

**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 кроссплатформенного программирования»

Тема: «Работа с кортежами в языке Python»

Выполнил: студент 1 курса

группы ИВТ-б-о-21-1

Назаров Никита Юрьевич

Ставрополь 2022

Выполнение работы:

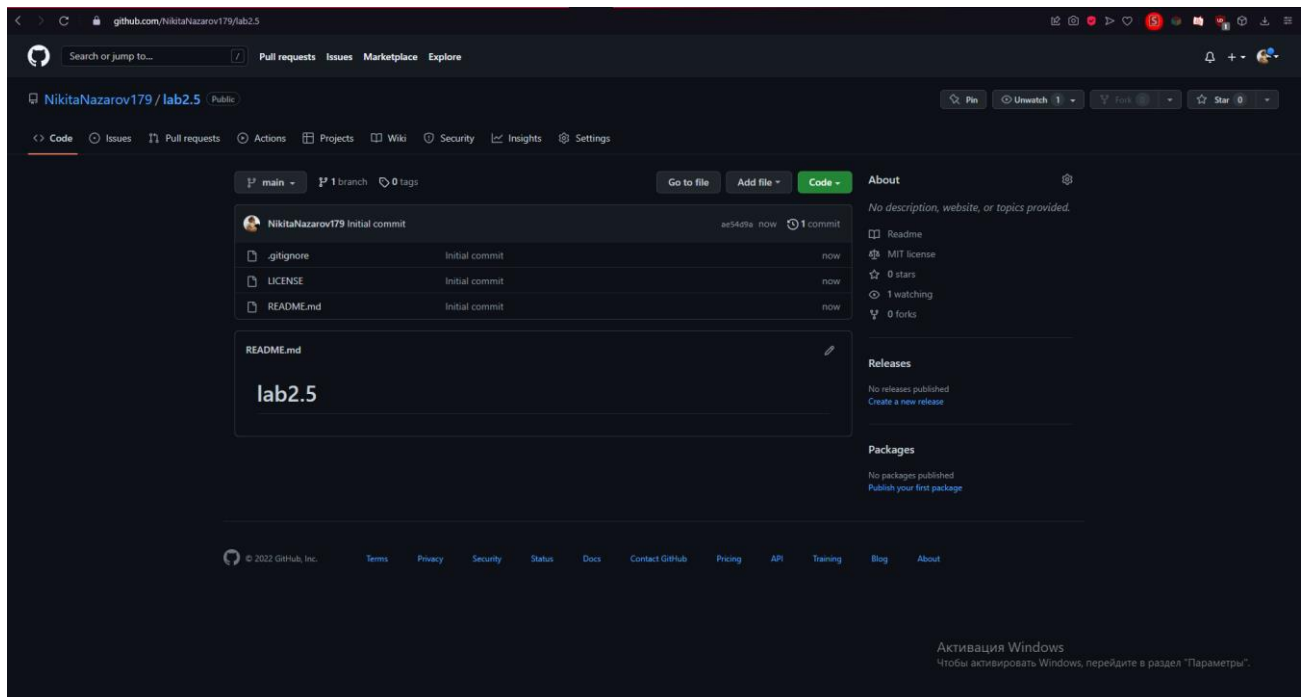


Рисунок 1 – новый репозиторий

```
H:\cross\git\nazarov>git clone https://github.com/NikitaNazarov179/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.
```

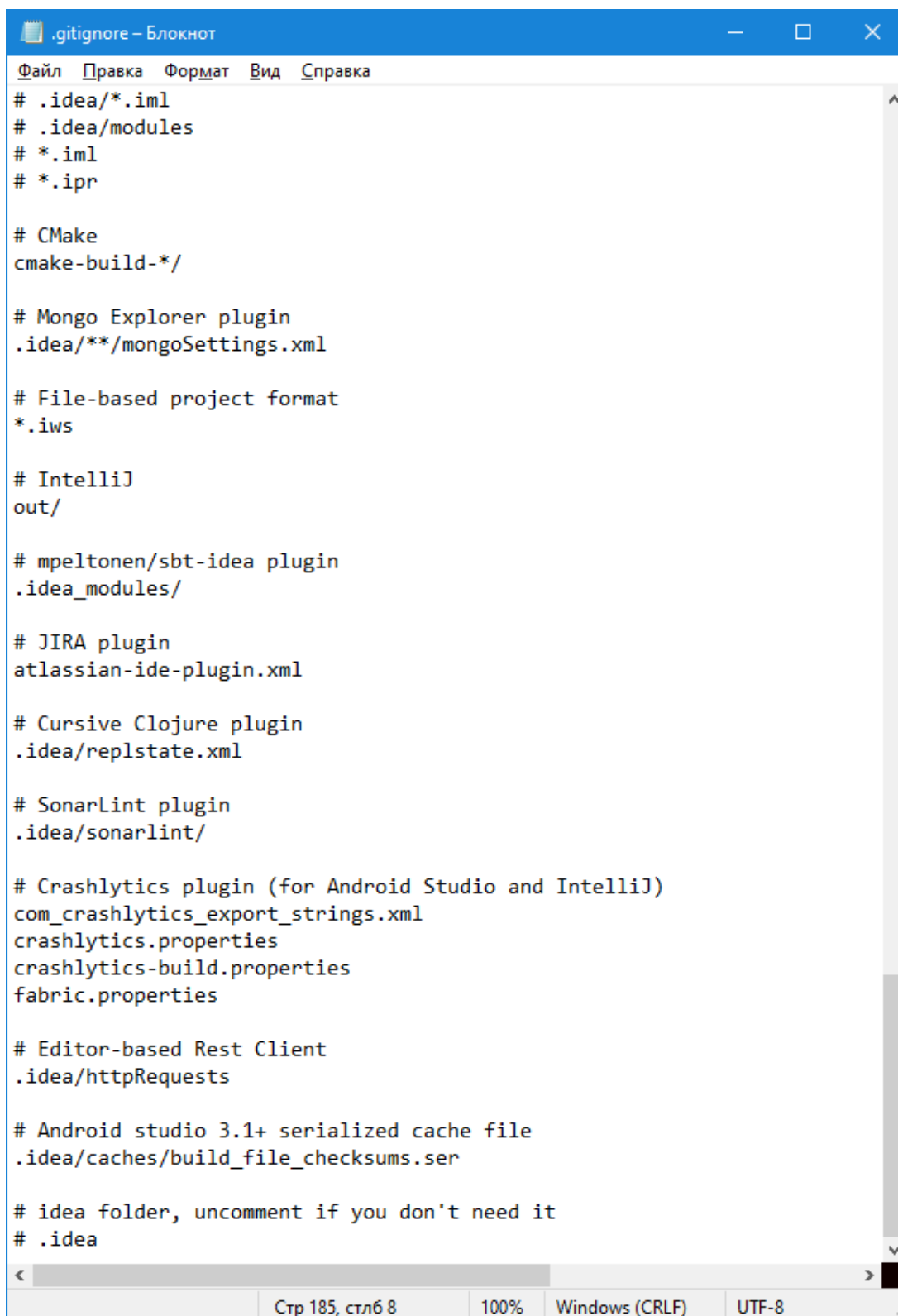
Рисунок 2 – клонирование репозитория

```
H:\cross\git\nazarov\lab2.5>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? [H:/cross/git/nazarov/lab2.5/.git/hooks]
```

Рисунок 3 – организация репозитория в соответствии с моделью git flow



```
.gitignore - Блокнот
Файл  Правка  Формат  Вид  Справка
# .idea/*.iml
# .idea/modules
# *.iml
# *.ipr

# CMake
cmake-build-*/

# Mongo Explorer plugin
.idea/**/mongoSettings.xml

# File-based project format
*.iws

# IntelliJ
out/

# mpeltonen/sbt-idea plugin
.idea_modules/

# JIRA plugin
atlassian-ide-plugin.xml

# Cursive Clojure plugin
.idea/replstate.xml

# SonarLint plugin
.idea/sonarlint/

# Crashlytics plugin (for Android Studio and IntelliJ)
com_crashlytics_export_strings.xml
crashlytics.properties
crashlytics-build.properties
fabric.properties

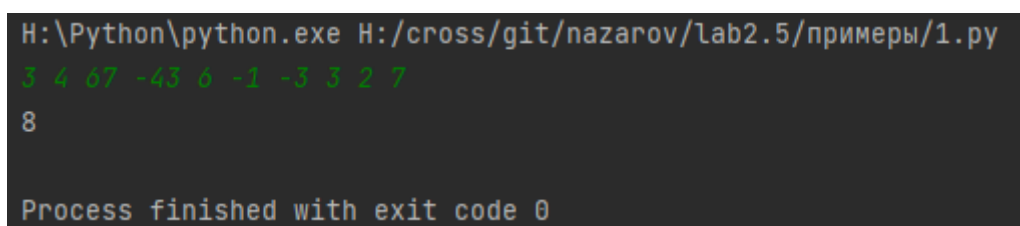
# Editor-based Rest Client
.idea/httpRequests

# Android studio 3.1+ serialized cache file
.idea/caches/build_file_checksums.ser

# idea folder, uncomment if you don't need it
# .idea
```

Стр 185, стр 6 8 100% Windows (CRLF) UTF-8

Рисунок 4 – редактирование gitignore



```
H:\Python\python.exe H:/cross/git/nazarov/lab2.5/примеры/1.py
3 4 67 -43 6 -1 -3 3 2 7
8

Process finished with exit code 0
```

Рисунок 5 – проработка примера 1

```

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

#Определить, является ли кортеж упорядоченным по возрастанию.
#В случае отрицательного ответа определить номер первого
#элемента, нарушающего такую упорядоченность.

if __name__ == '__main__':
    array = tuple(map(int, input().split()))
    result = None

    for n, i in enumerate(array):
        if (array[n] > (array[n + 1])):
            result = array[n + 1]
            break

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

```

Рисунок 6 – программа для решения индивидуального задания

```

1 2 3 4 5 6 7 8 9 2435 6
Первое число, нарушающее упорядоченность: 6

```

Рисунок 7 – результат работы программы

```

H:\cross\git\nazarov\lab2.5>git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 1 commit.
(use "git push" to publish your local commits)

H:\cross\git\nazarov\lab2.5>git merge develop
Updating 55ca7ab..5b6879b
Fast-forward
 .../1.py" | 20 ++++++
 .../1.py" | 21 ++++++
2 files changed, 41 insertions(+)
create mode 100644 "\320\270\320\275\320\264.\320\267\320\260\320\264\320\260\320\275\320\270\321\217\1.py"
create mode 100644 "\320\277\321\200\320\270\320\274\320\265\321\200\321\213\1.py"

H:\cross\git\nazarov\lab2.5>

```

Рисунок 8 – слияние веток

```
H:\cross\git\nazarov\lab2.5>git push
Enumerating objects: 13, done.
Counting objects: 100% (13/13), done.
Delta compression using up to 12 threads
Compressing objects: 100% (9/9), done.
Writing objects: 100% (11/11), 3.06 KiB | 1.53 MiB/s, done.
Total 11 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), done.
To https://github.com/NikitaNazarov179/lab2.5.git
   ae54d9a..5b6879b  main -> main

H:\cross\git\nazarov\lab2.5>_
```

Рисунок 9 – пуш веток на удаленный репозиторий

Контр. вопросы и ответы на них:

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.