

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ  
ФЕДЕРАЦИИ**

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

**Кафедра инфокоммуникаций  
«Работа с кортежами в языке Python»**

**Отчет по лабораторной работе № 2.5  
по дисциплине «Основы программной инженерии»**

Выполнил студент группы ПИЖ-б-о-21-1  
Кучеренко С. Ю. « » 2022г.

Подпись студента\_\_\_\_\_

Работа защищена « »\_\_\_\_\_2022г.

Проверил Воронкин Р.А. \_\_\_\_\_  
(подпись)

Ставрополь 2022

**Цель работы:** приобретение навыков по работе с кортежами при написании программ с помощью 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.

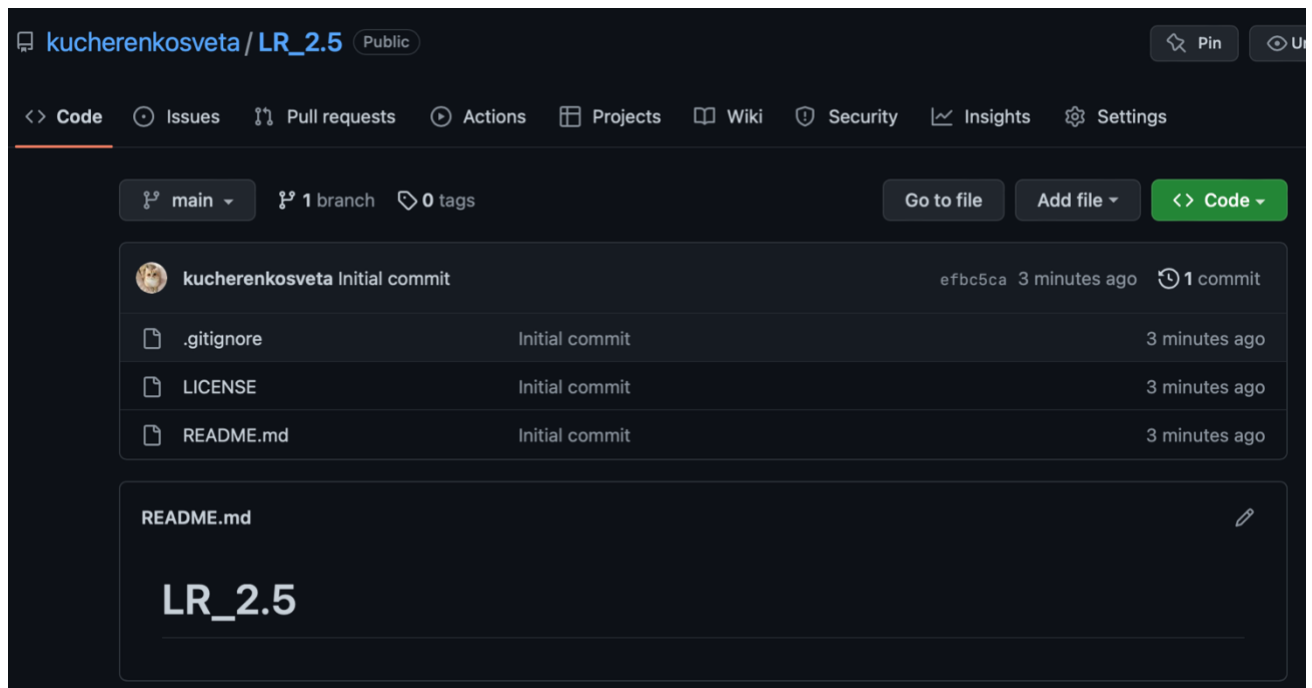


Рисунок 1 – Создание репозитория

### 3. Выполните клонирование созданного репозитория.

```
[(base) svetik@MacBook-Air-Svetik ~ % cd ~/Desktop/Laba8 ]
[(base) svetik@MacBook-Air-Svetik Laba8 % git clone https://github.com/kucherenko]
sveta/LR_2.5.git
Cloning into 'LR_2.5'...
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 Laba8 %
```

Рисунок 2 – Клонирование репозитория

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

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

```
[(base) svetik@MacBook-Air-Svetik LR_2.5 % 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/Laba8/LR_2.5/.git/hooks]
```

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

## 6. Создайте проект PyCharm в папке репозитория

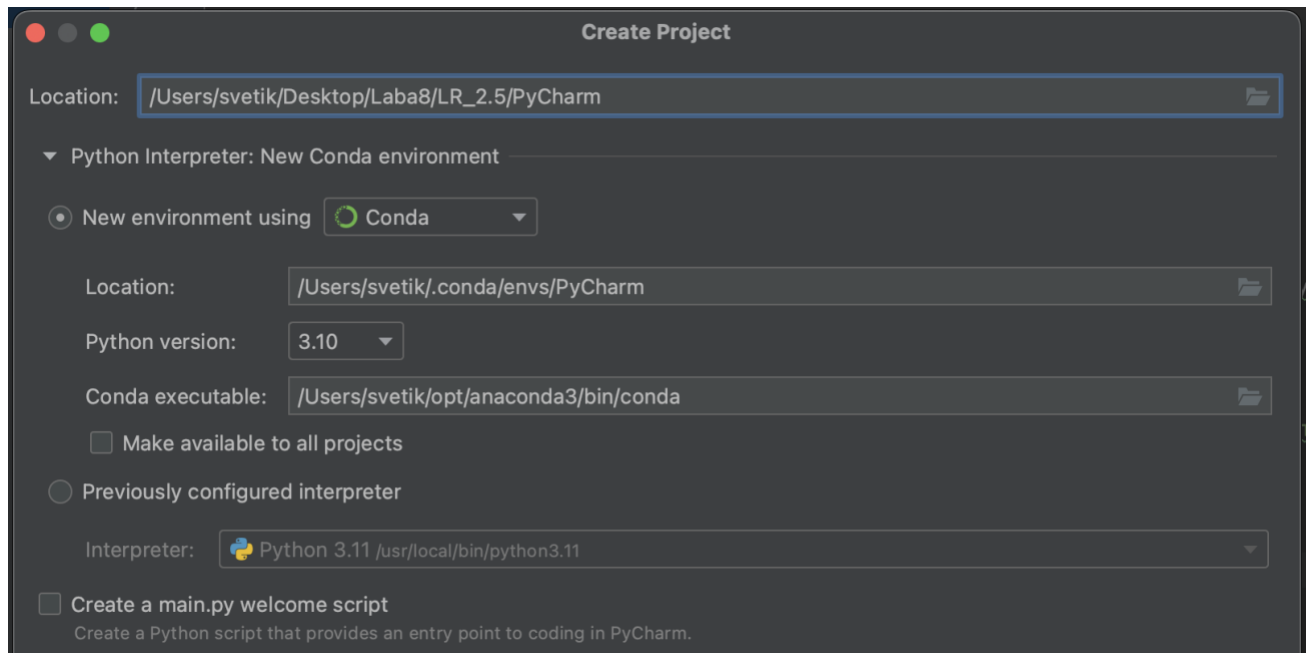


Рисунок 4 – Создание проекта PyCharm в папке репозитория

7. Проработайте примеры лабораторной работы. Создайте для каждого примера отдельный модуль языка Python. Зафиксируйте изменения в репозитории.

**Пример 1.** Ввести кортеж A из 10 элементов, найти сумму элементов, меньших по модулю 5, и вывести ее на экран. Использовать в программе вместо списков кортежи.

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

import sys

if __name__ == '__main__':
    # Ввести кортеж одной строкой.
    A = tuple(map(int, input().split()))
    # Проверить количество элементов кортежа.
    if len(A) != 10:
        print("Неверный размер кортежа", file=sys.stderr)
        exit(1)
    # Найти искомую сумму.
    s = 0
    for item in A:
        if abs(item) < 5:
            s += item
    print(s)
```

```
/usr/local/bin/python3.11 /Users/svetik/  
1 2 3 4 5 6 7 -5 33 11  
10  
  
Process finished with exit code 0
```

Рисунок 5 – Результат работы программы

Решение через списковые включения:

```
#!/usr/bin/env python3  
# -*- coding: utf-8 -*-  
  
import sys  
  
if __name__ == '__main__':  
    # Ввести список одной строкой.  
    A = list(map(int, input().split()))  
    # Проверить количество элементов списка.  
    if len(A) != 10:  
        print("Неверный размер списка", file=sys.stderr)  
        exit(1)  
  
    # Найти искомую сумму.  
    s = sum(a for a in A if abs(a) < 5)  
    print(s)
```

```
/usr/local/bin/python3.11 /Users/svet  
12 3 -5 24 5 2 2 11 43 5  
7  
  
Process finished with exit code 0
```

Рисунок 8 – Результат работы программы

8. Выполните индивидуальные задания, согласно своему варианту.

Вариант 9. Если в кортеже есть хотя бы одна тройка соседних чисел, в которой средний элемент больше своих «соседей», т. е. предшествующего и последующего, то напечатать все элементы, предшествующие элементам последней из таких троек.

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

if __name__ == '__main__':
    # Ввести кортеж одной строкой.
    A = tuple(map(int, input().split()))

    for item in range(len(A)-2, 1, -1):
        if A[item-1] < A[item] and A[item] > A[item+1]:
            print(A[:item - 1])
            break
```

```
/usr/local/bin/python3.11 /Users/sve
1 7 8 9 10 1
(1, 7, 8)

Process finished with exit code 0
```

Рисунок 9 – Результат работы программы

## **Вопросы для защиты работы**

### **1. Что такое списки в языке Python?**

Кортеж (tuple) – это неизменяемая структура данных, которая по своему подобию очень похожа на список. Как вы наверное знаете, список – это изменяемый тип данных. Т.

### **2. Каково назначение кортежей в языке Python?**

Т.к. кортеж – неизменяемый тип данных, это защищает данные от случайного изменения. К тому же, из-за отсутствия операций изменения делают работу кортежей быстрее и место они занимают меньше, чем списки.

### **3. Как осуществляется создание кортежей?**

Создание кортежей осуществляется следующим образом:

```
a = ()
```

```
b = tuple()
```

В скобках через запятую перечисляются элементы кортежа

Если кортеж состоит из одного элемента, то после него нужно поставить «,»:

```
tuple = (42,)
```

### **4. Как осуществляется доступ к элементам кортежа?**

Доступ к элементам кортежа осуществляется аналогично доступу к элементам списка – через индекс.

```
>>> a = (1, 2, 3, 4, 5)
```

```
>>> print(a[0])
```

```
1
```

### **5. Зачем нужна распаковка (деструктуризация) кортежа?**

Это облегчает доступ к элементам, осуществляя его не по индексу, а с помощью переменных

```
name_and_age = ('Bob', 42)
```

```
(name, age) = name_and_age
```

name # 'Bob'

age # 42



## 6. Какую роль играют кортежи в множественном присваивании?

Благодаря тому, что кортежи легко собирать и разбирать, в Python удобно делать такие вещи, как множественное присваивание. Смотрите:

Используя множественное присваивание, можно проверить интересный трюк: обмен значениями между двумя переменными. Вот код:

```
a = 100
```

```
b = 'foo'
```

```
(a, b) = (b, a)
```

```
a # 'foo'
```

```
b # 100
```

Строку `(a, b) = (b, a)` нужно понимать как "присвоить в `a` и `b` значения из кортежа, состоящего из значений переменных `b` и `a`".

## 7. Как выбрать элементы кортежа с помощью среза? Аналогично со списком:

```
>>> a = (1, 2, 3, 4, 5)
```

```
>>> print(a[1:3])
```

```
(2, 3)
```

## 8. Как выполняется конкатенация и повторение кортежей?

Конкатенация обозначается знаком «+», по сути, эта операция объединяет несколько кортежей в один новый. форма:  $T3 = T1 + T2$ , где  $T1, T2$  – кортежи, над которыми выполняется операция, а  $T3$  – новый кортеж.

## 9. Как выполняется обход элементов кортежа?

Обход элементов кортежа можно осуществить с помощью циклов `for` или `while`:

```
for i in A:
```

```
while i < len(A):
```

```
    i = i + 1
```

### **10. Как проверить принадлежность элемента кортежу?**

Чтобы проверить принадлежность элемента кортежу необходимо воспользоваться операцией `in`, форма: `a = i in A`, где `a = true/false`, `i` – искомый элемент, `A` – кортеж, в котором осуществляется поиск.

### **11. Какие методы работы с кортежами Вам известны?**

Метод `index()` осуществляет поиск позиции элемента в кортеже, форма: `pos = T.index(i)`, где `pos` – переменная, в которую будет записан индекс, `T` – кортеж в котором осуществляется поиск, `i` – искомый элемент.

Метод `count()` осуществляет подсчет количества вхождений элемента в кортеж, форма: `k = T.count(i)`, где `k` – искомое количество, `T` – кортеж в котором осуществляется поиск, `i` – искомый элемент.

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

Да

### **13. Как создать кортеж с помощью спискового включения.**

Аналогично списку, только скобки не квадратные, а круглые, и выражение дает на выходе специальный объект генератора, а не кортеж, для преобразования которого необходимо воспользоваться вызовом `tuple()`.