

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ  
ФЕДЕРАЦИИ**

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

**Кафедра инфокоммуникаций**

**«Модули и пакеты»**

**Отчет по лабораторной работе № 2.13**

**по дисциплине «Основы программной инженерии»**

Выполнил студент группы ПИЖ-б-о-21-1

Халимендик Я. Д. « » 2022г.

Подпись студента \_\_\_\_\_

Работа защищена « » \_\_\_\_\_ 20\_\_ г.

Проверил 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. Изучить теоретический материал работы.
2. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.

**Create a new repository**

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

---

**Owner \*** Yana-Kh / **Repository name \*** Lab 16 OPJ ✓

Great repository names: Your new repository will be created as Lab-16-OPJ. How about [symmetrical-octo-succotash?](#)

**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

This will set `main` as the default branch. Change the default name in your [settings](#).

**Info** You are creating a public repository in your personal account.

**Create repository**

Рисунок 1 – Создание репозитория

3. Выполните клонирование созданного репозитория.

```
C:\Users\ynakh\OneDrive\Рабочий стол\Git>git clone https://github.com/Yana-Kh/Lab-16-OPJ.git
Cloning into 'Lab-16-OPJ'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.

C:\Users\ynakh\OneDrive\Рабочий стол\Git>_
```

Рисунок 2 – Клонирование репозитория

4. Дополните файл .gitignore необходимыми правилами для работы с IDE PyCharm.

```
C:\Users\ynakh\OneDrive\Рабочий стол\Git\Lab-16-OPJ>git add .
C:\Users\ynakh\OneDrive\Рабочий стол\Git\Lab-16-OPJ>git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   .gitignore

C:\Users\ynakh\OneDrive\Рабочий стол\Git\Lab-16-OPJ>_
```

Рисунок 3 – Дополнение файла .gitignore

5. Организуйте свой репозиторий в соответствии с моделью ветвления git-flow.

```
C:\Users\ynakh\OneDrive\Рабочий стол\Git\Lab-16-OPJ>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/ynakh/OneDrive/Рабочий стол/Git/Lab-16-OPJ/.git/hooks]

C:\Users\ynakh\OneDrive\Рабочий стол\Git\Lab-16-OPJ>_
```

Рисунок 4 – организация репозитория в соответствии с моделью git-flow

## 6. Создайте проект PyCharm в папке репозитория.

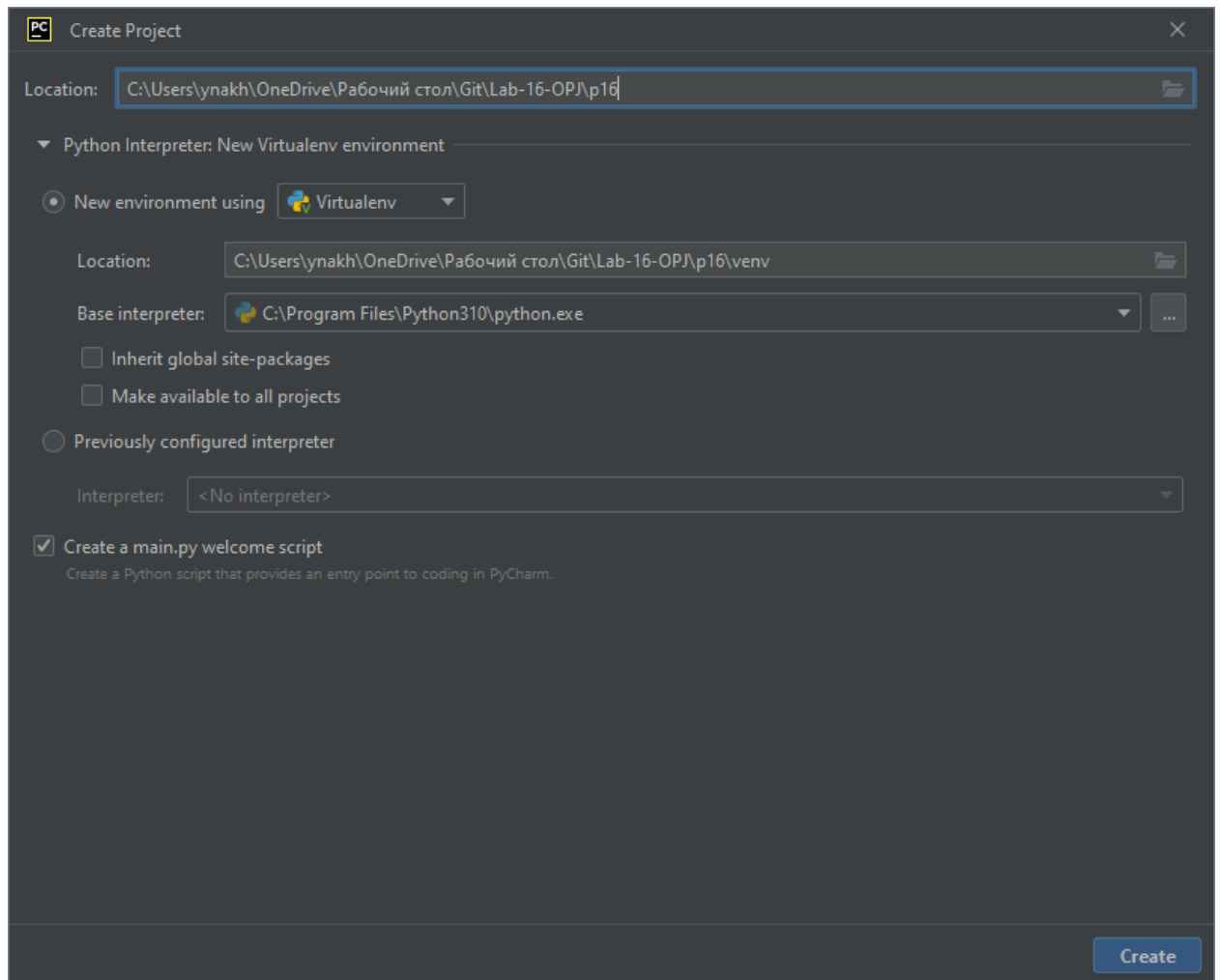


Рисунок 5 – Создание проекта PyCharm в папке репозитория

## 7. Выполните индивидуальные задания.

Вариант 32:

### 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.

- Используя замыкания функций, объявите внутреннюю функцию, которая заключает строку `s` (`s` – строка, параметр внутренней функции) в произвольный тег, содержащийся в переменной `tag` – параметре внешней функции. Далее, на вход программы поступает две строки: первая с тегом, вторая с некоторым содержимым. Вторую строку нужно поместить в тег из первой строки с помощью реализованного замыкания. Результат выведите на экран.

Модуль:

```
def get_tag(tag):  
    x = tag  
  
    def get_str(s):  
        nonlocal x  
        return ": ".join([x, s])  
    return get_str
```

Основная программа:

```
#!/usr/bin/env python3  
# -*- coding: utf-8 -*-  
  
from ex1_mod1 import*  
  
if __name__ == '__main__':  
    test_fun = get_tag("test tag")  
    print(test_fun("hello"))
```

```
test tag: hello  
  
Process finished with exit code 0
```

Рисунок 8 – Результат работы 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.

13. Использовать словарь, содержащий следующие ключи: фамилия, имя; номер телефона; дата рождения (список из трех чисел). Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть упорядочены по трем первым цифрам номера телефона; вывод на экран информации о человеке, чья фамилия введена с клавиатуры; если такого нет, выдать на дисплей соответствующее сообщение.

Содержание пакета:

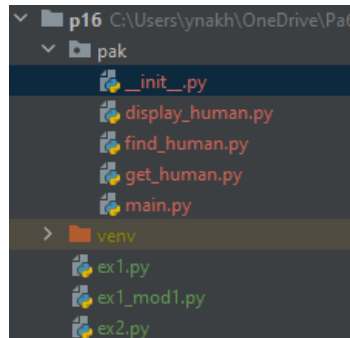


Рисунок 9 – Содержание проекта

Код:

`__init__.py`

```
__all__ = ["main", "display_human", "find_human", "get_human"]
```

`display_human.py`

```
def display_human(staff):
    """
    Отобразить список работников.
    """
    # Проверить, что список работников не пуст.
    if staff:
        # Заголовок таблицы.
        line = '+-{}-+-{}-+-{}-+-{}-+'.format(
            '-' * 4,
            '-' * 30,
            '-' * 15,
            '-' * 15
        )
        print(line)
        print(
            '| {:^4} | {:^30} | {:^15} | {:^15} |'.format(
                "№",
                "Ф и И.",
                "Телефон",
                "День рождения"
            )
        )
        print(line)

        # Вывести данные о всех сотрудниках.
        for idx, human in enumerate(staff, 1):
            print(
                f'| {idx:>4} |'
                f' {human.get("name", ""):<30} |'
                f' {human.get("phone", 0):<15} |'
                f' {human.get("birthday")} |'
            )
            print(line)
```

```
else:
    print("Список пуст.")
```

### find\_human.py

```
def find_human(staff, fname):
    """
    Выбрать работников с заданным стажем.
    """
    # Сформировать список людей.
    result = []
    for h in staff:
        if fname in str(h.values()):
            result.append(h)

    # Проверка на наличие записей
    if len(result) == 0:
        return print("Запись не найдена")

    # Возвратить список выбранных работников.
    return result
```

### get\_human.py

```
import datetime

def get_human():
    """
    Запросить данные о работнике.
    """
    name = input("Фамилия и имя: ")
    phone = int(input("Номер телефона: +7"))
    bday = list(map(int, input("Дата рождения: ").split('.')))
    d_bday = datetime.date(bday[2], bday[1], bday[0])

    # Вернуть словарь.
    return {
        'name': name,
        'phone': phone,
        'birthday': d_bday
    }
```

### main.py

```
import sys

from pak.find_human import find_human
from pak.get_human import get_human
from pak.display_human import display_human

def main():
    """
    Главная функция программы.
    """
    # Список работников.
    people = []
```

```

# Организовать бесконечный цикл запроса команд.
while True:
    # Запросить команду из терминала.
    command = input(">>> ").lower()

    # Выполнить действие в соответствие с командой.
    if command == 'exit':
        break

    elif command == 'add':
        # Запросить данные о работнике.
        human = get_human()

        # Добавить словарь в список.
        people.append(human)
        # Отсортировать список в случае необходимости.
        if len(people) > 1:
            people.sort(key=lambda item: item.get('phone', 0))

    elif command == 'list':
        # Отобразить всех работников.
        display_human(people)

    elif command == 'find':
        f = input('Введите фамилию: ')

        # Выбрать людей с заданной фамилией.
        selected = find_human(people, f)
        # Отобразить выбранных работников.
        display_human(selected)

    elif command == 'help':
        # Вывести справку о работе с программой.
        print("Список команд:\n")
        print("add - добавить человека;")
        print("list - вывести список людей;")
        print("find - найти человека по фамилии;")
        print("help - отобразить справку;")
        print("exit - завершить работу с программой.")

    else:
        print(f"Неизвестная команда {command}", file=sys.stderr)

```

### Код основной программы:

```

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

from pak.main import *

if __name__ == '__main__':
    main()

```



```

>>> help
Список команд:

add - добавить человека;
list - вывести список людей;
find - найти человека по фамилии;
help - отобразить справку;
exit - завершить работу с программой.
>>> add
Фамилия и имя: Ivanov Ivan
Номер телефона: +79097727643
Дата рождения: 09.12.2020
>>> add
Фамилия и имя: Petrov
Номер телефона: +79823021000
Дата рождения: 01.08.2000
>>> list
+-----+-----+-----+-----+
| № |           Ф и И.           |   Телефон   |   День рождения   |
+-----+-----+-----+-----+
|   1 | Ivanov Ivan                | 9097727643   | 2020-12-09        |
+-----+-----+-----+-----+
|   2 | Petrov                    | 9823021000   | 2000-08-01        |
+-----+-----+-----+-----+
>>> find
Введите фамилию: Ivanov
+-----+-----+-----+-----+
| № |           Ф и И.           |   Телефон   |   День рождения   |
+-----+-----+-----+-----+
|   1 | Ivanov Ivan                | 9097727643   | 2020-12-09        |
+-----+-----+-----+-----+
>>> find
Введите фамилию: Sidorov
Запись не найдена
Список пуст.
>>> exit

Process finished with exit code 0

```

Рисунок 8 – Результат работы программы

## Вопросы для защиты работы

### 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. Стоит заметить, что модули могут быть написаны не только на языке Python, но и на других языках (например C).

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

Самый простой способ импортировать модуль в Python это воспользоваться конструкцией:

```
import имя_модуля
```

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 as новое_имя
```

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

```
from имя_модуля import имя_объекта1, имя_объекта2
```

### 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`?

В `__init__.py` файл заставляет Python рассматривать каталоги, содержащие его, как модули. Кроме того, это первый файл, загружаемый в модуль, поэтому вы можете использовать его для выполнения кода, который хотите запускать каждый раз при загрузке модуля, или для указания экспортируемых подмодулей.

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

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

```
from имя_пакета import *
```

Вывод: в ходе выполнения практической работы были приобретены навыки по работе декораторами функций при написании программ с помощью языка программирования Python.