

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 №14**  
**дисциплины «Основы программной инженерии»**

Выполнил:  
Матвеев Александр Иванович  
2 курс, группа ПИЖ-б-о-22-1,  
09.03.04 «Программная инженерия»,  
направленность (профиль) «Разработка и  
сопровождение программного  
обеспечения», очная форма обучения

\_\_\_\_\_  
(подпись)

Проверил Воронкин Роман Александрович

\_\_\_\_\_  
(подпись)

Отчет защищен с оценкой \_\_\_\_\_ Дата защиты \_\_\_\_\_

Ставрополь, 2023 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.

1. Создание нового репозитория с лицензией MIT.

**Create a new repository**

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

*Required fields are marked with an asterisk (\*).*

**Owner \*** SashkaHacker / **Repository name \*** laba10  
✔ laba10 is available.

Great repository names are short and memorable. Need inspiration? How about [fluffy-guide](#) ?

**Description** (optional)

☒ **Public**  
Anyone on the internet can see this repository. You choose who can commit.

☐ **Private**  
You choose who can see and commit to this repository.

**Initialize this repository with:**

☐ **Add a README file**  
This is where you can write a long description for your project. [Learn more about READMEs.](#)

**Add .gitignore**  
.gitignore template: Python

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

**Choose a license**  
License: MIT License

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

ⓘ You are creating a public repository in your personal account.

**Create repository**

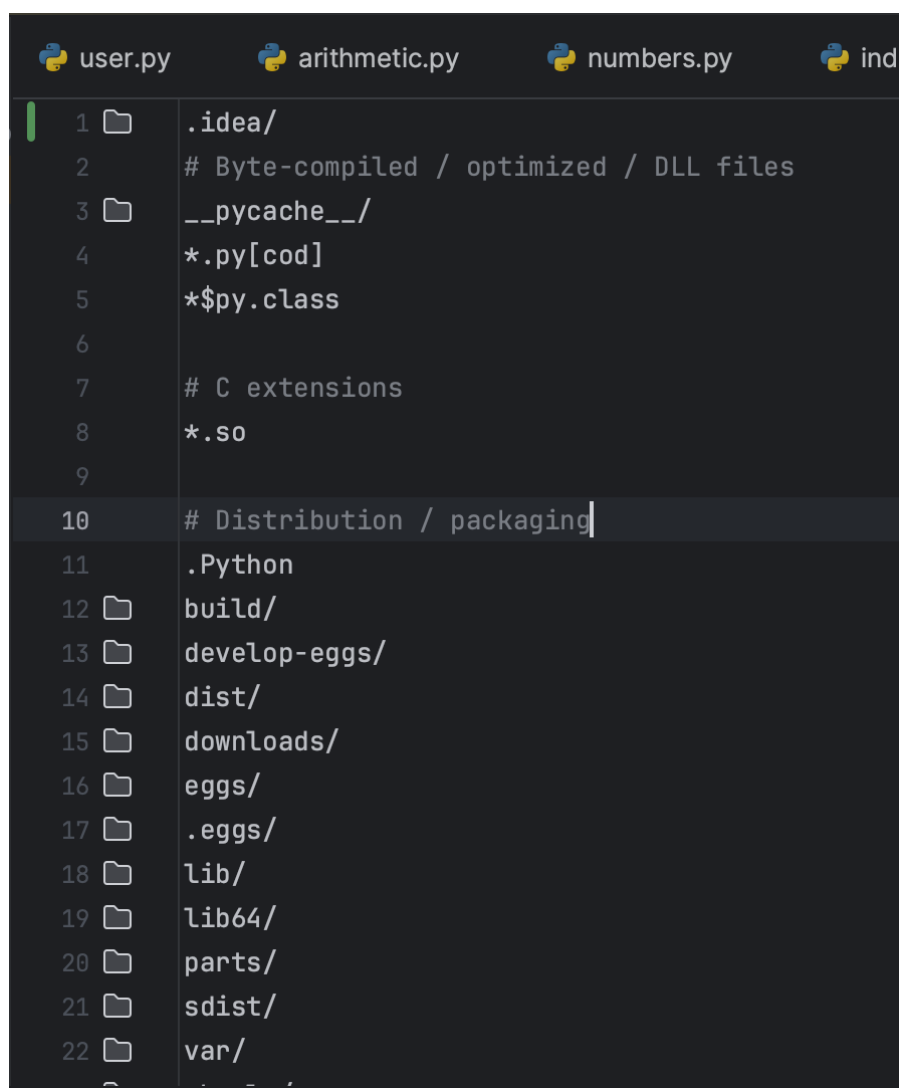
Рисунок 1 – Создание репозитория

2. Клонировал репозиторий на рабочий ПК.

```
Sashka@DESKTOP-U4RPSBI MINGW64 ~/Documents/GitHub
$ git clone https://github.com/SashkaHacker/lab10.git
Cloning into 'lab10'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (4/4), done.
```

Рисунок 2 – Клонирование репозитория

3. Дополнил файл .gitignore необходимыми инструкциями.



The screenshot shows a code editor with a dark theme. At the top, there are four tabs: 'user.py', 'arithmetic.py', 'numbers.py', and 'ind'. The active tab is 'user.py'. The editor displays the content of a '.gitignore' file. The file contains the following text:

```
1  .idea/
2  # Byte-compiled / optimized / DLL files
3  __pycache__/
4  *.py[cod]
5  *$py.class
6
7  # C extensions
8  *.so
9
10 # Distribution / packaging
11 .Python
12 build/
13 develop-eggs/
14 dist/
15 downloads/
16 eggs/
17 .eggs/
18 lib/
19 lib64/
20 parts/
21 sdist/
22 var/
```

Рисунок 3 – Файл .gitignore

4. Проработка примеров из лабораторных задач.

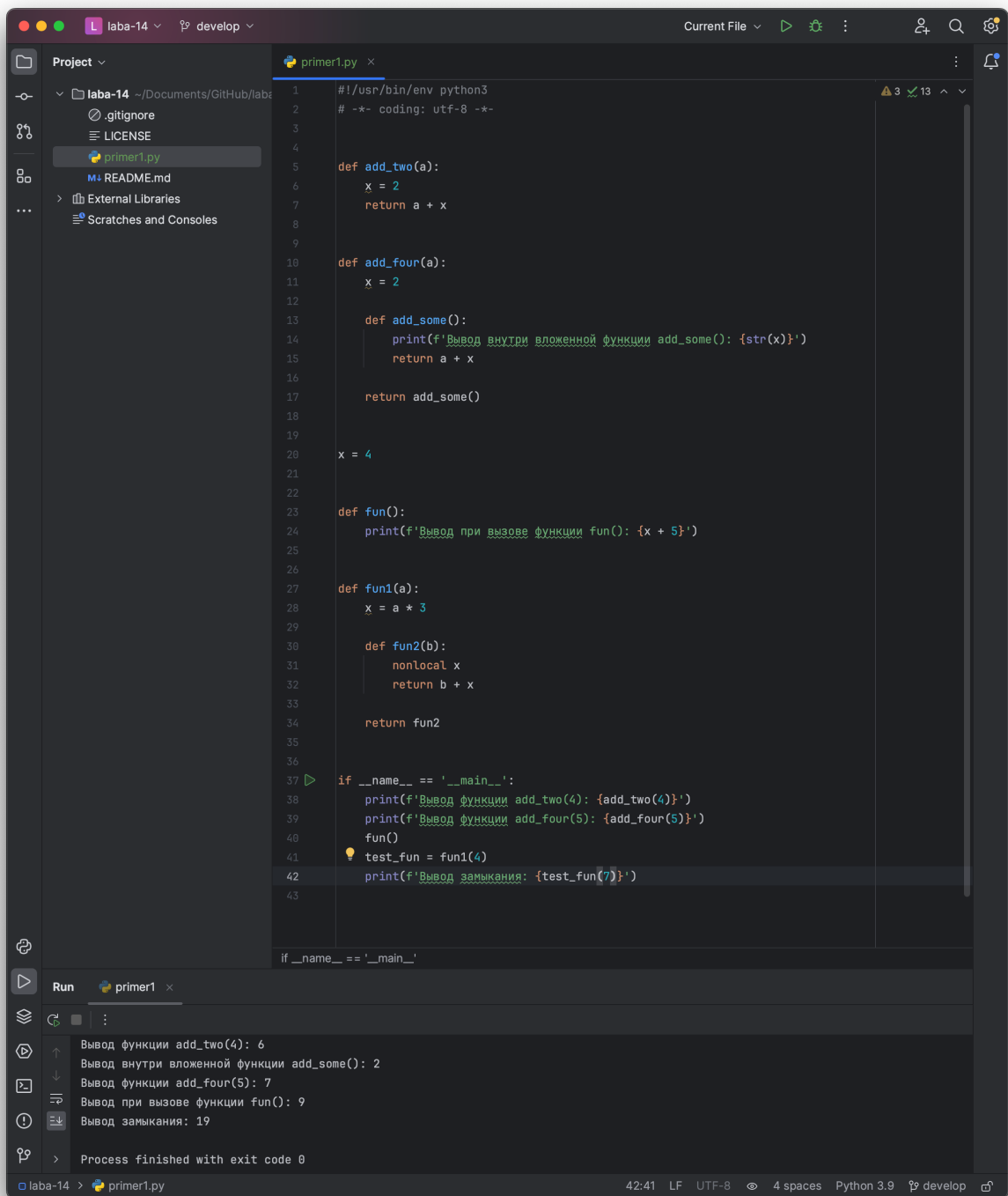


Рисунок 4 – Примеры

## 5. Выполнение индивидуального задания. (Вариант - 11)

Условие задачи: Используя замыкания функций, определите вложенную функцию, которая бы увеличивала значение переданного параметра на 3 и возвращала бы вычисленный результат. Вызовите внешнюю функцию для получения ссылки на внутреннюю функцию и присвойте ее переменной с

именем `cnt`. Затем, вызовите внутреннюю функцию через переменную `cnt` со значением `k`, введенным с клавиатуры.

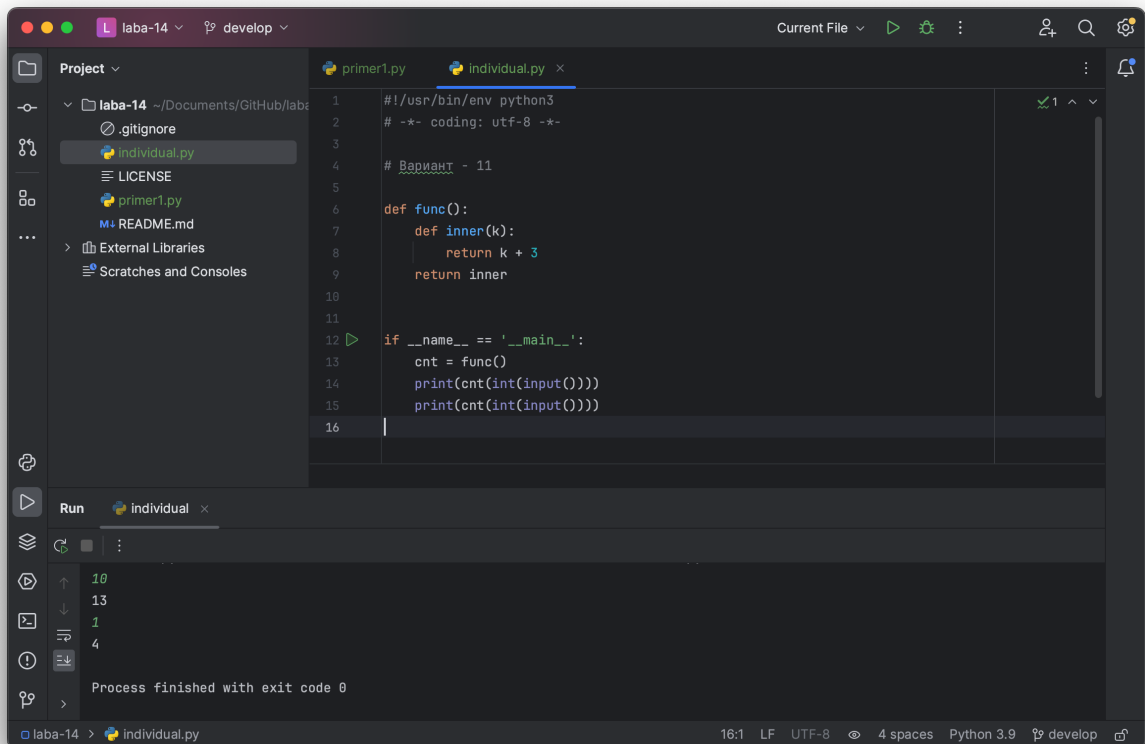


Рисунок 5 – Индивидуальное задание

Контрольные вопросы:

1. Замыкание, это функция, определенная внутри другой функции, которая имеет ссылки на переменные, лежащие в области `enclosing`.
2. В python замыкание реализуется следующим образом:

```
def names():
    all_names = []

    def inner(name: str) -> list:
        all_names.append(name)
        return all_names

    return inner # ссылка на функцию

if __name__ == '__main__':
    students = names() # создание объекта функции
    print(students('Vasya'))
    print(students('Ivan'))
    print(students('Sasha'))

# -> ['Vasya', 'Ivan', 'Sasha']
```

Рисунок 6 – Реализация замыкания

3. Local область видимости в Python подразумевает, что имя переменной определено внутри блока кода функции, то есть имеет локальную область видимости.
4. Enclosing область видимости в Python подразумевает, что переменная находится внутри внешних функций.
5. Global область видимости в Python подразумевает, что переменная, объявленная вне функции или в глобальной области видимости, называется глобальной переменной.
6. Build-in область видимости в Python подразумевает, что переменная находится во встроенной области видимости, которая содержит зарезервированные значения Python.
7. Замыкания в Python можно использовать для следующих целей: сохранение состояния между вызовами функций; инкапсуляции данных; построение иерархических данных.
8. Замыкания в Python могут быть использованы для построения иерархических данных. Это свойство замыкания позволяет создавать вложенные структуры данных, используя внутренние функции и переменные внешней функции.