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 №11
## дисциплины «Основы программной инженерии»

Выполнил:
Яблоновский Дмитрий Николаевич
2 курс, группа ПИЖ-б-о-22-1,
09.03.04 «Программная инженерия»,
направленность (профиль) «Разработка и сопровождение программного обеспечения», очная форма обучения

_____
(подпись)

Руководитель практики:

Богданов С.С., ассистент кафедры инфокоммуникаций

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2023 г.

**Тема:** Работа с множествами в языке Python.

**Цель работы:** приобретение навыков по работе со строками при написании программ с помощью языка программирования Python версии 3.x.

## Порядок выполнения работы

1. Создал репозиторий GitHub.

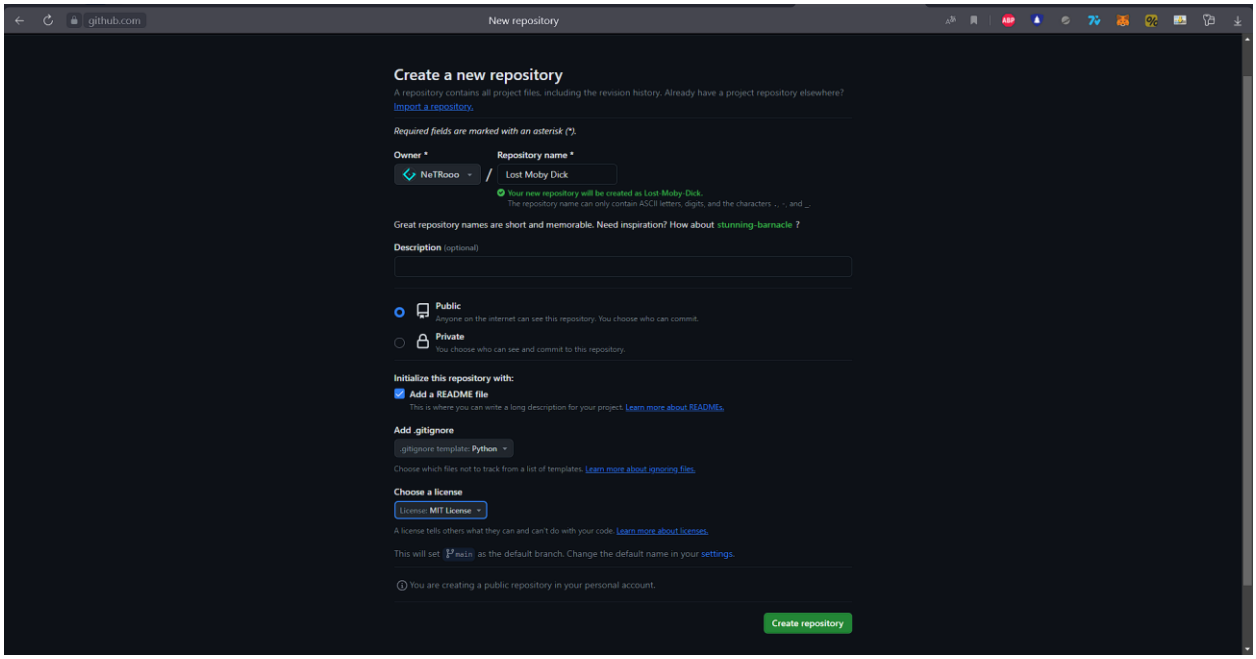


Рисунок 1- Создание репозитория

2. Проработал примеры из лабораторной работы.

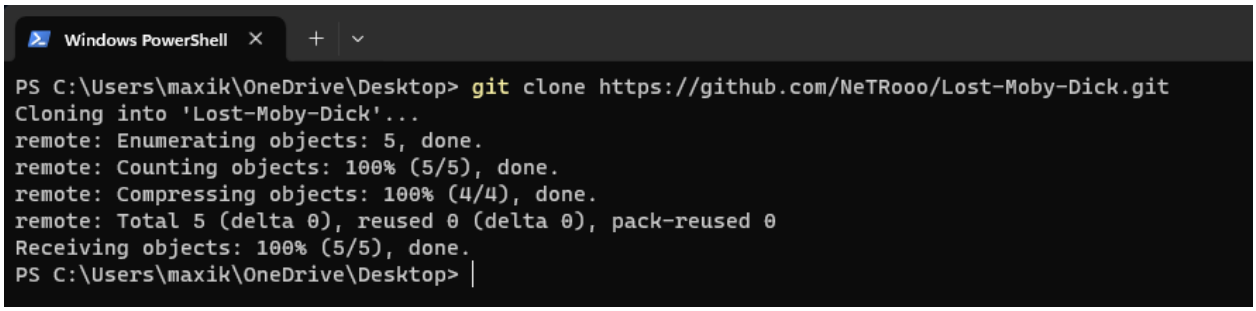


Рисунок 2 – Выполнил клонирование созданного репозитория

Код программы из примера:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
import sys
from datetime import date
def get_worker():
    """
    Запросить данные о работнике.
    """
    name = input("Фамилия и инициалы? ")
    post = input("Должность? ")
    year = int(input("Год поступления? "))
    # Создать словарь.
    return {
    'name': name,
        'post': post,
        'year': year,
    }
def display_workers(staff):
    """
    Отобразить список работников.
    """
    # Проверить, что список работников не пуст.
```

```
if staff:
    # Заголовок таблицы.
    line = '+-{}-+-{}-+-{}-+-{}-+'.format(
    '-' * 4,
        '-' * 30,
        '-' * 20,
        '-' * 8
    )
    print(line)
    print(
    '| {:^4} | {:^30} | {:^20} | {:^8} |'.format(
        "№",
        "Ф.И.О.",
        "Должность",
        "Год"
    )
)
    print(line)
    # Вывести данные о всех сотрудниках.
    for idx, worker in enumerate(staff, 1):
        print(
        '| {:>4} | {:<30} | {:<20} | {:>8} |'.format(
            idx,
```

```
            worker.get('name', ''),
            worker.get('post', ''),
        worker.get('year', 0)
        )
        )
        print(line)


    else:
        print("Список работников пуст.")
def select_workers(staff, period):
    """
    Выбрать работников с заданным стажем.
    """
    # Получить текущую дату.
    today = date.today()
    # Сформировать список работников.
    result = []
    for employee in staff:
        if today.year - employee.get('year', today.year) >= period:
            result.append(employee)

    # Возвратить список выбранных работников.
    return result
```

```
def main():
    """
    Главная функция программы.
    """
    # Список работников.
    workers = []
    # 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.
            worker = get_worker()
            # Добавить словарь в список.
            workers.append(worker)
            # Отсортировать список в случае необходимости.
            if len(workers) > 1:
                workers.sort(key=lambda item: item.get('name', ''))
        elif command == 'list':
            # Отобразить всех работников.
```

```
            display_workers(workers)
        elif command.startswith('select '):
            # Разбить команду на части для выделения стажа.
            parts = command.split(' ', maxsplit=1)
            # Получить требуемый стаж.
            period = int(parts[1])
            # Выбрать работников с заданным стажем.
            selected = select_workers(workers, period)
            # Отобразить выбранных работников.
            display_workers(selected)
        elif command == 'help':
            # Вывести справку о работе с программой.
            print("Список команд:\n")
            print("add - добавить работника;")
            print("list - вывести список работников;")
            print("select <стаж> - запросить работников со стажем;")
            print("help - отобразить справку;")
            print("exit - завершить работу с программой.")
        else:
            print(f"Неизвестная команда {command}", file=sys.stderr)
if __name__ == '__main__':
    main()
```

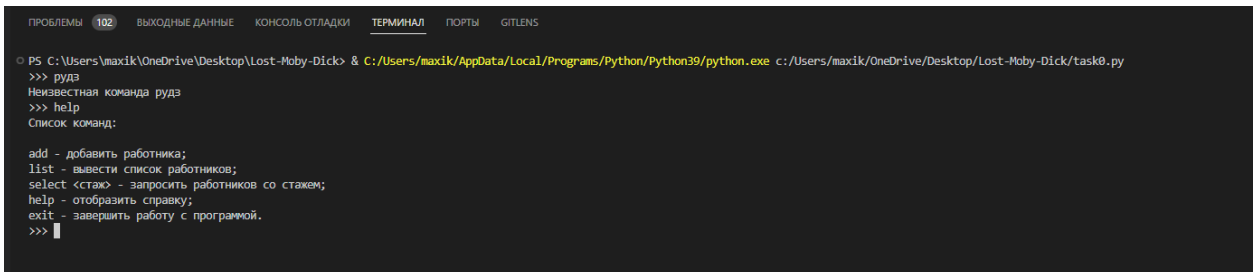


Рисунок 3 – Выполнение кода из примера.

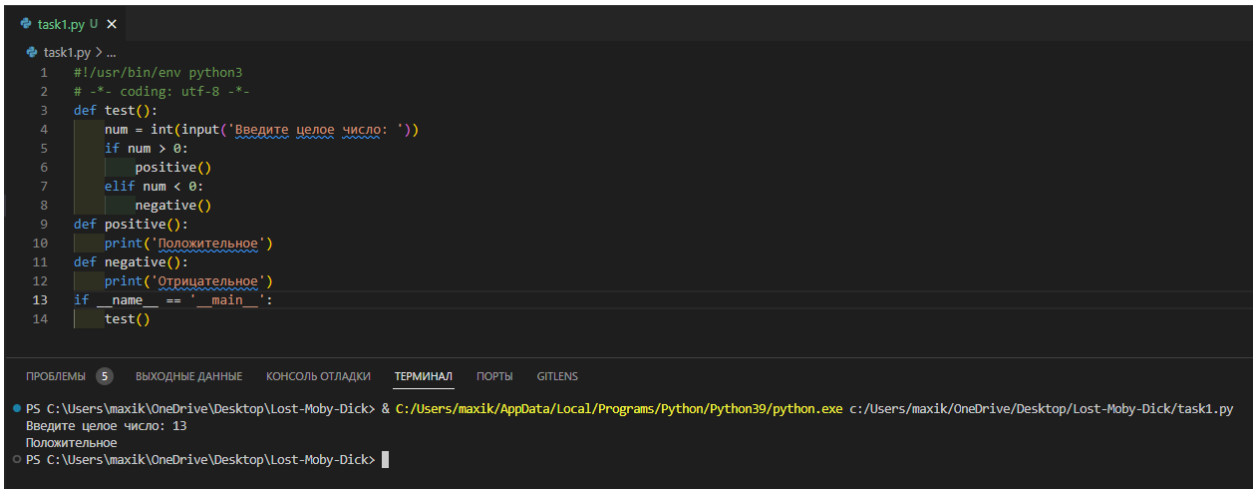


Рисунок 4 – Код задания и вывод при выполнении. (задание №1)

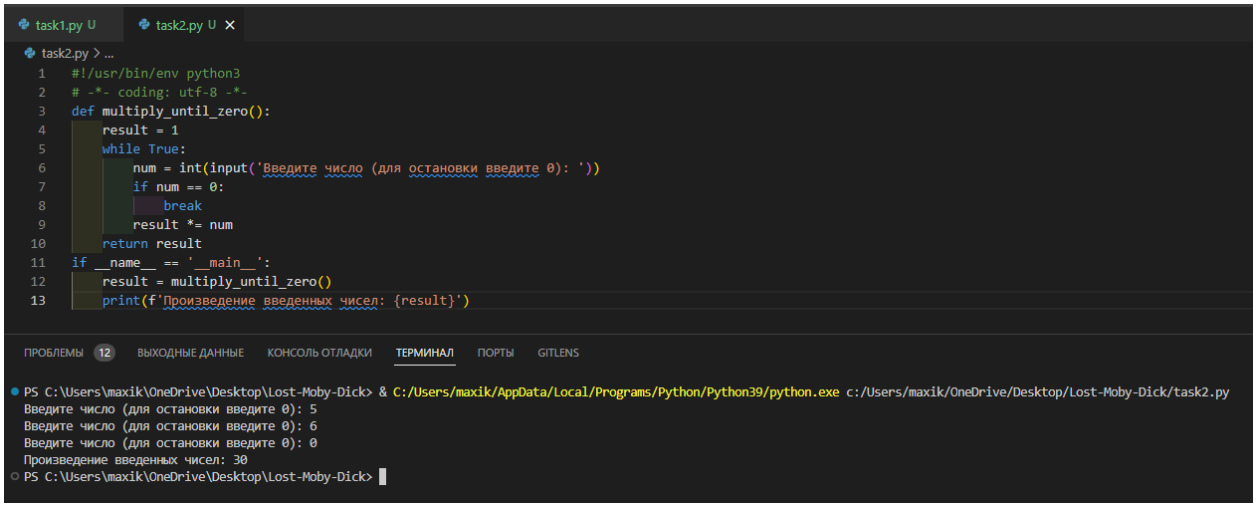


Рисунок 5 – Код задания и вывод при выполнении. (задание №2)

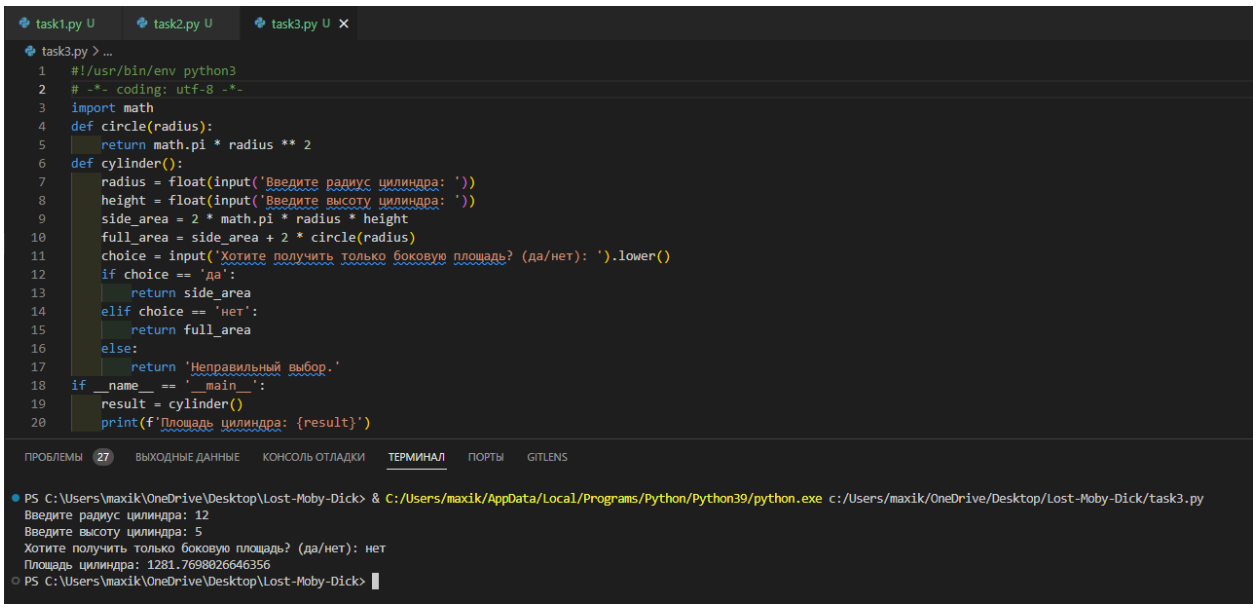


Рисунок 6 – Код задания и вывод при выполнении. (задание №3)

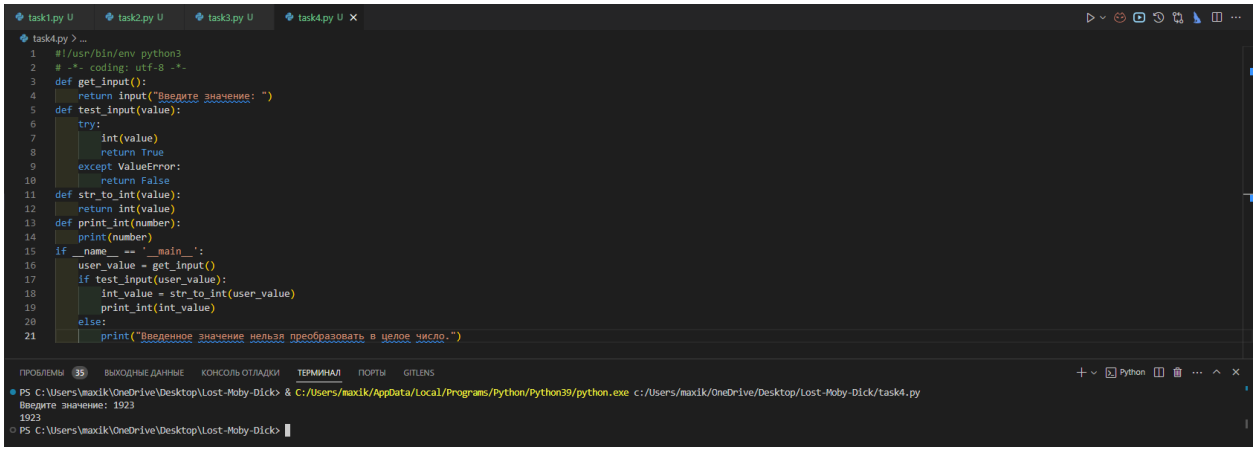


Рисунок 7 – Код задания и вывод при выполнении. (задание №4)

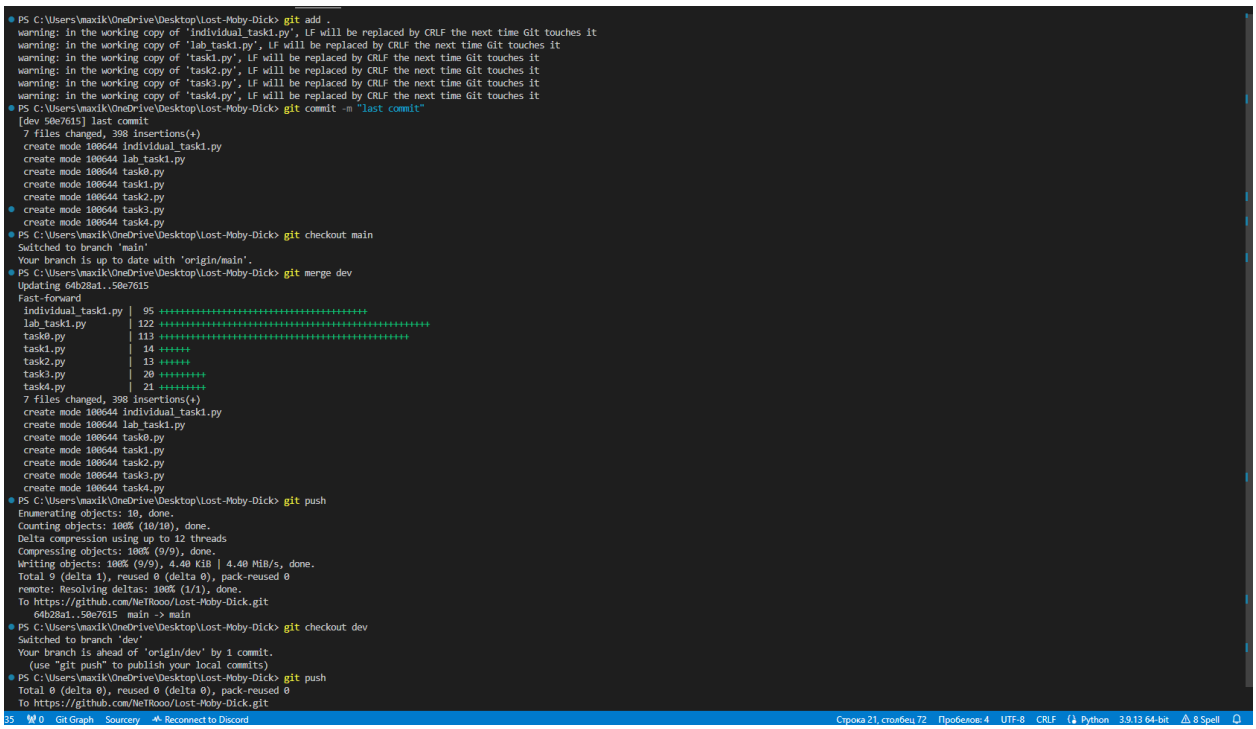


Рисунок 8 – Создание коммита и отправка коммита.

Вывод: в ходе выполнения лабораторной работы были получены навыки работы с множествами в языке Python.

Ссылка на репозиторий: https://github.com/NeTRooo/Lost-Moby-Dick

## Ответы на контрольные вопросы

1. Что такое списки в языке Python?

Списки в языке Python - это упорядоченные изменяемые коллекции элементов.

2. Каково назначение кортежей в языке Python?

Кортежи в языке Python используются для создания неизменяемых упорядоченных коллекций элементов.

3. Как осуществляется создание кортежей?

Кортежи создаются с использованием круглых скобок, например, my_tuple = (1, 2, 3).

4. Как осуществляется доступ к элементам кортежа?

Доступ к элементам кортежа осуществляется по индексу, например, element = my_tuple[0].

5. Зачем нужна распаковка (деструктуризация) кортежа?

Распаковка кортежа (деструктуризация) позволяет присваивать значения элементам кортежа одной строкой.

6. Какую роль играют кортежи в множественном присваивании?

Кортежи играют ключевую роль в множественном присваивании, где значения присваиваются сразу нескольким переменным.

7. Как выбрать элементы кортежа с помощью среза?

Выбор элементов кортежа с помощью среза осуществляется, например, subset = my_tuple[1:3].

8. Как выполняется конкатенация и повторение кортежей?

Конкатенация кортежей выполняется оператором +, а повторение - оператором *.

9. Как выполняется обход элементов кортежа?

Обход элементов кортежа осуществляется с использованием цикла, например, for item in my_tuple:.

10. Как проверить принадлежность элемента кортежу.

Принадлежность элемента кортежу можно проверить с использованием оператора in

11. Какие методы работы с кортежами Вам известны?

Некоторые методы работы с кортежами включают count() для подсчета элементов и index() для поиска индекса элемента.

12. Допустимо ли использование функций агрегации таких как len(), sum() и т. д. при работе с кортежами?

Да, функции агрегации, такие как len(), sum(), могут использоваться с кортежами.

13. Как создать кортеж с помощью спискового включения?

Кортеж можно создать с помощью спискового включения, например, my_tuple = tuple(x for x in my_list).