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

Кучеренко С. Ю. « » 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.

Выполнение работы:

1. Изучить теоретический материал работы.

2. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.

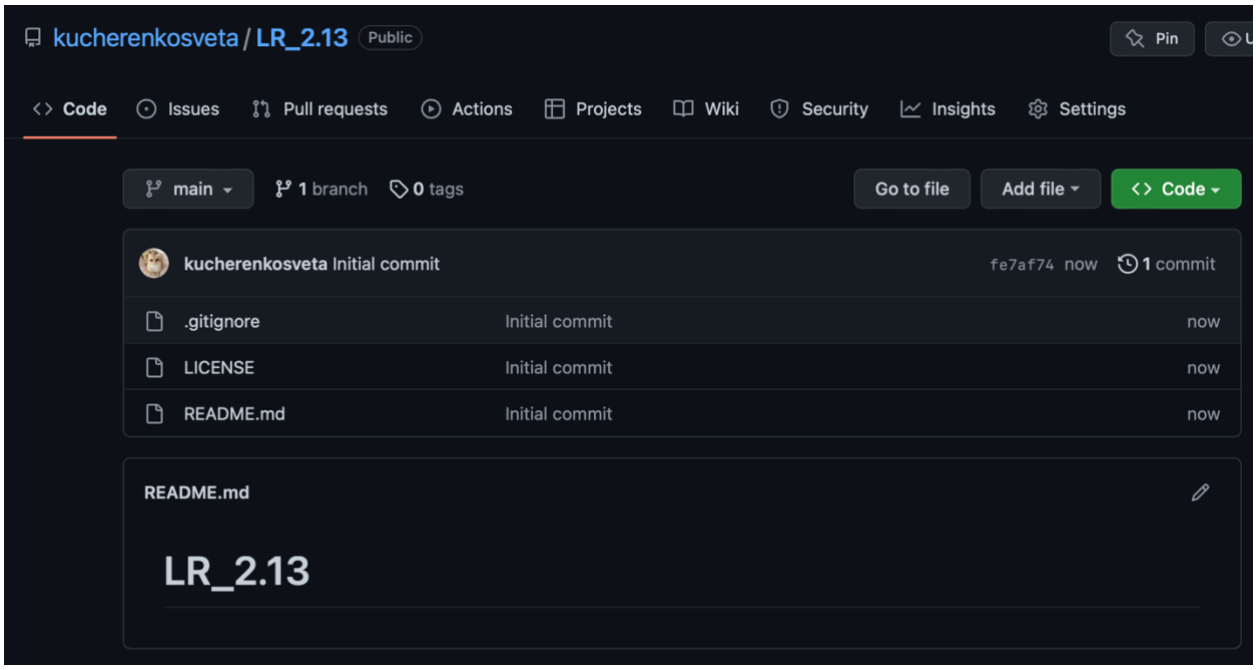

Рисунок 1 – Создание репозитория

3. Выполните клонирование созданного репозитория.

```
[(base) svetik@MacBook-Air-Svetik ~ % cd ~/Desktop/Laba16
[(base) svetik@MacBook-Air-Svetik Laba16 % git clone https://github.com/kucherenk
osveta/LR_2.13.git
Cloning into 'LR_2.13'...
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.
(base) svetik@MacBook-Air-Svetik Laba16 %
```

Рисунок 2 – Клонирование репозитория

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

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

```
(base) svetik@MacBook-Air-Svetik LR_2.13 % 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? [/Users/svetik/Desktop/Laba16/LR_2.13/.git/hooks]
(base) svetik@MacBook-Air-Svetik LR_2.13 %
```

Рисунок 3 – организация репозитория в соответствии с моделью git-flow

6. Создайте проект PyCharm в папке репозитория.

7. Выполните индивидуальные задания.

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 принимает в качестве аргумента коллекцию (список или кортеж) и возвращает или минимальное значение, или максимальное, в зависимости от значения параметра type внешней функции. Если type равен «max», то возвращается максимальное значение, иначе – минимальное. По умолчанию type должно принимать значение «max». Вызовите внутреннюю функцию замыкания и отобразите на экране результат ее работы.

Модуль:

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

def max_min(type='max'):
    def helper(collection):
        if type == 'max':
```

```
            return max(collection)
        else:
            return min(collection)
    return helper
```

Программа:

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

from ind1_m import max_min

if __name__ == '__main__':
    print(max_min(input("Введите max или min: "))
          (map(int, input("Введите список: ").split(' '))))
```

```
Введите max или min: max
Введите список: 1 2 3 7 -4 5
7
```

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.

Использовать словарь, содержащий следующие ключи: название начального пункта маршрута; название конечного пункта маршрута; номер маршрута. Написать программу, выполняющую следующие действия:

- ввод с клавиатуры данных в список, состоящий из словарей заданной структуры;
- записи должны быть упорядочены по номерам маршрутов;
- вывод на экран информации о маршруте, номер которого введен с клавиатуры; если таких маршрутов нет, выдать на дисплей соответствующее сообщение.

Содержание пакета:

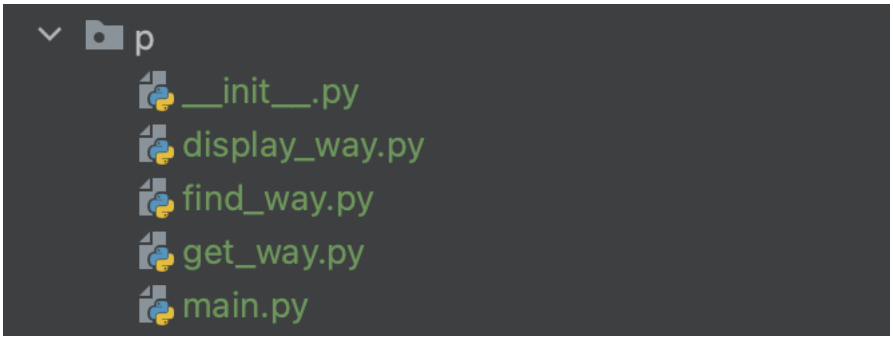

Рисунок 9 – Содержание проекта

__init__.py

```
__all__ = ["main", "display_way", "find_way", "get_way"]
```

display_way.py

```
def display_way(numbers):
    """
    Отобразить список маршрутов.
    """
    if numbers:
        # Заголовок таблицы.
        line = '+-{}-+-{}-+-{}-+-{}-+'.format(
            '-' * 4,
            '-' * 30,
            '-' * 30,
            '-' * 15
        )
        print(line)
        print(
            '| {:^4} | {:^30} | {:^30} | {:^15} |'.format(
                "No",
                "Название начального маршрута",
                "Название конечного маршрута",
                "Номер маршрута"
            )
        )
        print(line)

        # Вывести данные о всех маршрутах.
        for idx, way in enumerate(numbers, 1):
            print(
                '| {:>4} | {:<30} | {:<30} | {:>15} |'.format(
                    idx,
                    way.get('start', ''),
                    way.get('finish', ''),
                    way.get('num', 0)
                )
            )
        print(line)

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

find_way.py

```
def find_way(numbers, nw):
    """
    Выбрать маршрут с данным номером.
    """
    # Список маршрутов
    result = []
    for h in numbers:
        if nw in str(h.values()):
            result.append(h)

    # Проверка на наличие записей
    if len(result) == 0:
        return None

    # Возвратить список выбранных маршрутов.
    return result
```

get_way.py

```
def get_way():
    """
    Запросить данные о маршруте.
    """
    start = input('Название начального маршрута: ')
    finish = input('Название конечного маршрута: ')
    num = int(input('Номер маршрута: '))

    # Вернуть словарь.
    return {
        'start': start,
        'finish': finish,
        'num': num
    }
```

main.py

```
import sys

from p.find_way import find_way
from p.get_way import get_way
from p.display_way import display_way


def main():
    """
    Главная функция программы.
    """
    # Маршруты
    ways = []

    # 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 маршруте.
            way = get_way()

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

        elif command == 'list':
            # Отобразить все маршруты.
            display_way(ways)

        elif command == 'find':
            f = input('Введите номер маршрута: ')
            selected = find_way(ways, f)
            display_way(selected)

        elif command == 'help':
            # Вывести справку.
            print("Список команд:\n")
            print("add - добавить маршрут;")
            print("list - вывести список маршрутов;")
            print("find - вывод информации о маршруте;")
            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 p.main import main

if __name__ == '__main__':
    main()
```

```
>>> add
Название начального маршрута: fhbed
Название конечного маршрута: fhth
Номер маршрута: 22
>>> add
Название начального маршрута: dfhn
Название конечного маршрута: edth
Номер маршрута: 55
>>> list
+------+--------------------------------+--------------------------------+-----------------+
|  No  |  Название начального маршрута  |  Название конечного маршрута   | Номер маршрута  |
+------+--------------------------------+--------------------------------+-----------------+
|    1 | fhbed                          | fhth                           |              22 |
|    2 | dfhn                           | edth                           |              55 |
+------+--------------------------------+--------------------------------+-----------------+
>>> help
Список команд:

add - добавить маршрут;
list - вывести список маршрутов;
find - вывод информации о маршруте;
help - отобразить справку;
exit - завершить работу с программой.
```

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 имортирования 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.