

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования «СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ
УНИВЕРСИТЕТ»

Институт цифрового развития Кафедра
инфокоммуникаций

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №20
дисциплины «Основы программной инженерии»

Выполнил:

Рядская Мария Александровна

2 курс, группа ПИЖ-б-о-22-1,

09.03.04 «Программная инженерия»,

направленность (профиль) «Разработка и

сопровождение программного

обеспечения», очная форма обучения

Проверил Воронкин Роман Александрович

Ставрополь, 2024 г.

Тема: Лабораторная работа 2.17. Разработка приложений с интерфейсом

командной строки (CLI) в Python3

Цель работы: приобретение построения приложений с интерфейсом

командной строки с помощью языка программирования Python версии

3.x.

Ход работы.

1. Создание нового репозитория с лицензией MIT.

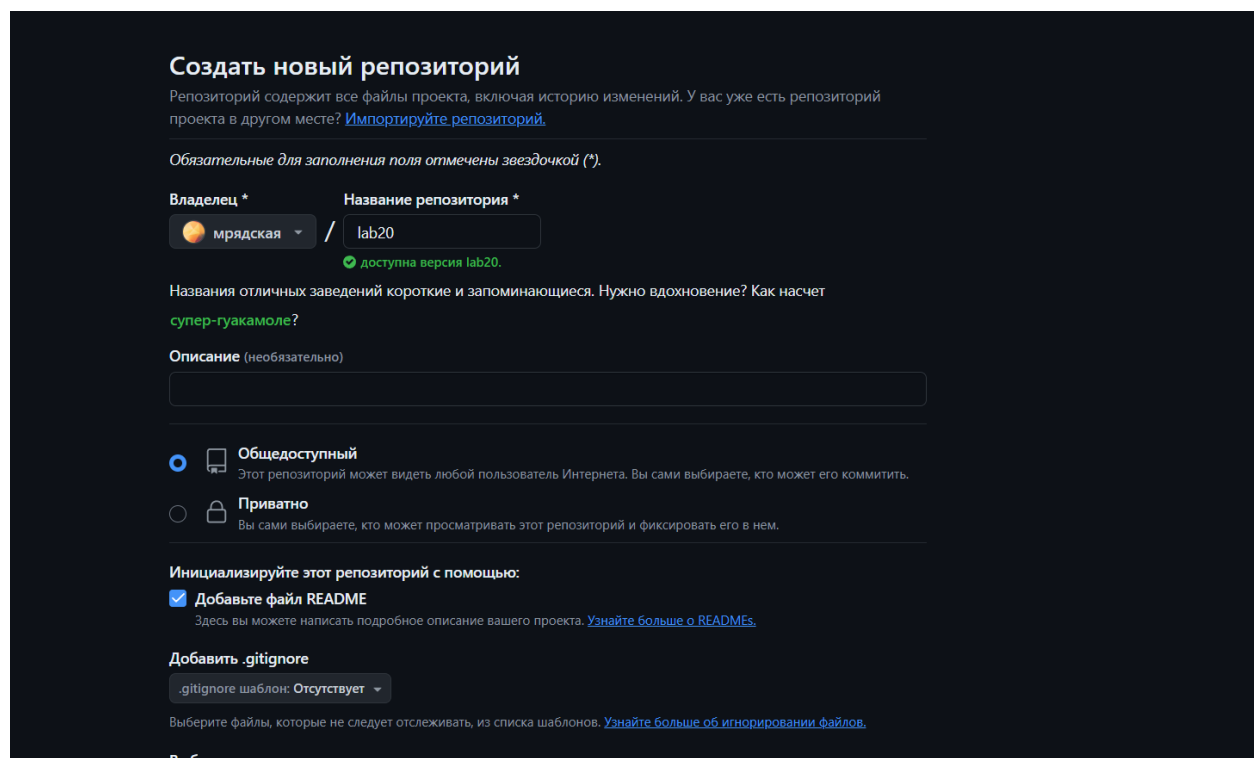


Рисунок 1 – создание репозитория

2. Клонировал репозиторий на рабочий ПК.

```

C:\Users>cd ..

C:\>cd git1

C:\git1>git clone https://github.com/mryadskaya/lab20.git
Cloning into 'lab20'...
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:\git1>

```

Рисунок 2 – клонирование репозитория

```

C:\git1>cd lab20

C:\git1\lab20>git checkout -b develop
Switched to a new branch 'develop'

C:\git1\lab20>

```

Рисунок 3 – организация ветки

4. Создала виртуальное окружение и установила нужные пакеты.

```

(venv) (base) PS C:\git1\lab19> pip install flake8, black, pre-commit Collecting flake8
Collecting flake8
  Obtaining dependency information for flake8 from https://files.pythonhosted.org/packages/e3/01/cc8cdec7b61db0315c2ab62d80677a138ef0
6832ec17f04d87e6ef858f7f/flake8-7.0.0-py2.py3-none-any.whl.metadata
  Downloading flake8-7.0.0-py2.py3-none-any.whl.metadata (3.8 kB)
Collecting black
  Obtaining dependency information for black from https://files.pythonhosted.org/packages/0f/11/fa05ac9429d971d0fc10da85f24dafc3fa578
8733fbd0d1c186b7577cefd/black-24.4.0-cp311-cp311-win_amd64.whl.metadata
  Downloading black-24.4.0-cp311-cp311-win_amd64.whl.metadata (76 kB)
----- 76.4/76.4 kB 1.4 MB/s eta 0:00:00
Collecting pre-commit
  Obtaining dependency information for pre-commit from https://files.pythonhosted.org/packages/53/1a/1870d52d0d86d534d4d5f7a58e65d40a
4610b57a12700917faa0e63818fa/pre-commit-3.7.0-py2.py3-none-any.whl.metadata

```

Рисунок 5 - создание виртуального окружения

5.Проработала примеры из лабораторной работы.

```
[
  {
    "name": "ryadskysya",
    "знак зодиака": "rak",
    "year": 2005
  }
]
```

6. Для своего варианта лабораторной работы 2.16 необходимо дополнительно реализовать интерфейс командной строки (CLI).

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

import argparse
import json
import os.path

def add_flight(flights, destination, departure_date, aircraft_type):

    flights.append(
        {
            "name": name,
            "знак зодиака": post,
            "year": year,
        }
    )

    return flights

def display_flights(flights):

    if flights:
        line = "+-{}-+-{}-+-{}-+-{}-+".format(
            *args: "-" * 4, "-" * 30, "-" * 20, "-" * 8
        )
        print(line)
        print(
            "| {:^4} | {:^30} | {:^20} | {:^8} |".format(
                *args: "No", "Name", "знак зодиака ", "year"
            )
        )
    )
```

Рисунок 8 – код для реализации задания

9. Самостоятельно изучите работу с пакетом click для построения интерфейса командной строки (CLI). Для своего варианта лабораторной работы 2.16 необходимо реализовать интерфейс командной строки с использованием пакета click .

Рисунок 9 – код для выполнения задания повышенной сложности

```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  > import ...
5
6
7
8
9
10 @click.group()
11 @click.version_option(version="0.1.0")
12 def flights():
13     """
14     Manage flight data.
15     """
16     pass
17
18
19 @flights.command()
20 @click.option(
21     "-f", "--filename", default="rrm.json", help="The data file name"
22 )
23 @click.option(
24     "-d",
25     "--name ",
26     prompt="name ",
27     help="name  of the flight",
28 )
29 @click.option(
30     "-dd",
31     "--знак зодиака",
32     prompt="post Date",
33     help="post date of the flight",
34 )
```

Контрольные вопросы:

1.Отличие терминала и консоли:

Терминал (или терминальное окно) - это программа, которая предоставляет пользователю интерфейс для взаимодействия с операционной системой через текстовый интерфейс. Терминал обычно предоставляет доступ к командной строке, где пользователь может вводить команды для выполнения различных задач.

Консоль - это текстовый интерфейс, предоставляемый операционной системой для взаимодействия с пользователем или другими программами. Он может быть встроенным в окружение рабочего стола или виртуальной машины. Консоль обычно работает поверх терминала и обеспечивает вывод текстовой информации и ввод команд пользователя.

2.Консольное приложение:

Консольное приложение - это приложение, которое работает в текстовом режиме и взаимодействует с пользователем через командную

строку или терминал. Оно обычно используется для выполнения задач
в

операционной системе или автоматизации определенных процессов.

Примеры

консольных приложений включают текстовые редакторы, утилиты
командной

строки и скрипты.

3.Средства Python для построения приложений командной строки:

Модуль sys: для доступа к аргументам командной строки и другим
системным параметрам.

Модуль getopt: для парсинга аргументов командной строки в стиле
POSIX.

Модуль argparse: для создания гибких и мощных интерфейсов
командной строки с поддержкой подпрограмм, позиционных и
именованных

аргументов, справочной информации и др.

4.Особенности построения CLI с использованием модуля sys:

Модуль sys предоставляет базовые инструменты для работы с
аргументами командной строки, доступом к системным переменным и
другими системными функциями.

С его помощью можно получить доступ к аргументам командной
строки через список sys.argv и осуществить простой парсинг
аргументов.

5. Особенности построения CLI с использованием модуля getopt:

Модуль getopt позволяет парсить аргументы командной строки в стиле

POSIX с короткими и длинными опциями.

Он предоставляет более гибкие возможности по обработке аргументов,

чем модуль sys, но требует более сложного кода для использования.

6. Особенности построения CLI с использованием модуля argparse:

Модуль argparse предоставляет мощные средства для создания гибких интерфейсов командной строки.

Он поддерживает позиционные и именованные аргументы,

подпрограммы, группы аргументов, справочную информацию и многое

другое.

argparse обеспечивает автоматическую проверку типов аргументов,

генерацию справки и сообщений об ошибках.