

**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 №2.13

Дисциплина: «Программирование на Python»

Тема:

«Модули и пакеты»

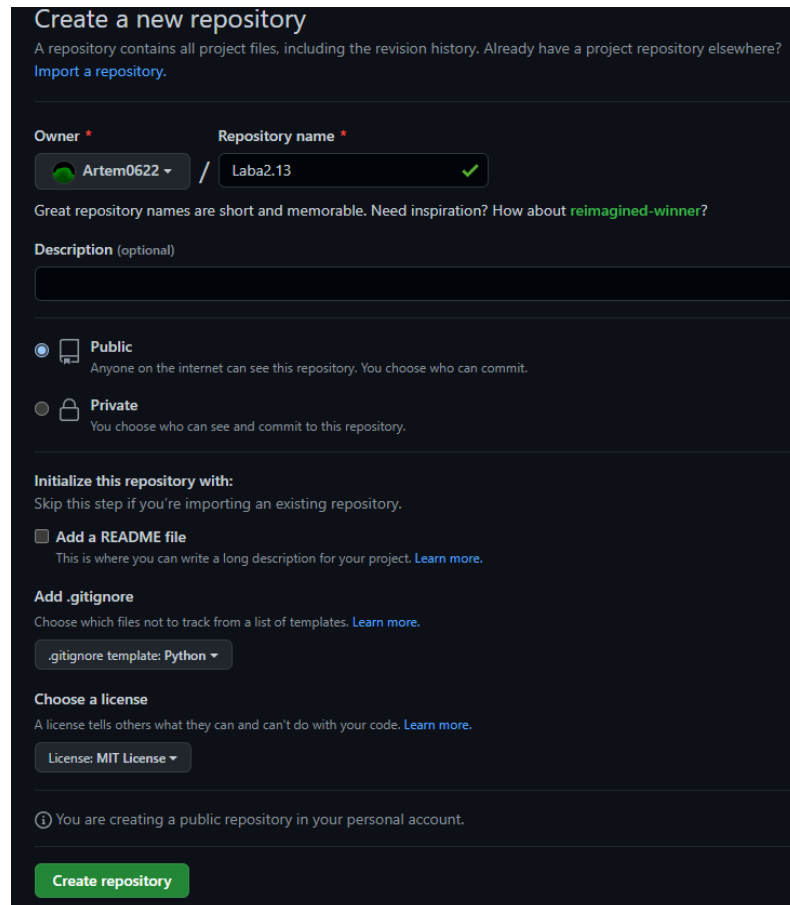
Выполнил: студент 2 курса
группы ИВТ-б-о-21-1

Богдашов Артём
Владимирович

Ставрополь 2022

Выполнение работы:

1. Создал репозиторий в GitHub «rep Laba2.13» 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 смоделью ветвления git-flow.



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

Owner * **Repository name ***
Artem0622 / Laba2.13 ✓

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

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:
Skip this step if you're importing an existing repository.

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

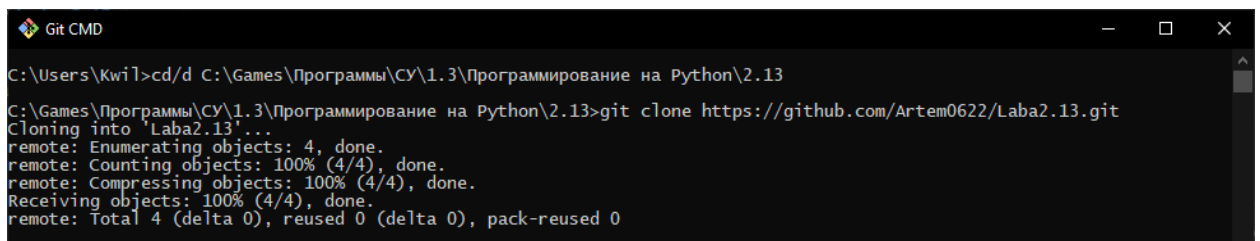
Add .gitignore
Choose which files not to track from a list of templates. [Learn more.](#)
.gitignore template: Python

Choose a license
A license tells others what they can and can't do with your code. [Learn more.](#)
License: MIT License

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

Create repository

Рисунок 1.1 Создание репозитория



```
Git CMD
C:\Users\Kwirl>cd/d C:\Games\Программы\СУ\1.3\Программирование на Python\2.13
C:\Games\Программы\СУ\1.3\Программирование на Python\2.13>git clone https://github.com/Artem0622/Laba2.13.git
Cloning into 'Laba2.13'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
Receiving objects: 100% (4/4), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
```

Рисунок 1.2 Клонирование репозитория

```

C:\Games\Программы\СУ\1.3\Программирование на Python\2.13\Laba2.13>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:/Games/Программы/СУ/1.3/Программирование на Python/2.13/Laba2.13/.git/hooks]

```

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

```

130 lines (106 sloc) | 1.76 KB
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/

```

Рисунок 1.4 Изменение .gitignore

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

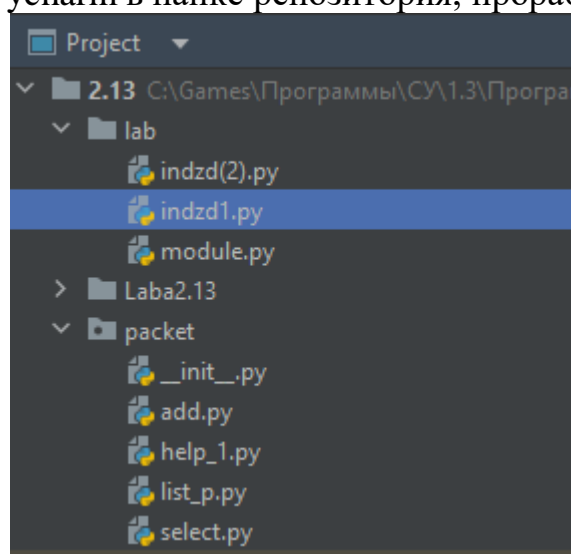


Рисунок 2.1 – Созданные проекты

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

from math import cos

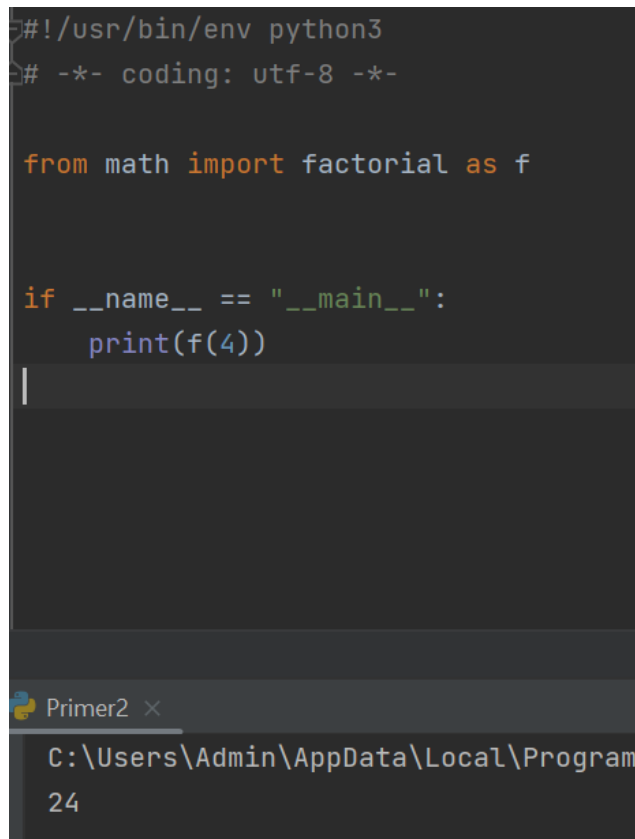
if __name__ == "__main__":
    print(cos(3.14))
```

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

2. Индивидуальные задания. Вариант 3.

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.

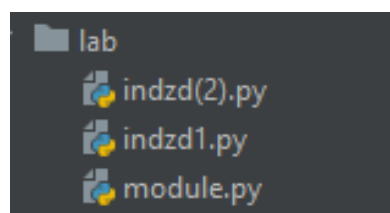


Рисунок 3.1 – Созданные проекты

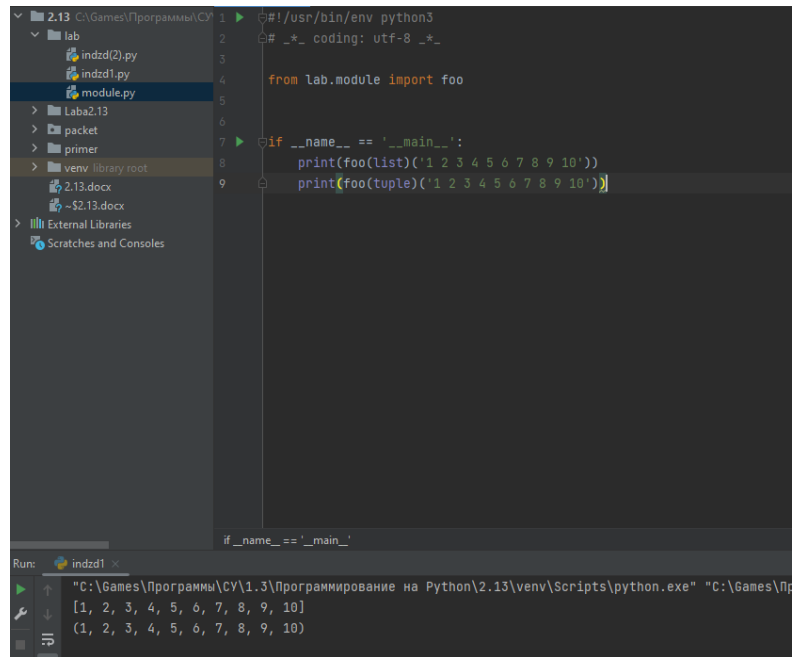


Рисунок 3.2 – Результат выполнения программы

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.

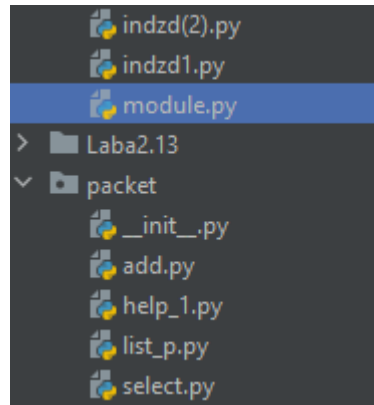


Рисунок 3.3 – Созданные проекты

```

1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import ...
5
6
7
8
9
10 if __name__ == '__main__':
11     # Список студентов.
12     students = []
13     # Организовать бесконечный цикл запроса команд.
14     while True:
15         # Запросить команду из терминала.
16         command = input(">>> ").lower()
17         # Выполнить действие в соответствие с командой.
18         if command == 'exit':
19             break
20         elif command == 'add':
21             students = add(students)
22         elif command == 'list':
23             list_p(students)
24         elif command == 'select':
25             select(command, students)
26         elif command == 'help':
27             help_d()
28         else:
29             print(f"Неизвестная команда {command}", file=sys.stderr)

```

Run: indzd(2) ×

Фамилия и инициалы? Гуляницкий
 Номер группы? ИТГ
 Успеваемость? 5 4 3 4 4
 >>> add
 Фамилия и инициалы? Гуляницкий
 Номер группы? ИТГ
 Успеваемость? 5 3 3 3 4
 >>> list

№	Ф.И.О.	Номер группы	Успеваемость
1	Гуляницкий	ИТГ	5 2 2 3 4
2	Иванов Иван	ИТГ	5 4 3 4 4

Рисунок 3.4 – Результат выполнения программы

```

C:\Games\Программы\СУ\1.3\Программирование на Python\2.13\Laba2.13>git add .
C:\Games\Программы\СУ\1.3\Программирование на Python\2.13\Laba2.13>git commit -m"indzd"
[develop 4668d5b] indzd
10 files changed, 151 insertions(+)
create mode 100644 lab/indzd(2).py
create mode 100644 lab/indzd1.py
create mode 100644 lab/module.py
create mode 100644 lab/packet/__init__.py
create mode 100644 lab/packet/add.py
create mode 100644 lab/packet/help_1.py
create mode 100644 lab/packet/list_p.py
create mode 100644 lab/packet/select.py
create mode 100644 primer/primer1.py
create mode 100644 primer/primer2.py

```

Рисунок 4.1 Коммит изменений

```

C:\Games\Программы\СУ\1.3\Программирование на Python\2.13\Laba2.13>git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

```

Рисунок 4.2 Переход на ветку main

```
C:\Games\Программы\СУ\1.3\Программирование на Python\2.13\Laba2.13>git merge develop
Updating b84b459..4668d5b
Fast-forward
 lab/indzd(2).py | 29 ++++++
 lab/indzd1.py   | 9 ++++++
 lab/module.py   | 11 ++++++
 lab/packet/__init__.py | 4 +++++
 lab/packet/add.py | 21 ++++++
 lab/packet/help_1.py | 11 ++++++
 lab/packet/list_p.py | 32 ++++++
 lab/packet/select.py | 18 ++++++
 primer/primer1.py | 8 ++++++
 primer/primer2.py | 8 ++++++
10 files changed, 151 insertions(+)
create mode 100644 lab/indzd(2).py
create mode 100644 lab/indzd1.py
create mode 100644 lab/module.py
create mode 100644 lab/packet/__init__.py
create mode 100644 lab/packet/add.py
create mode 100644 lab/packet/help_1.py
create mode 100644 lab/packet/list_p.py
create mode 100644 lab/packet/select.py
create mode 100644 primer/primer1.py
create mode 100644 primer/primer2.py
```

Рисунок 4.3 Слияние ветки main с develop

```
C:\Games\Программы\СУ\1.3\Программирование на Python\2.13\Laba2.13>git push
Enumerating objects: 19, done.
Counting objects: 100% (19/19), done.
Delta compression using up to 12 threads
Compressing objects: 100% (17/17), done.
Writing objects: 100% (17/17), 3.50 KiB | 1.17 MiB/s, done.
Total 17 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/Artem0622/Laba2.13.git
 9e31842..066f725 main -> main
```

Рисунок 4.4 Пуш изменений

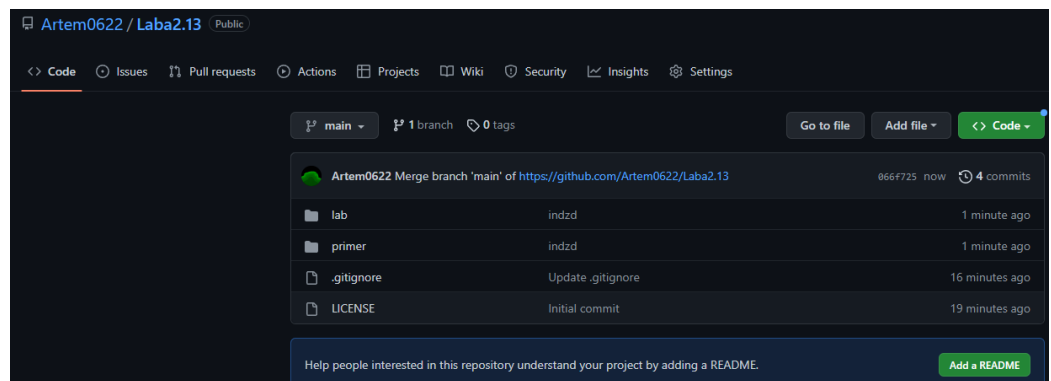


Рисунок 4.5 Изменение на уд сервере

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 мо-дули 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 мо-дули, 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.