

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ
ФЕДЕРАЦИИ**
**Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРОКАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

Кафедра инфокоммуникаций

Институт цифрового развития

ОТЧЁТ

по лабораторной работе №2.11

Дисциплина: «Программирование на Python»

Тема:

«Замыкания в языке Python Python»

Выполнил: студент 2 курса
группы ИВТ-б-о-21-1

Богдашов Артём
Владимирович

Ставрополь 2022

Выполнение работы:

1. Создал репозиторий в GitHub «rep 2.11» в который добавил .gitignore, который дополнил правила для работы с IDE PyCharm с ЯП Python, выбрал лицензию MIT, клонировал его на лок. сервер и организовал в соответствии с моделью ветвления git-flow.

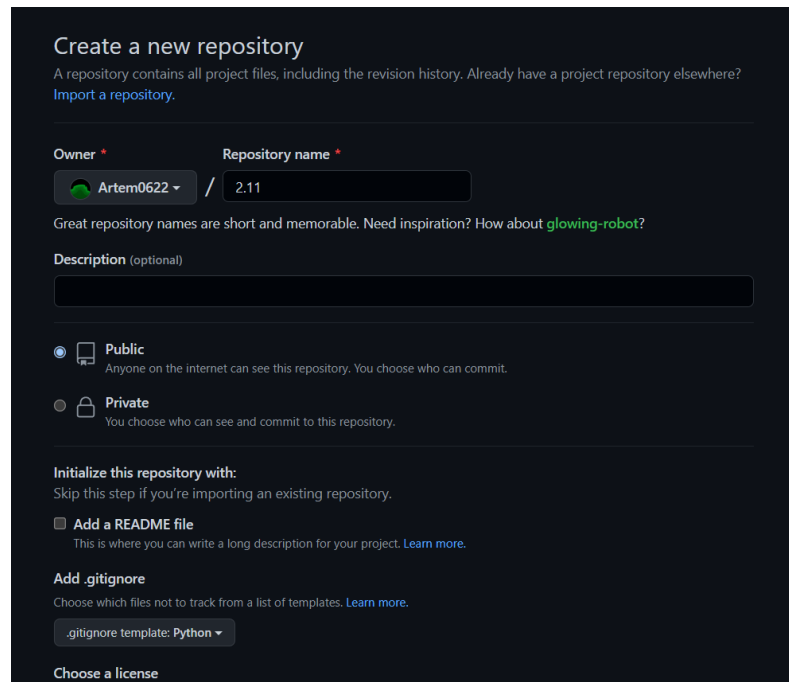


Рисунок 1.1 Создание репозитория

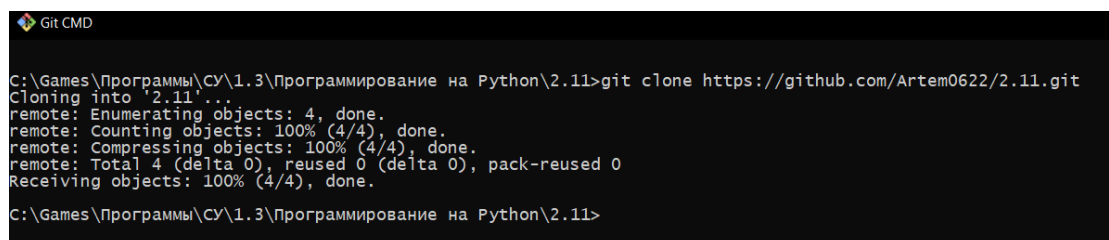


Рисунок 1.2 Клонирование репозитория

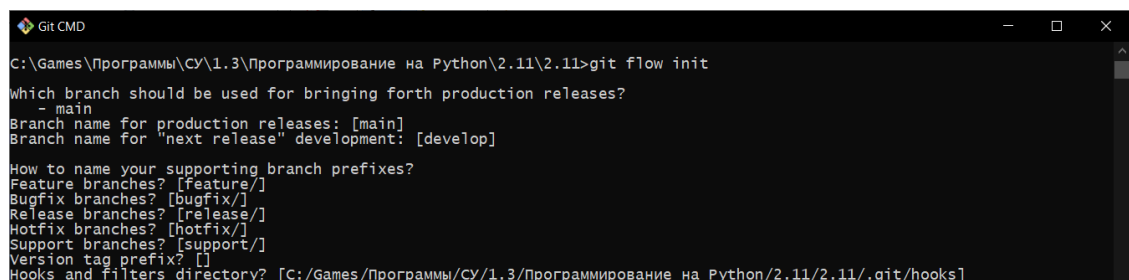


Рисунок 1.3 Организация репозитория в соответствии с моделью ветвления
git-flow

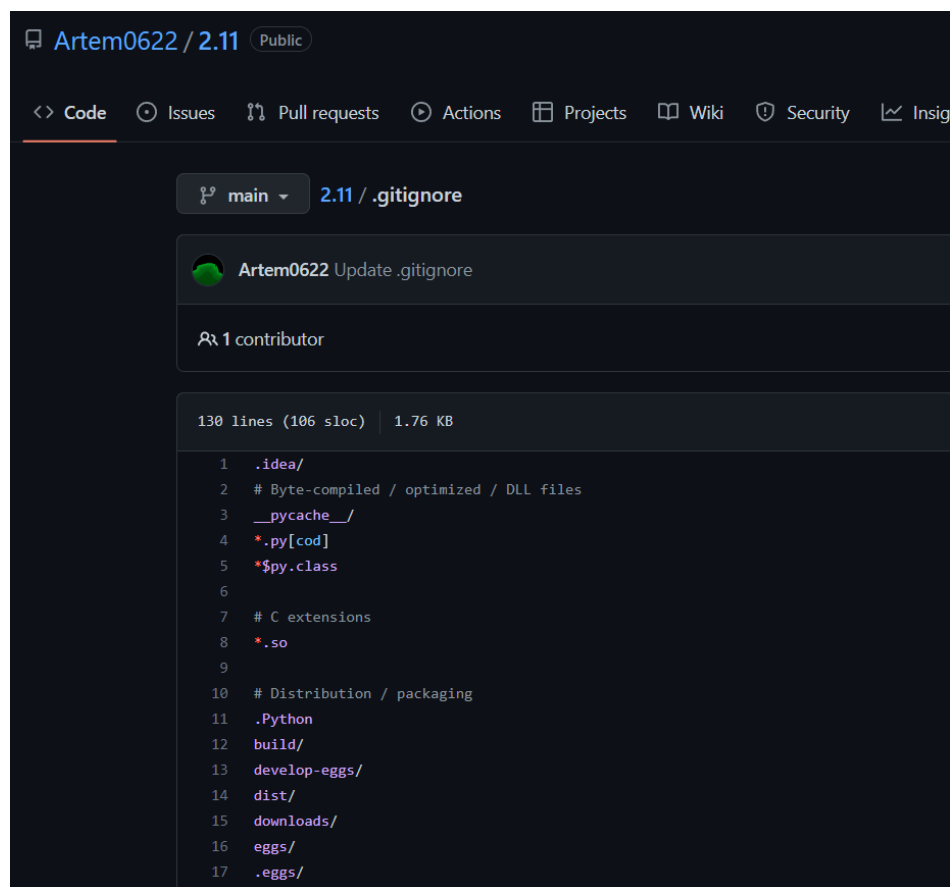
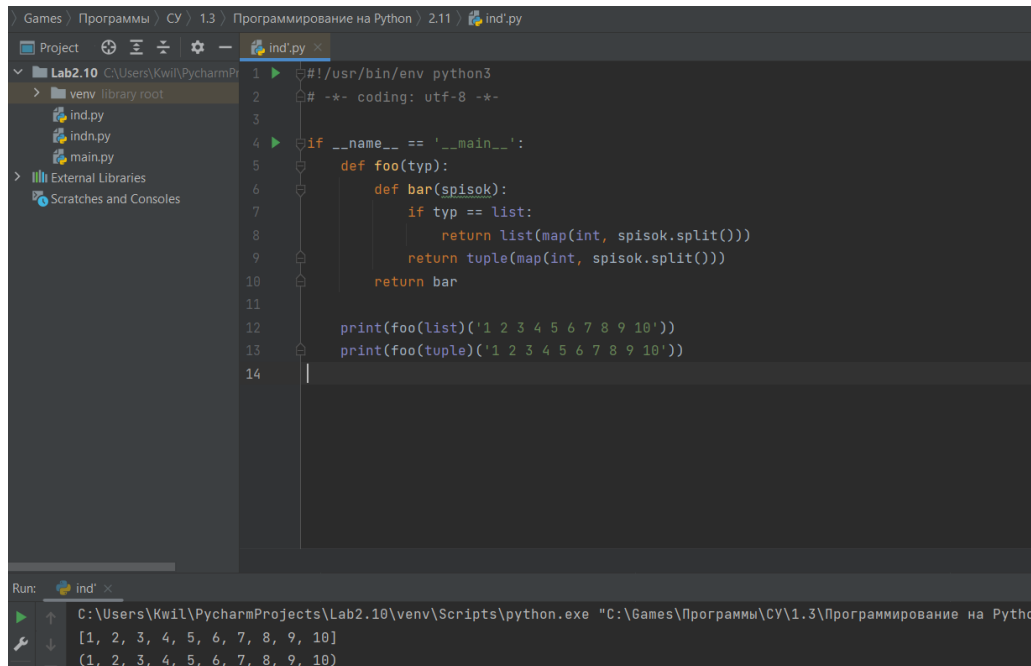


Рисунок 1.4 Изменение .gitignore

3.(3 вариант). Выполнил индивидуальное задание.

3. Используя замыкания функций, объявите внутреннюю функцию, которая преобразует строку из списка целых чисел, записанных через пробел, либо в список, либо в кортеж. Тип коллекции определяется параметром `type` внешней функции. Если `type = 'list'`, то используется список, иначе – кортеж. Далее, на вход программы поступает две строки: первая – это значение для параметра `type`; вторая – список целых чисел, записанных через пробел. С помощью реализованного замыкания преобразовать эту строку в соответствующую коллекцию. Результат работы замыкания выведите на экран.



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

if __name__ == '__main__':
    def foo(typ):
        def bar(spisok):
            if typ == list:
                return list(map(int, spisok.split()))
            return tuple(map(int, spisok.split()))
        return bar

    print(foo(list)('1 2 3 4 5 6 7 8 9 10'))
    print(foo(tuple)('1 2 3 4 5 6 7 8 9 10'))
```

Run: ind' x

```
C:\Users\Kwll\PycharmProjects\Lab2.10\venv\Scripts\python.exe "C:\Games\Программы\СУ\1.3\Программирование на Python\2.11\2.11>git add .
C:\Games\Программы\СУ\1.3\Программирование на Python\2.11\2.11>git commit -m "save ind"
[develop af0ddd] save ind
1 file changed, 13 insertions(+)
create mode 100644 ind'.py
C:\Games\Программы\СУ\1.3\Программирование на Python\2.11\2.11>
```

Рисунок 2.1 Вывод программы индивидуального задания

```
C:\Games\Программы\СУ\1.3\Программирование на Python\2.11\2.11>git add .
C:\Games\Программы\СУ\1.3\Программирование на Python\2.11\2.11>git commit -m "save ind"
[develop af0ddd] save ind
1 file changed, 13 insertions(+)
create mode 100644 ind'.py
C:\Games\Программы\СУ\1.3\Программирование на Python\2.11\2.11>
```

Рисунок 3.1 Коммит изменений

```
C:\Games\Программы\СУ\1.3\Программирование на Python\2.11\2.11>git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
```

Рисунок 3.2 Переход на ветку main

```
C:\Games\Программы\СУ\1.3\Программирование на Python\2.11\2.11>git merge develop
Updating c8475df..af0ddd
Fast-forward
 ind'.py | 13 ++++++++
1 file changed, 13 insertions(+)
create mode 100644 ind'.py
```

Рисунок 3.3 Слияние ветки main с develop

```
C:\Games\Программы\СУ\1.3\Программирование на Python\2.11\2.11>git push
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 12 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (5/5), 818 bytes | 818.00 KiB/s, done.
Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/Artem0622/2.11.git
 c3eae10..f2995b5 main -> main
```

Рисунок 3.4 Пуш изменений

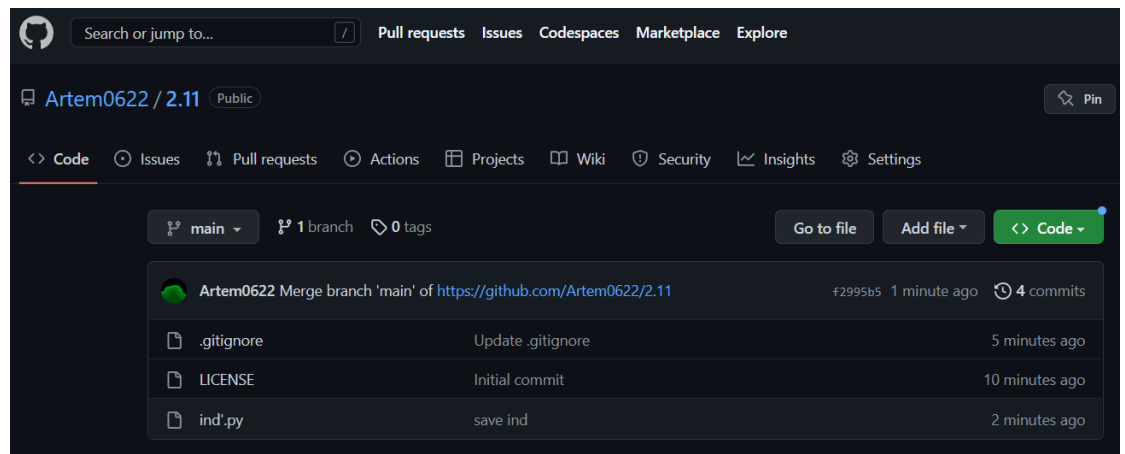


Рисунок 3.5 Изменение на уд сервере

Ответы на 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: local, enclosing, global, build-in.

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.

1. Что подразумевает под собой область видимости Global?

Переменные области видимости `global` – это глобальные переменные уровня модуля (модуль – это файл с расширением `.py`).

2. Что подразумевает под собой 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.

3. 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.

