

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ
ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего
образования
«СЕВЕРОКАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ
УНИВЕРСИТЕТ»**

**Кафедра
инфокоммуникаций
Институт цифрового
развития**

**ОТЧЁТ
по лабораторной работе №2.5
Дисциплина: «Основы программной инженерии»
Тема: «Работа с кортежами в языке Python»**

Выполнила:
студентка 2 курса
группы Пиж-б-о-21-1
Логвинов Иван
Васильевич

Ставрополь 2022

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

1. Был создан репозиторий в Github в который были добавлены правила gitignore для работы IDE PyCharm, была выбрана лицензия MIT, сам репозиторий был клонирован на локальный сервер и был организован в соответствии с моделью ветвления git-flow.

Owner * / Repository name * ✓

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

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.](#)

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

This will set **main** as the default branch. Change the default name in your [settings](#).

You are creating a public repository in your personal account.

Рисунок 1.1 – Создание репозитория

```
Командная строка
Microsoft Windows [Version 10.0.18363.418]
(c) Корпорация Майкрософт (Microsoft Corporation), 2019. Все права защищены.

C:\Users\Иван>cd desktop

C:\Users\Иван\Desktop>git clone https://github.com/IvanLogvinov11/lab2_5.git
Cloning into 'lab2_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.

C:\Users\Иван\Desktop>
```

Рисунок 1.2 – Клонирование репозитория

2. Был создана папка PyCharm в которой хранятся примеры из лабораторной работы.

Имя	Дата изменения	Тип	Размер
pycharm	08.12.2022 19:13	Папка с файлами	
.gitignore	08.12.2022 19:05	Текстовый докум...	2 КБ
LICENSE	08.12.2022 19:05	Файл	2 КБ
README	08.12.2022 19:05	Файл "MD"	1 КБ

Рисунок 2.1 – Папка PyCharm для примеров

```
C:\Users\Иван\AppData\Local\Programs\Python\Pyt
3 5 1 2 6 7 3 9 1 3
13

Process finished with exit code 0
|
```

Рисунок 2.2 – Результат работы первого примера

3. Было выполнено индивидуальное задание согласно 5 варианту.

Задание для 10 варианта

Если в кортеже есть хотя бы одна пара одинаковых соседних элементов, то напечатать все элементы, следующие за элементами первой из таких пар.

```

import random

array = []
result = None
size = int(input('Введите размер последовательности: '))

for i in range(1, size + 1):
    array.append(random.randrange(1, 10000))
print(array)

for i in range(len(array)):
    if (array[i]) > (array[i + 1]):
        result = array[i + 1]
        break

if result is None:
    print('Последовательность упорядочена')
else:
    print('Первое число, нарушающее упорядоченность: ', result)

```

```

Введите размер последовательности: 14
[5387, 9582, 4222, 5870, 9913, 6614, 2263, 1777, 4874, 3259, 2739, 2725, 172, 1905]
Первое число, нарушающее упорядоченность: 4222

Process finished with exit code 0
|

```

Рисунок 3.1 – Результат работы программы

4. Было осуществлен коммит и слияние веток main и develop, также запущены изменения на удаленный сервер.

```

C:\Users\Иван\Desktop\lab2_5>git add .
warning: in the working copy of 'pycharm/.idea/inspectionProfiles/profiles_settings.xml',
C:\Users\Иван\Desktop\lab2_5>git commit -m "new"
[develop 8979441] new
 8 files changed, 71 insertions(+)
 create mode 100644 pycharm/.idea/.gitignore
 create mode 100644 pycharm/.idea/inspectionProfiles/profiles_settings.xml
 create mode 100644 pycharm/.idea/misc.xml
 create mode 100644 pycharm/.idea/modules.xml
 create mode 100644 pycharm/.idea/pycharm.iml
 create mode 100644 pycharm/.idea/vcs.xml
 create mode 100644 pycharm/ind.py
 create mode 100644 pycharm/p1.py

C:\Users\Иван\Desktop\lab2_5>git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

C:\Users\Иван\Desktop\lab2_5>git merge develop
Updating 94df898..8979441
Fast-forward
 pycharm/.idea/.gitignore          | 3 +++
 .../idea/inspectionProfiles/profiles_settings.xml | 6 +++++
 pycharm/.idea/misc.xml           | 4 ++++
 pycharm/.idea/modules.xml        | 8 ++++++++
 pycharm/.idea/pycharm.iml        | 8 ++++++++
 pycharm/.idea/vcs.xml            | 6 ++++++
 pycharm/ind.py                   | 19 +++++++++++++++++++++
 pycharm/p1.py                    | 17 ++++++++++++++++++
 8 files changed, 71 insertions(+)
 create mode 100644 pycharm/.idea/.gitignore
 create mode 100644 pycharm/.idea/inspectionProfiles/profiles_settings.xml
 create mode 100644 pycharm/.idea/misc.xml
 create mode 100644 pycharm/.idea/modules.xml
 create mode 100644 pycharm/.idea/pycharm.iml
 create mode 100644 pycharm/.idea/vcs.xml
 create mode 100644 pycharm/ind.py
 create mode 100644 pycharm/p1.py

C:\Users\Иван\Desktop\lab2_5>

```

Рисунок 4.1 – Коммит изменений и слияние веток main и develop

```

C:\Users\Иван\Desktop\lab2_5>git push
Enumerating objects: 14, done.
Counting objects: 100% (14/14), done.
Delta compression using up to 8 threads
Compressing objects: 100% (11/11), done.
Writing objects: 100% (13/13), 2.11 KiB | 1.05 MiB/s, done.
Total 13 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/IvanLogvinov11/lab2_5.git
 94df898..8979441  main -> main

C:\Users\Иван\Desktop\lab2_5>

```

Рисунок 4.2 – Пуш на удаленный сервер

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

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 проверить 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.