

**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  
«Программирование на языке Python»**

Выполнил студент группы ИВТ-б-о-22-1

Сумин Никита Сергеевич.

« » \_\_\_\_\_ 2023г.

Подпись студента \_\_\_\_\_

Работа защищена « » \_\_\_\_\_ 20\_\_ г.

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

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

**Порядок выполнения работы:**

1. Создал общедоступный репозиторий на GitHub, в котором использована лицензия MIT и язык программирования Python.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner \* / Repository name \*

BrandooDi / mywork8

Great repository names are short and memorable. Need inspiration? How about [solid-disco?](#)

Description (optional)

☒ Public  
Anyone on the internet can see this repository. You choose who can commit.

☐ Private  
You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

☐ Add a README file  
This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore

Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: Python

Choose a license

A license tells others what they can and can't do with your code. [Learn more.](#)

License: MIT License

① You are creating a public repository in your personal account.

Create repository

Рисунок 1 - Создание репозитория

2. Выполните клонирование созданного репозитория.

```
MINGW64:/d/www/mywork8
Compressing objects: 100% (8/8), done.
Writing objects: 100% (8/8), 238.06 KiB | 34.01 MiB/s, done.
Total 8 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/BrandooDi/mywork7.git
   9ca9779..db67bd3  main -> main

user@MINGW64 /d/www/mywork7 (main)
$ cd d:www

user@MINGW64 /d/www
$ git clone https://github.com/BrandooDi/mywork8.git
Cloning into 'mywork8'...
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 7 (delta 1), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (7/7), 4.18 KiB | 4.18 MiB/s, done.
Resolving deltas: 100% (1/1), done.

user@MINGW64 /d/www
$ cd d:www/mywork8

user@MINGW64 /d/www/mywork8 (main)
$
```

Рисунок 2 - Клонирование репозитория

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

```
MINGW64:/d/www/mywork8
Resolving deltas: 100% (1/1), done.

user@MINGW64 /d/www
$ cd d:www/mywork8

user@MINGW64 /d/www/mywork8 (main)
$ 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? [D:/www/mywork8/.git/hooks]

user@MINGW64 /d/www/mywork8 (develop)
$
```

Рисунок 3 - Ветвление по модели git-flow 4.

Создайте проект PyCharm в папке репозитория.

Имя	Дата изменения	Тип	Размер
.git	13.09.2022 21:39	Папка с файлами	
PyCharm Project	13.09.2022 21:42	Папка с файлами	
.gitignore	13.09.2022 21:35	Текстовый докум...	6 КБ
LICENSE	13.09.2022 21:35	Файл	2 КБ



Рисунок 4 - Проект PyCharm

5. 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.

```
"D:\www\mywork8\PyCharm Project\venv\Scripts\python.exe" "D:/www/mywork8/PyCharm Project/main.py"
0 1 2 3 4 5 6 7 8 9
10
Process finished with exit code 0
|
```

Рисунок 5 – Результат выполнения программы

6. Выполните индивидуальное задание .

Известно количество очков, набранных каждой из 20 команд – участниц первенства по футболу. Перечень очков дан в порядке убывания (ни одна пара команд не набрала одинаковое количество очков). Определить, какое место заняла команда, набравшая очков (естественно, что значение имеется в перечне). Условный оператор не использовать.

```
"D:\www\mywork8\PyCharm Project\venv\Scripts\python.exe" "D:/www/mywork8/PyCharm Project/ind1.py"
Введите очки команд:
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
Введите кол-во очков: 12
Команда заняла 9 место
Process finished with exit code 0
|
```

Рисунок 6 - Результат выполнения программы **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? $a = ()$ $b = \text{tuple}()$**

### **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  $i, i+1, \dots, j-1$ . 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?**

Доступно.

**13. Как создать кортеж с помощью спискового включения. Так же как и список**

**Вывод:** приобрел навыки по работе с кортежами при написании программ с помощью языка программирования Python версии 3.x.