

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙ-
СКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

Кафедра инфокоммуникаций

Основы кроссплатформенного программирования

Отчет по лабораторной работе №2.13

Тема: «Модули и пакеты»

Выполнил студент группы

ИВТ-б-о-21-1

Криворот В.Г. « » _____ 20__ г.

Подпись студента _____

Работа защищена « » _____ 20__ г.

Проверил доцент

Кафедры инфокоммуникаций, старший
преподаватель

Воронкин Р.А.

(подпись)

Ставрополь 2022

Цель работы: приобретение навыков по работе с модулями и пакетами языка программирования Python версии 3.x.

Ход работы:

1. Создал репозиторий в GitHub, дополнил правила в .gitignore для работы с IDE PyCharm с ЯП Python, выбрал лицензию MIT, клонировал его на компьютер и организовал в соответствии с моделью ветвления git-flow.

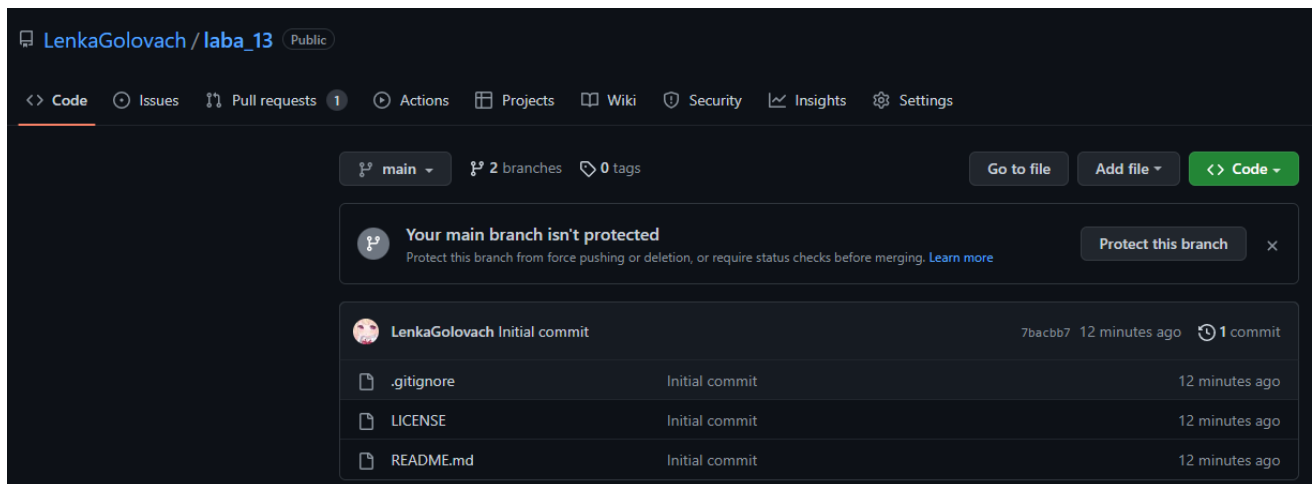


Рисунок 1.1 – Созданный репозиторий

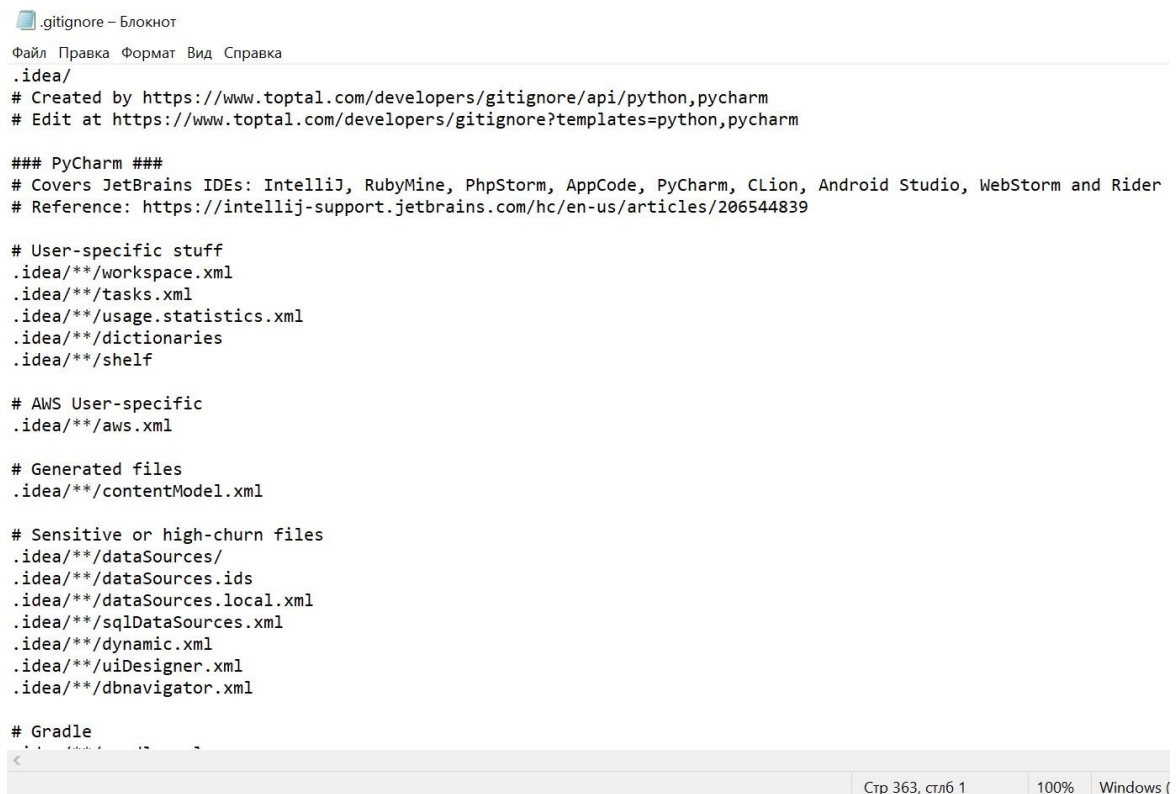


Рисунок 1.2 – Дополнил правила в .gitignore

```

c:\Users\Admin\Desktop\git\Python8>git flow init

Which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [C:/Users/Admin/Desktop/git/Python8/.git/hooks]

c:\Users\Admin\Desktop\git\Python8>

```

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

2. Создал проект Pycharm в папке репозитория, проработал примеры ЛР.

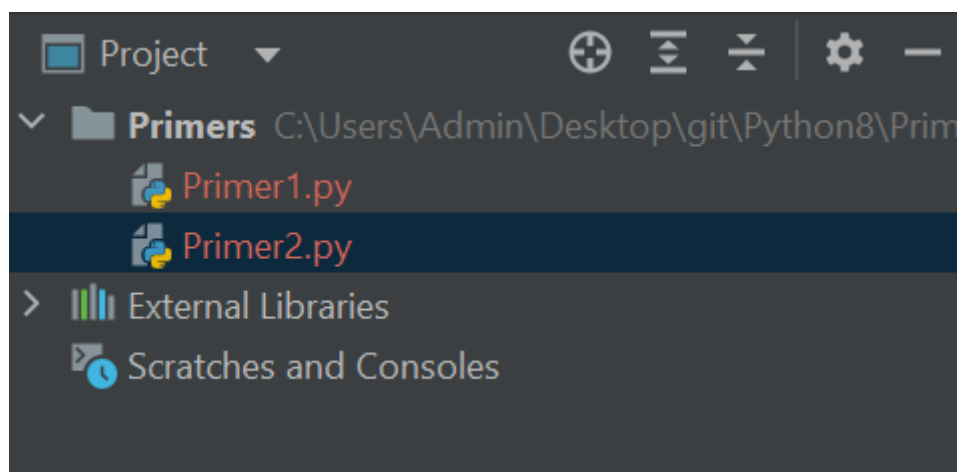


Рисунок 2.1 – Созданные проекты

```

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

from math import cos

if __name__ == "__main__":
    print(cos(3.14))

```

Рисунок 2.2 – Пример №1

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

from math import factorial as f

if __name__ == "__main__":
    print(f(4))
```

Primer2 x

C:\Users\Admin\AppData\Local\Program

24

Рисунок 2.3 – Пример №2

3. Индивидуальные задания. В – 1.

Задание 1. Выполнить индивидуальное задание лабораторной работы 2.11, оформив все функции программы в виде отдельного модуля. Разработанный модуль должен быть подключен в основную программу с помощью одного из вариантов команды `import`. Номер варианта уточнить у преподавателя.

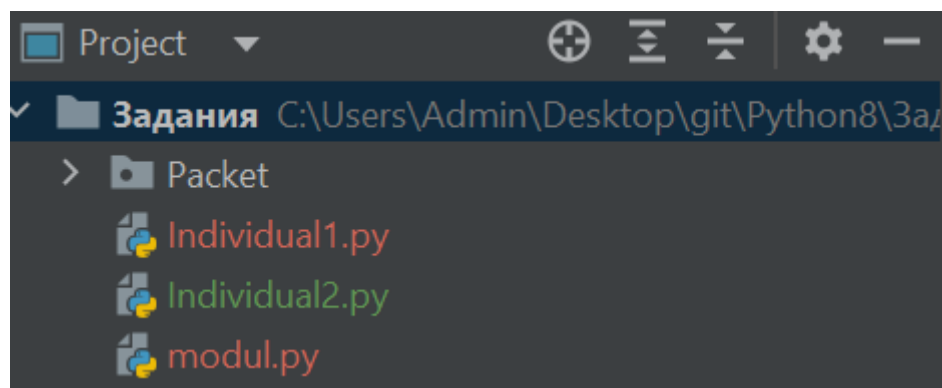
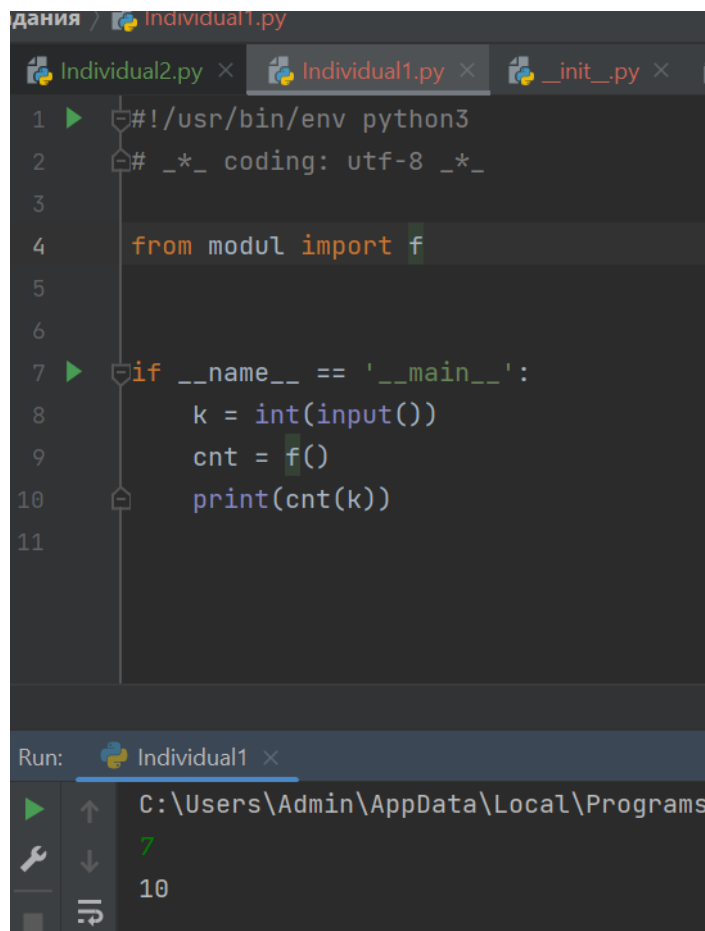


Рисунок 3.1 – Созданные проекты



```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 from modul import f
5
6
7 if __name__ == '__main__':
8     k = int(input())
9     cnt = f()
10    print(cnt(k))
11
```

Run: Individual1

C:\Users\Admin\AppData\Local\Programs

7

10

Рисунок 3.2 – Результат выполнения программы

Задание 2. Выполнить индивидуальное задание лабораторной работы 2.8, оформив все классы программы в виде отдельного пакета. Разработанный пакет должен быть подключен в основную программу с помощью одного из вариантов команды `import`. Настроить соответствующим образом переменную `__all__` в файле `__init__.py` пакета. Номер варианта уточнить у преподавателя.

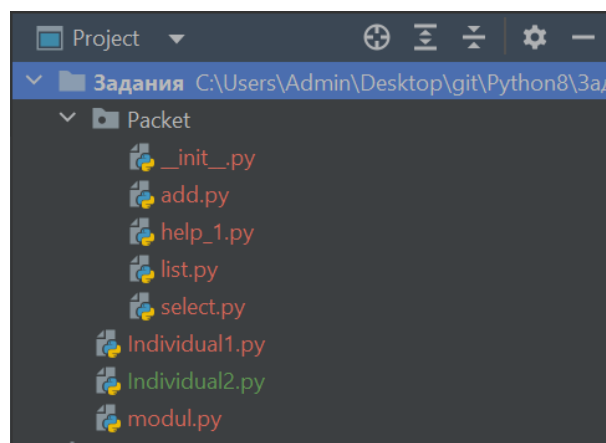


Рисунок 3.3 – Созданные проекты

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

import sys
from Packet.add import add
from Packet.list import listt
from Packet.select import selectt
from Packet.help_1 import help_1

def main():
    # Список студентов.
    students = []

    main() > while True > if command == 'exit'
```

Individual1 x

```
>>> listt
```

№	Ф.И.О.	Группа	Успеваемость
1	Богда ВА	2	3.3.4.2.1
2	Бракуенко	4	5.5.5.5.5

```
>>>
```

Рисунок 3.4 – Результат выполнения программы

Вывод: в результате выполнения лабораторной работы были приобретены теоретические знания и практические навыки по работе с модулями и пакетами языка программирования Python версии 3.x.

Ответы на контрольные вопросы:

1. Что является модулем языка Python?

Под модулем в Python понимается файл с расширением .py.

Модули предназначены для того, чтобы в них хранить часто используемые функции, классы, константы и т. п. Можно условно разделить модули и программы: программы предназначены для непосредственного запуска, а модули для импортирования их в другие программы.

2. Какие существуют способы подключения модулей в языке Python?

За один раз можно импортировать сразу несколько модулей, для этого их нужно перечислить через запятую после слова `import`.

Если вы хотите задать псевдоним для модуля в вашей программе, можно воспользоваться вот таким синтаксисом:

```
import имя_модуля
```

```
import имя_модуля1, имя_модуля2
```

Используя любой из вышеперечисленных подходов, при вызове функции из импортированного модуля, вам всегда придется указывать имя модуля (или псевдоним). Для того, чтобы этого избежать делайте импорт через конструкцию `from ... import`.

Для импортирования нескольких функций из модуля, можно перечислить их имена через запятую.

Импортируемому объекту можно задать псевдоним. `import имя_модуля as новое_имя`.

3. Что является пакетом языка Python?

Пакет в Python – это каталог, включающий в себя другие каталоги и модули, но при этом дополнительно содержащий файл `__init__.py`. Пакеты используются для формирования пространства имен, что позволяет работать с модулями через указание уровня вложенности (через точку).

4. Каково назначение файла `__init__.py`?

Файл `__init__.py` нужен для объявления структуры пакета.

5. Каково назначение переменной `__all__` файла `__init__.py`?

В переменную `__all__` вносятся все модули пакета.