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 ПРАКТИЧЕСКОЙ РАБОТЕ №2.5
дисциплины
«Основы кроссплатформенного программирования»

Выполнила студентка группы
ИВТ-б-о-21-1
Яковлева Е.А. « »_____20__г.
Подпись студента_____
Работа защищена
« »_____20__г.

Проверил доцент
Кафедры инфокоммуникаций,
старший преподаватель
Воронкин Р.А.
_____
(подпись)

Ставрополь, 2022 г.

Тема: Работа с кортежами в языке Python

Цель: приобретение навыков по работе с кортежами при написании программ с помощью языка программирования Python версии 3.x.

Порядок выполнения работы:

1. Создала общедоступный репозиторий на GitHub, в котором использованы лицензия MIT и язык программирования Python.

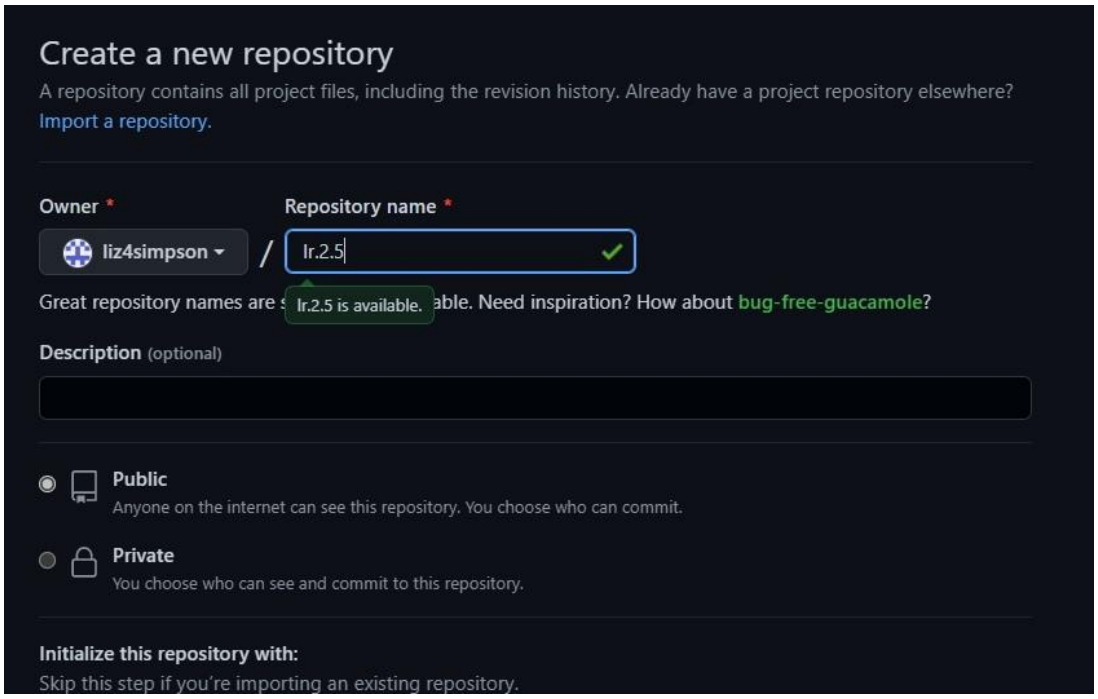

Рисунок 1. Создание репозитория

Выполнила клонирование созданного репозитория.

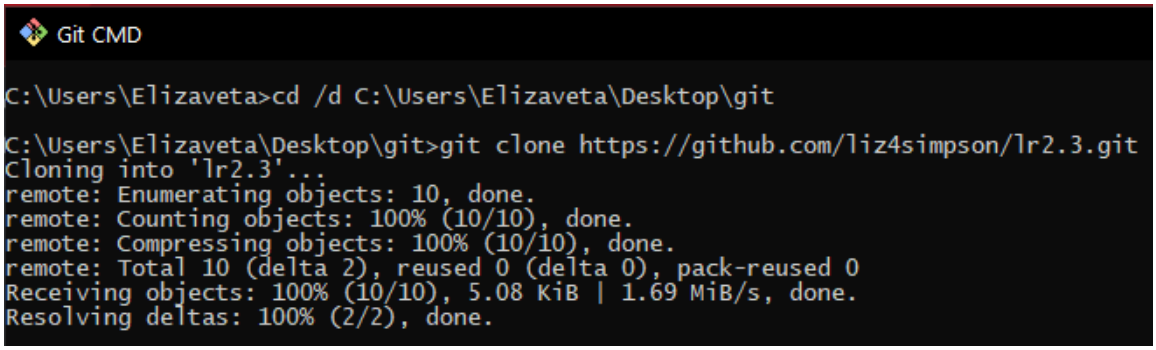

Рисунок 2. Клонирование репозитория

Организовала свой репозиторий в соответствие с моделью ветвления git-flow.

```
C:\Users\Elizaveta\Desktop\lr2.4>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/Elizaveta/Desktop/lr2.4/.git/hooks]
```

Рисунок 3. Организация репозитория согласно модели ветвления get-flow

2. Создала проект PyCharm в папке репозитория, проработала примеры из лабораторной работы.

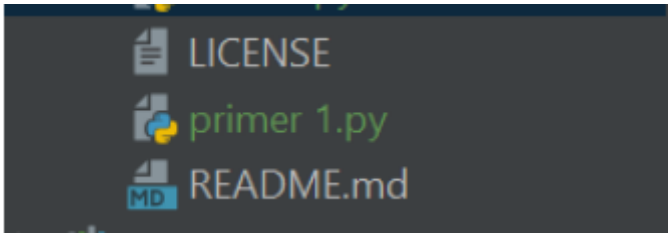

Рисунок 4. Проработанный пример

```
C:\Users\Elizaveta\Desktop\git\lr2.5\primers>C:/Users/Elizaveta/Desktop/git/lr2.
5/primers/primer1.py
1 4 3 5 3 6 3 5 12 4
18
```

Рисунок 5. Результат выполнения программы

3. Выполнила индивидуальное задание (в23)

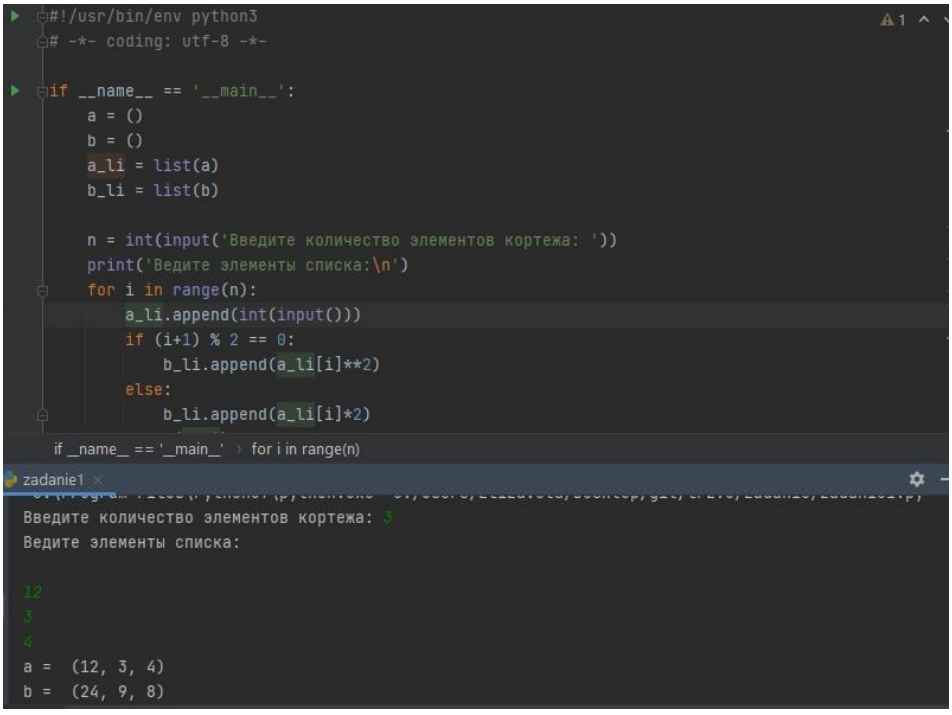

Рисунок 6. Индивидуальное задание

4. Сделала коммит, выполнила слияние с веткой main, и запушила изменения в удаленный репозиторий.

```
C:\Users\Elizaveta\Desktop\git\lr2.5>git add .

C:\Users\Elizaveta\Desktop\git\lr2.5>git commit -m "fine
[develop ef2e9df] fine
 8 files changed, 76 insertions(+)
 create mode 100644 .idea/.gitignore
 create mode 100644 .idea/inspectionProfiles/profiles_settings.xml
 create mode 100644 .idea/lr2.5.iml
 create mode 100644 .idea/misc.xml
 create mode 100644 .idea/modules.xml
 create mode 100644 .idea/vcs.xml
 create mode 100644 primers/primer1.py
 create mode 100644 zadanie/zadanie1.py
```

Рисунок 7. Фиксация и коммит файлов

```
C:\Users\Elizaveta\Desktop\git\lr2.5>git merge develop
Merge made by the 'ort' strategy.
 .idea/.gitignore                                 |  3 +++
 .idea/inspectionProfiles/profiles_settings.xml   |  6 ++++++
 .idea/lr2.5.iml                                  |  8 ++++++++
 .idea/misc.xml                                   |  4 ++++
 .idea/modules.xml                                |  8 ++++++++
 .idea/vcs.xml                                    |  6 ++++++
 primers/primer1.py                               | 21 +++++++++++++++++++++
 zadanie/zadanie1.py                              | 20 ++++++++++++++++++++
 8 files changed, 76 insertions(+)
```

Рисунок 8. Слияние ветки develop с main

```
C:\Users\Elizaveta\Desktop\git\lr2.5>git  push
Enumerating objects: 20, done.
Counting objects: 100% (20/20), done.
Delta compression using up to 4 threads
Compressing objects: 100% (14/14), done.
Writing objects: 100% (18/18), 2.60 KiB | 242.00 KiB/s, done.
Total 18 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), done.
To https://github.com/liz4simpson/lr2.5.git
   09b5c3f..f1dd6aa  main -> main
```

Рисунок 9. Отправка изменений на удаленный репозиторий

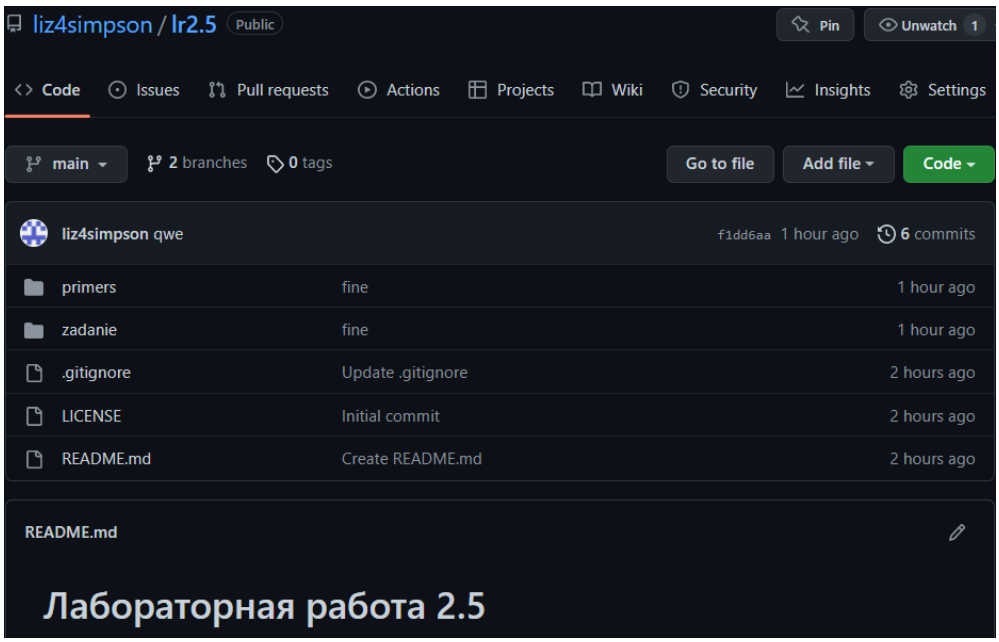

Рисунок 10. Изменения на удаленном репозитории

Вывод: в ходе лабораторной работы были приобретены навыки по работе с кортежами при написании программ.

Ответы на контрольные вопросы:

1. Что такое кортежи в языке Python?

Кортеж (tuple) – это неизменяемая структура данных, которая по-своему подобию очень похожа на список.

2. Каково назначение кортежей в языке Python?

Существует несколько причин, по которым стоит использовать кортежи вместо списков. Одна из них – это обезопасить данные от случайного изменения. Если мы получили откуда-то массив данных, и у нас есть желание поработать с ним, но при этом непосредственно менять данные мы не собираемся, тогда, это как раз тот случай, когда кортежи придутся как нельзя кстати. Кортежи в памяти занимают меньший объем по сравнению со списками. Кортежи работают быстрее, чем списки

3. Как осуществляется создание кортежей?

a = ()

b = tuple()

4. Как осуществляется доступ к элементам кортежа?

Доступ к элементам кортежа осуществляется также как к элементам списка – через указание индекса.

5. Зачем нужна распаковка (деструктуризация) кортежа?

Обращение по индексу, это не самый удобный способ работы с кортежами. Дело в том, что кортежи часто содержат значения разных типов, и помнить, по какому индексу что лежит — очень непросто.

6. Какую роль играют кортежи в множественном присваивании?

Используя множественное присваивание, можно провернуть интересный трюк: обмен значениями между двумя переменными.

7. Как выбрать элементы кортежа с помощью среза?

С помощью операции взятия среза можно получить другой кортеж.

Общая форма операции взятия среза для кортежа следующая

T2 = T1[i:j]

здесь

• T2 – новый кортеж, который получается из кортежа T1;

• T1 – исходный кортеж, для которого происходит срез;

• i, j – соответственно нижняя и верхняя границы среза. Фактически берутся ко вниманию элементы, лежащие на позициях i, i+1, …, j-1. Значение j определяет позицию за последним элементом среза.

8. Как выполняется конкатенация и повторение кортежей?

Для кортежей можно выполнять операцию конкатенации, которая обозначается символом +.

T3 = T1 + T2

9. Как выполняется обход элементов кортежа?

Элементы кортежа можно последовательно просмотреть с помощью операторов цикла while или for.

10. Как проверить принадлежность элемента кортежу?

Проверка вхождения элемента в кортеж - оператор in.

11. Какие методы работы с кортежами Вам известны?

index(), count().

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

Допустимо.

13. Как создать кортеж с помощью спискового включения.

Также как и список