

**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 №16.

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 2023

Тема: Лабораторная работа 2.13 Модули и пакеты.

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

Выполнение работы:

1. Изучил теоретический материал работы.
2. Создал репозиторий на git.hub.

**Create a new repository**

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

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

**Owner \*** aregdz / **Repository name \*** 16  
✔ 16 is available.

Great repository names are short and memorable. Need inspiration? How about **refactored-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.](#)

Рисунок 1 – создание репозитория

3. Клонировал репозиторий.

```

aregd@DESKTOP-5KV9QA9 MINGW64 ~
$ cd d:

aregd@DESKTOP-5KV9QA9 MINGW64 /d
$ cd "D:\Рабочий стол\4 семестр\опи\16"

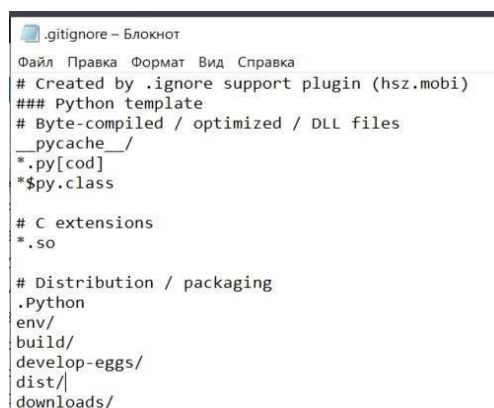
aregd@DESKTOP-5KV9QA9 MINGW64 /d/Рабочий стол/4 семестр/опи/16
$ git clone https://github.com/aregdz/16.git
Cloning into '16'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (4/4), done.

aregd@DESKTOP-5KV9QA9 MINGW64 /d/Рабочий стол/4 семестр/опи/16
$ |

```

Рисунок 2 – клонирование репозитория 4.

Дополнить файл gitignore необходимыми правилами.



```

.gitignore – Блокнот
Файл Правка Формат Вид Справка
# Created by .ignore support plugin (hsz.mobi)
### Python template
# Byte-compiled / optimized / DLL files
__pycache__/
*.py[cod]
*$py.class

# C extensions
*.so

# Distribution / packaging
.Python
env/
build/
develop-eggs/
dist/
downloads/

```

Рисунок 3 – .gitignore для IDE PyCharm

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

```

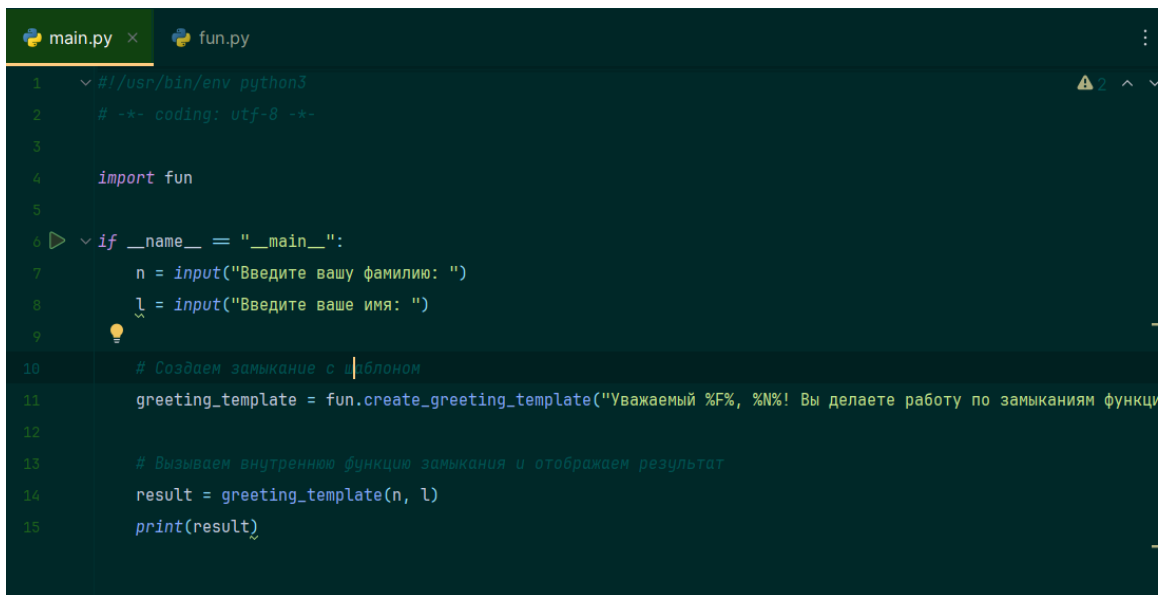
aregd@DESKTOP-5KV9QA9 MINGW64 /d/Рабочий стол/4 семестр/опи/16/16 (main)
$ git checkout -b develop
Switched to a new branch 'develop'

aregd@DESKTOP-5KV9QA9 MINGW64 /d/Рабочий стол/4 семестр/опи/16/16 (develop)
$

```

Рисунок 4 – создание ветки develop

5. 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.



```
main.py x fun.py
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import fun
5
6  if __name__ == "__main__":
7      n = input("Введите вашу фамилию: ")
8      l = input("Введите ваше имя: ")
9
10     # Создаем замыкание с шаблоном
11     greeting_template = fun.create_greeting_template("Уважаемый %F%, %N%! Вы делаете работу по замыканиям функци
12
13     # Вызываем внутреннюю функцию замыкания и отображаем результат
14     result = greeting_template(n, l)
15     print(result)
```

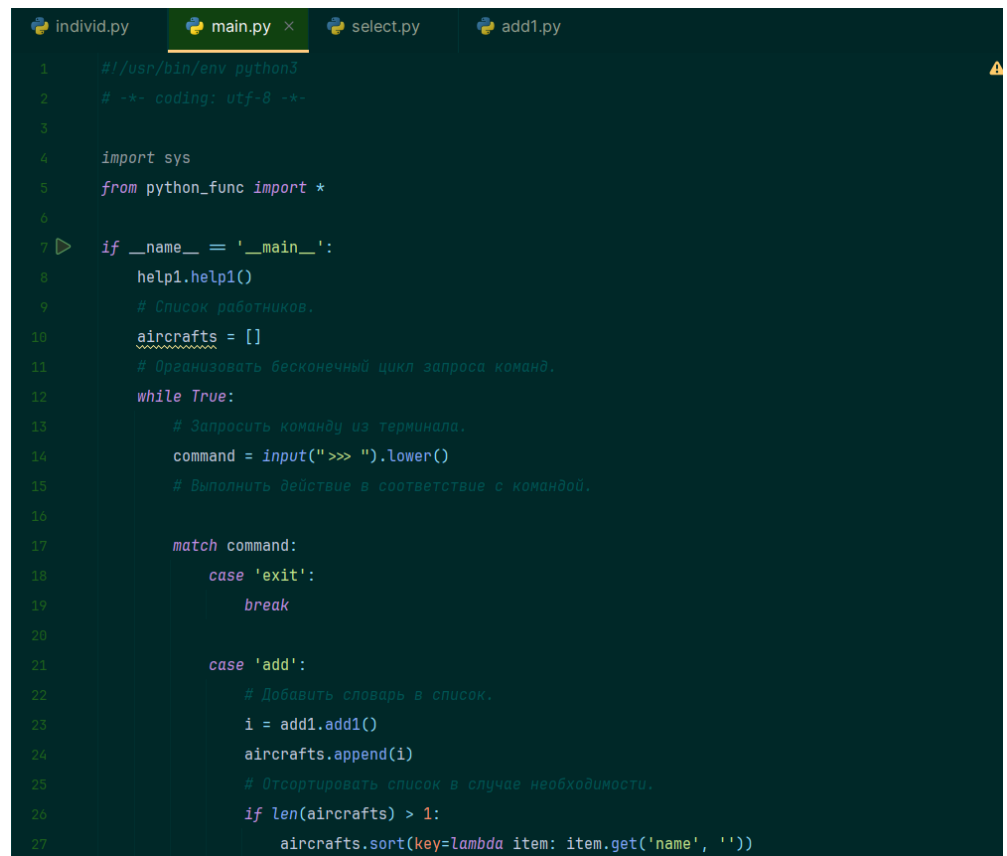
Рисунок 5 – главная часть программы



```
main.py fun.py x
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  def create_greeting_template(template):
5
6      def inner_function(last_name, first_name):
7          formatted_template = template.replace('%F%', last_name).replace('%N%', first_name)
8          return formatted_template
9
10     return inner_function
```

Рисунок 6 – модуль

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



```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sys
5  from python_func import *
6
7  if __name__ == '__main__':
8      help1.help1()
9      # Список работников.
10     aircrafts = []
11     # Организовать бесконечный цикл запроса команд.
12     while True:
13         # Запросить команду из терминала.
14         command = input(">>> ").lower()
15         # Выполнить действие в соответствие с командой.
16
17         match command:
18             case 'exit':
19                 break
20
21             case 'add':
22                 # Добавить словарь в список.
23                 i = add1.add1()
24                 aircrafts.append(i)
25                 # Отсортировать список в случае необходимости.
26                 if len(aircrafts) > 1:
27                     aircrafts.sort(key=lambda item: item.get('name', ''))
```

Рисунок 7 – файл main.py

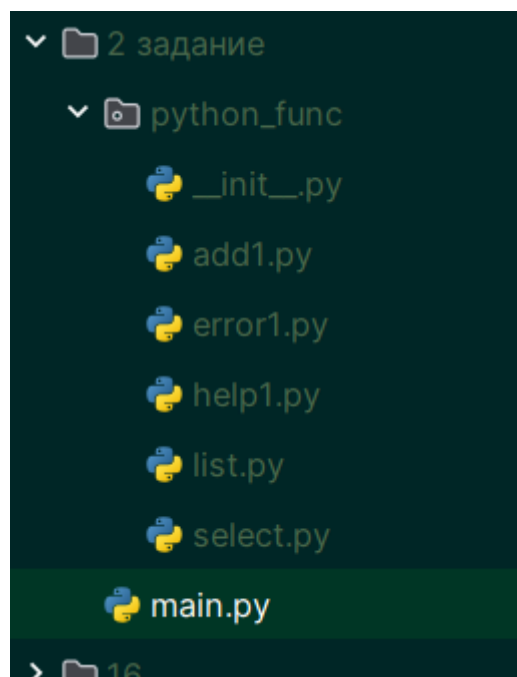
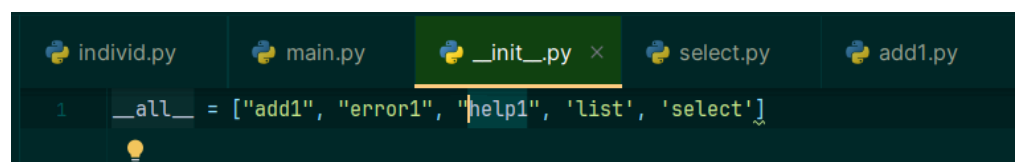


Рисунок 8 – Структура



```
1  __all__ = ["add1", "error1", "help1", "list", "select"]
```

Рисунок 9 – Файл \_\_init\_\_.py

```

add - добавить рейс;
list - вывести список рейсов;
select <тип> - вывод на экран пунктов назначения и номеров рейсов для данного типа самолёта
help - отобразить справку;
exit - завершить работу с программой.
>>> add
Название пункта назначения рейса? Афины
Номер рейса? 12
Тип самолета? c
>>> list
+-----+-----+-----+
| 1 | Афины | 12 | c |
+-----+-----+-----+
>>> select
Введите тип:c
Пункт назначения - Афины
Номер рейса - 12
>>> |

```

Рисунок 10 – результат работы программы

7.Зафиксировал все изменения в github в ветке develop.

```

aregd@DESKTOP-5KV9QA9 MINGW64 /d/Рабочий стол/4 семестр/опи/16/16 (develop)
$ git add .

aregd@DESKTOP-5KV9QA9 MINGW64 /d/Рабочий стол/4 семестр/опи/16/16 (develop)
$ git commit -m"12"
[develop 0d63804] 12
 9 files changed, 157 insertions(+)
 create mode 100644 "PyCharm/1 \320\267\320\260\320\264\320\260\320\275\320\270\320\265\fun.py"
 create mode 100644 "PyCharm/1 \320\267\320\260\320\264\320\260\320\275\320\270\320\265/main.py"
 create mode 100644 "PyCharm/2 \320\267\320\260\320\264\320\260\320\275\320\270\320\265/main.py"
 create mode 100644 "PyCharm/2 \320\267\320\260\320\264\320\260\320\275\320\270\320\265/python_func/__init__.py"
 create mode 100644 "PyCharm/2 \320\267\320\260\320\264\320\260\320\275\320\270\320\265/python_func/add1.py"
 create mode 100644 "PyCharm/2 \320\267\320\260\320\264\320\260\320\275\320\270\320\265/python_func/error1.py"
 create mode 100644 "PyCharm/2 \320\267\320\260\320\264\320\260\320\275\320\270\320\265/python_func/help1.py"
 create mode 100644 "PyCharm/2 \320\267\320\260\320\264\320\260\320\275\320\270\320\265/python_func/list.py"
 create mode 100644 "PyCharm/2 \320\267\320\260\320\264\320\260\320\275\320\270\320\265/python_func/select.py"

aregd@DESKTOP-5KV9QA9 MINGW64 /d/Рабочий стол/4 семестр/опи/16/16 (develop)
$ git push origin develop
Enumerating objects: 16, done.
Counting objects: 100% (16/16), done.
Delta compression using up to 8 threads
Compressing objects: 100% (15/15), done.
Writing objects: 100% (15/15), 3.56 KiB | 1.19 MiB/s, done.
Total 15 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'develop' on GitHub by visiting:
remote:   https://github.com/aregdz/16/pull/new/develop
remote:
To https://github.com/aregdz/16.git
 * [new branch]      develop -> develop

```

Рисунок 8 – фиксация изменений в ветку develop

8.Слил ветки.

```

aregd@DESKTOP-5KV9QA9 MINGW64 /d/Рабочий стол/4 семестр/опи/16/16 (develop)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

aregd@DESKTOP-5KV9QA9 MINGW64 /d/Рабочий стол/4 семестр/опи/16/16 (main)
$ git merge develop
Updating 073108c..0d63804
Fast-forward
 .../fun.py" | 10 ++++++
 .../main.py" | 15 ++++++++
 .../main.py" | 40 ++++++
 .../python_func/__init__.py" | 1 +
 .../python_func/add1.py" | 19 ++++++++
 .../python_func/error1.py" | 8 +++++
 .../python_func/help1.py" | 14 ++++++
 .../python_func/list.py" | 27 ++++++
 .../python_func/select.py" | 23 ++++++
 9 files changed, 157 insertions(+)
 create mode 100644 "PyCharm/1 \320\267\320\260\320\264\320\260\320\275\320\270\320\265\fun.py"
 create mode 100644 "PyCharm/1 \320\267\320\260\320\264\320\260\320\275\320\270\320\265\main.py"
 create mode 100644 "PyCharm/2 \320\267\320\260\320\264\320\260\320\275\320\270\320\265\main.py"
 create mode 100644 "PyCharm/2 \320\267\320\260\320\264\320\260\320\275\320\270\320\265\python_func\__init__.py"
 create mode 100644 "PyCharm/2 \320\267\320\260\320\264\320\260\320\275\320\270\320\265\python_func\add1.py"
 create mode 100644 "PyCharm/2 \320\267\320\260\320\264\320\260\320\275\320\270\320\265\python_func\error1.py"
 create mode 100644 "PyCharm/2 \320\267\320\260\320\264\320\260\320\275\320\270\320\265\python_func\help1.py"
 create mode 100644 "PyCharm/2 \320\267\320\260\320\264\320\260\320\275\320\270\320\265\python_func\list.py"
 create mode 100644 "PyCharm/2 \320\267\320\260\320\264\320\260\320\275\320\270\320\265\python_func\select.py"

aregd@DESKTOP-5KV9QA9 MINGW64 /d/Рабочий стол/4 семестр/опи/16/16 (main)
$

```

Рисунок 9 – сливание ветки develop в ветку main

## Контрольные вопросы:

### 1 Что является модулем языка Python?

В Python модуль - это файл с расширением .py, содержащий код на языке Python. Модуль может содержать определения функций, классов, переменных и другие инструкции, которые могут быть использованы в других модулях или скриптах.

### 2 Какие существуют способы подключения модулей в языке Python?

Импорт по имени модуля - `import module`

Импорт с переименованием - `import module as m`

Импорт конкретных элементов модуля – `from module_name import item1, item2`

Импорт всех элементов модуля - `from module_name import *`

### 3 Что является пакетом языка Python?

В Python пакет - это директория, которая содержит один или несколько модулей, а также файл `__init__.py`, который указывает Python, что эта директория является пакетом. Пакеты используются для организации и структурирования больших проектов на Python. Они позволяют логически группировать связанные модули вместе, что облегчает управление и поддержку кода.

#### **4 Каково назначение файла `__init__.py` ?**

Файл `__init__.py` может быть пустым или может содержать переменную `__all__`, хранящую список модулей, который импортируется при загрузке через конструкцию.

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

Переменная `__all__` в файле `__init__.py` в Python используется для определения списка символов, которые будут импортированы, когда мы используем конструкцию `from package_name import *`. Когда мы импортируем все символы из пакета с помощью `from package_name import *`, Python импортирует только те символы, которые перечислены в списке `__all__`.