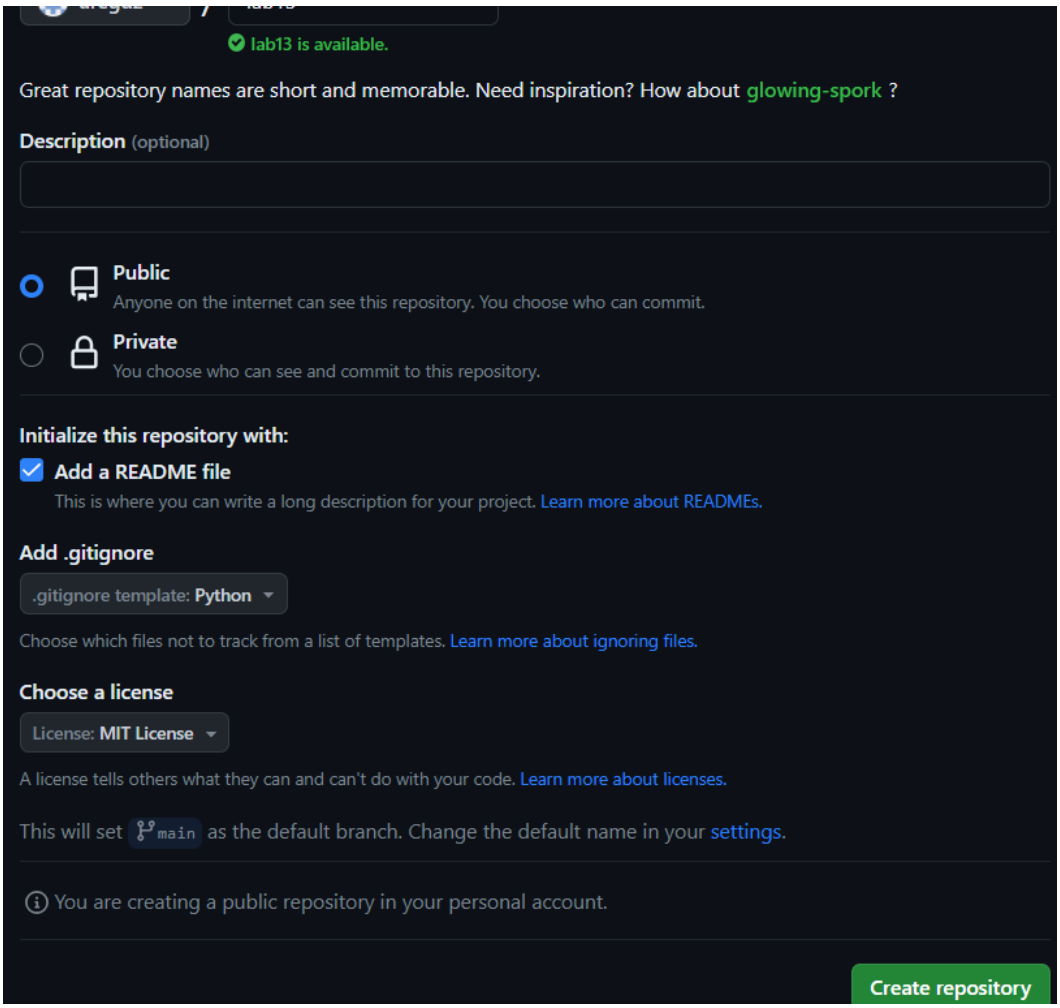

Рисунок 1 – создание репозитория

3. Клонировал репозиторий.

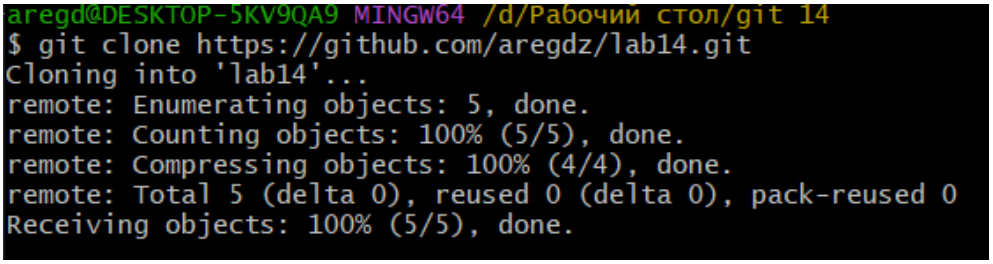

Рисунок 2 – клонирование репозитория 4.

Дополнить файл gitignore необходимыми правилами.

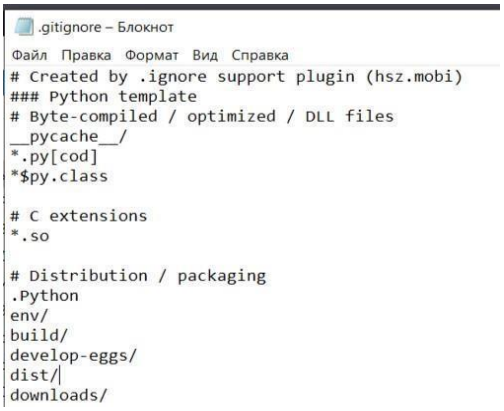

Рисунок 3 – .gitignore для IDE PyCharm

4.Организовать свой репозиторий в соответствии с моделью ветвления git-flow.

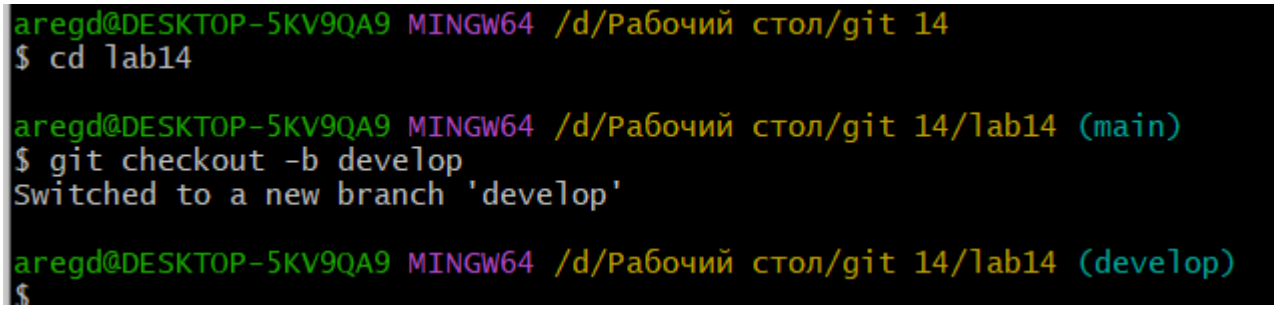

Рисунок 4 – создание ветки develop

5. Проработал примеры из методички.

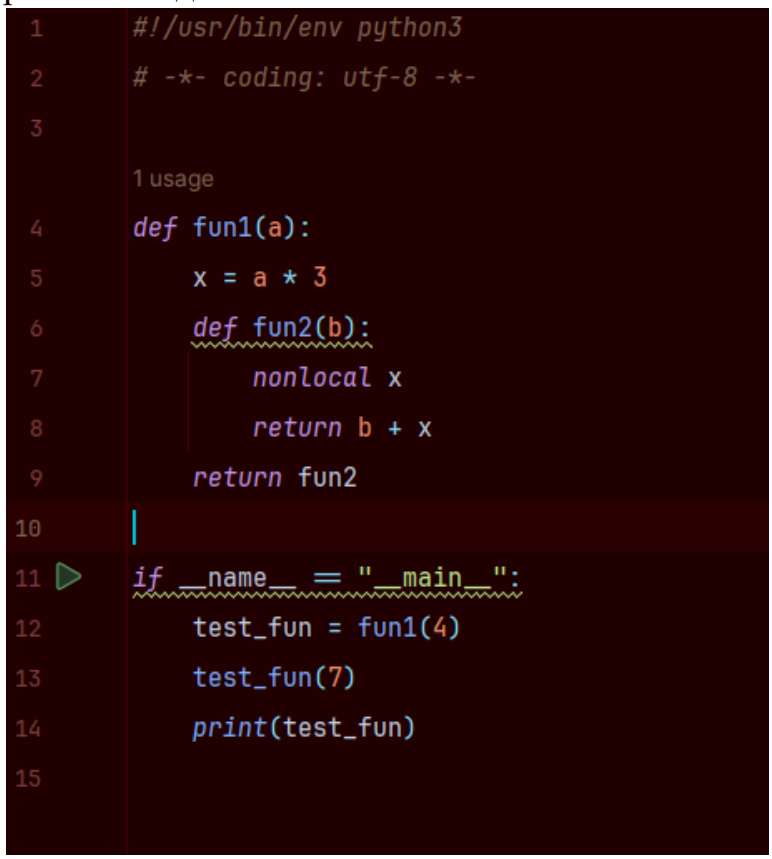

Рисунок 5 – пример 1

6. Используя замыкания функций, объявите внутреннюю функцию, которая принимает в качестве параметров фамилию и имя, а затем, заносит в шаблон эти данные. Сам шаблон – это строка, которая передается внешней функции и, например, может иметь такой вид: «Уважаемый %F%, %N%! Вы делаете работу по замыканиям функций.» Здесь %F% - это фрагмент куда нужно подставить фамилию, а %N% - фрагмент, куда нужно подставить имя. (Шаблон может быть и другим, вы это определяете сами). Здесь важно, чтобы внутренняя функция умела подставлять данные в шаблон, формировать новую строку и возвращать результат. Вызовите внутреннюю функцию замыкания и отобразите на экране результат ее работы.

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

1 usage
def create_greeting_template(template):
    def inner_function(last_name, first_name):
        formatted_template = template.replace('%F%', last_name).replace('%N%', first_name)
        return formatted_template
    return inner_function

if __name__ == "__main__":
    n = input("Введите вашу фамилию: ")
    l = input("Введите ваше имя: ")

    # Создаем замыкание с шаблоном
    greeting_template = create_greeting_template("Уважаемый %F%, %N%! Вы делаете работу по замыканиям функций.")

    # Вызываем внутреннюю функцию замыкания и отображаем результат
    result = greeting_template(n, l)
    print(result)
```

Рисунок 6 – индивидуальное заадние

```
C:\Users\aregd\AppData\Local\Programs\Python\Python311\python.exe "D:\Рабочи
Введите вашу фамилию: Джараян
Введите ваше имя: Арег
Уважаемый Джараян , Арег! Вы делаете работу по замыканиям функций.

Process finished with exit code 0
```

Рисунок 7 – индивидуальное задние

7.Зафиксировал все изменения в github в ветке develop.

```
aregd@DESKTOP-5KV9QA9 MINGW64 /d/Рабочий стол/git 14/lab14 (develop)
$ git add .

aregd@DESKTOP-5KV9QA9 MINGW64 /d/Рабочий стол/git 14/lab14 (develop)
$ git commit -m"индивид. и пример"
[develop acd51f5] индивид. и пример
 2 files changed, 33 insertions(+)
 create mode 100644 PyCharm/individual.py
 create mode 100644 PyCharm/primer.py
```

Рисунок 8 – фиксация изменений в ветку develop

8.Слил ветки.

```
aregd@DESKTOP-5KV9QA9 MINGW64 /d/Рабочий стол/git 14/lab14 (develop)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

aregd@DESKTOP-5KV9QA9 MINGW64 /d/Рабочий стол/git 14/lab14 (main)
$ git merge develop
Updating 9b69efd..acd51f5
Fast-forward
 PyCharm/individual.py | 19 ++++++++++++++++++
 PyCharm/primer.py     | 14 ++++++++++++++
 2 files changed, 33 insertions(+)
 create mode 100644 PyCharm/individual.py
 create mode 100644 PyCharm/primer.py
```

Рисунок 9 – сливание ветки develop в ветку main

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?

Замыкание (closure) в Python - это функция, которая запоминает значения в окружающей (внешней) области видимости, даже если эта область видимости больше не существует. Замыкание возникает, когда внутри функции определены вложенные функции, и вложенная функция ссылается на переменные из внешней 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 Build-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?

Использование замыканий в Python обычно связано с созданием функций внутри других функций и возвратом этих вложенных функций. Замыкания полезны, когда вам нужно передать часть контекста (переменные) в функцию, чтобы она могла использовать их даже после того, как внешняя функция завершила свою работу.

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

Замыкания в Python можно использовать для построения иерархических данных, таких как деревья или вложенные структуры. Замыкания позволяют сохранять состояние внешней функции внутри вложенной функции, что обеспечивает уровень иерархии.