

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

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

**Отчет по лабораторной работе № 2.15**  
**по дисциплине «Программирование на Python»**

Выполнил студент группы ИВТ-б-о-21-1

Сумин Никита Сергеевич.

«» \_\_\_\_\_ 2022г.

Подпись студента \_\_\_\_\_

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

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

**Цель работы:** приобретение навыков по работе с текстовыми файлами 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 \*

 BrandooDi ▾

Repository name \*

/ mywork2.15 ✓

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

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 ▾

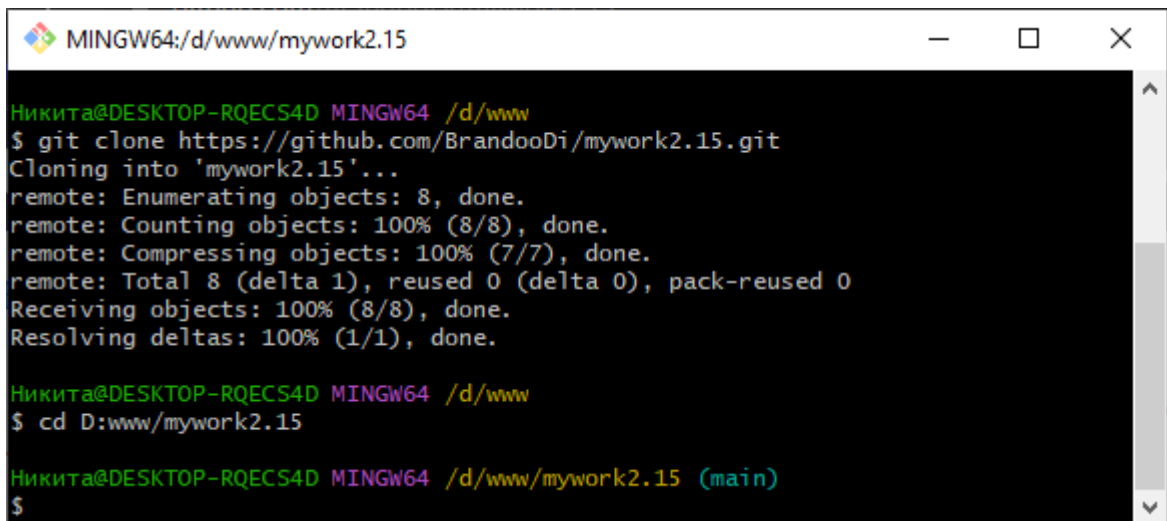
This will set  main as the default branch. Change the default name in your [settings](#).

 You are creating a public repository in your personal account.

Create repository

Рисунок 1 - Создание репозитория

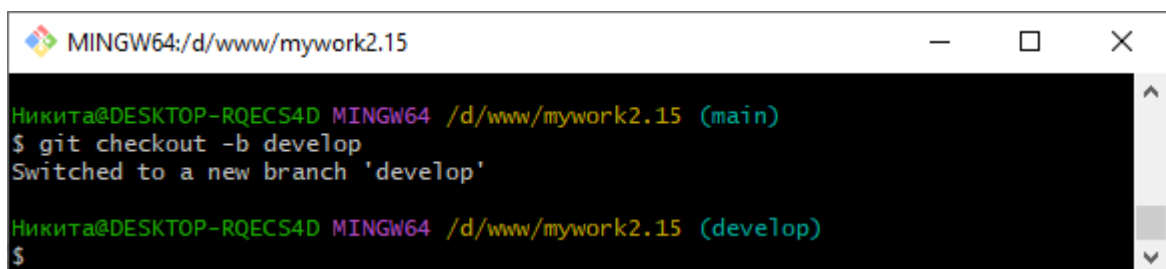
2. Выполните клонирование созданного репозитория.



```
MINGW64:/d/www/mywork2.15
Никита@DESKTOP-RQEC54D MINGW64 /d/www
$ git clone https://github.com/BrandooDi/mywork2.15.git
Cloning into 'mywork2.15'...
remote: Enumerating objects: 8, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 8 (delta 1), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (8/8), done.
Resolving deltas: 100% (1/1), done.
Никита@DESKTOP-RQEC54D MINGW64 /d/www
$ cd D:\www\mywork2.15
Никита@DESKTOP-RQEC54D MINGW64 /d/www/mywork2.15 (main)
$
```

Рисунок 2 - Клонирование репозитория

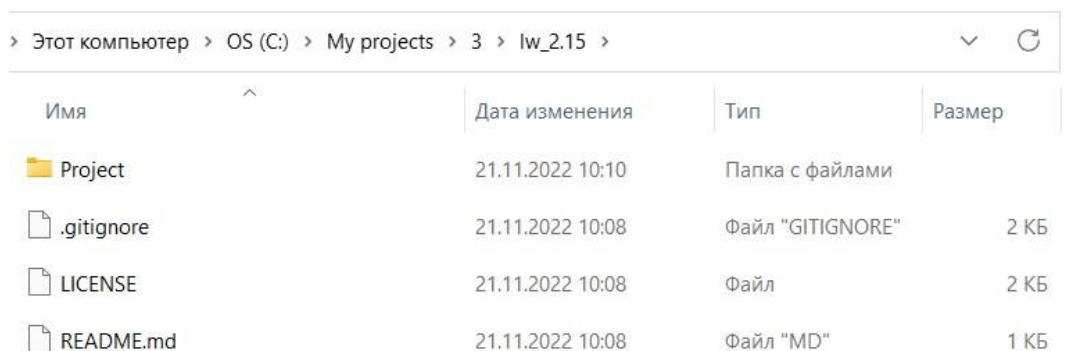
3. Организуйте свой репозиторий в соответствии с моделью ветвления git-flow.



```
MINGW64:/d/www/mywork2.15
Никита@DESKTOP-RQEC54D MINGW64 /d/www/mywork2.15 (main)
$ git checkout -b develop
Switched to a new branch 'develop'
Никита@DESKTOP-RQEC54D MINGW64 /d/www/mywork2.15 (develop)
$
```

Рисунок 3 - Ветвление по модели git-flow 4.

Создайте проект PyCharm в папке репозитория.



Этот компьютер > OS (C:) > My projects > 3 > lw_2.15 >				
Имя	Дата изменения	Тип	Размер	
Project	21.11.2022 10:10	Папка с файлами		
.gitignore	21.11.2022 10:08	Файл "GITIGNORE"	2 КБ	
LICENSE	21.11.2022 10:08	Файл	2 КБ	
README.md	21.11.2022 10:08	Файл "MD"	1 КБ	

Рисунок 4 - Создание проекта 5.

Проработать примеры лабораторной работы.

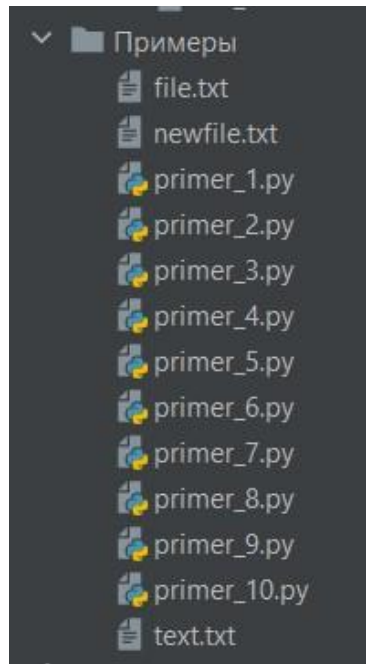


Рисунок 5 - Выполненные примеры

6. Выполнить индивидуальные задания.

**Индивидуальное задание 1.** Написать программу, которая считывает из текстового файла три предложения и выводит их в обратном порядке.

```
D:\www\mywork2.15\Project\venv\Scripts\python.exe D:\www\mywork2.15\Project\ind1.py
Files

number 2.15|

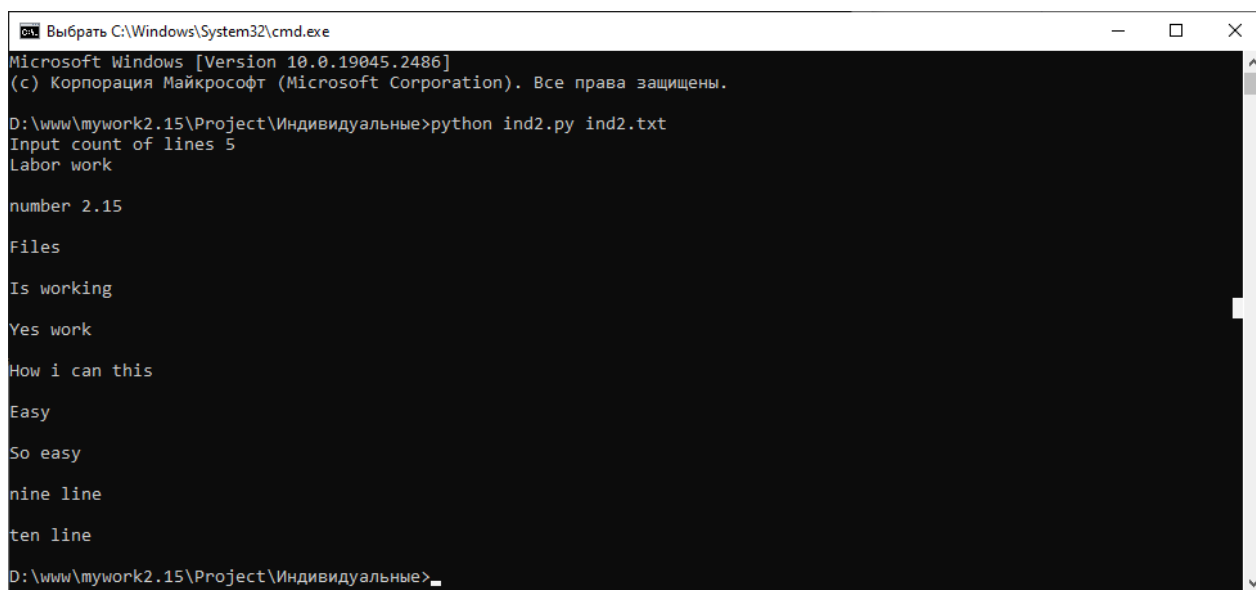
Labor work

Process finished with exit code 0
```

Рисунок 6 - Результат выполнения индивидуального задания 1

**Индивидуальное задание 2.** В операционных системах на базе Unix обычно присутствует утилита с названием head. Она выводит первые десять строк содержимого файла, имя которого передается в качестве аргумента командной строки. Напишите программу на Python, имитирующую поведение этой утилиты. Если файла, указанного пользователем, не существует, или не

задан аргумент командной строки, необходимо вывести соответствующее сообщение об ошибке.



```
Выбрать C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19045.2486]
(c) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

D:\www\mywork2.15\Project\Индивидуальные>python ind2.py ind2.txt
Input count of lines 5
Labor work

number 2.15

Files

Is working

Yes work

How i can this

Easy

So easy

nine line

ten line

D:\www\mywork2.15\Project\Индивидуальные>
```

Рисунок 7 - Результат выполнения индивидуального задания 2

**Индивидуальное задание 3.** Самостоятельно подберите или придумайте задачу для работы с изученными функциями модуля os. Приведите решение этой задачи.

**Задача:** создать новый текстовый файл, затем требуется узнать число ядер процессора, а потом записать их в созданный файл, изменить имя файла на myprocessor.txt если его еще не существует, если он существует, то выдать соответствующее сообщение.

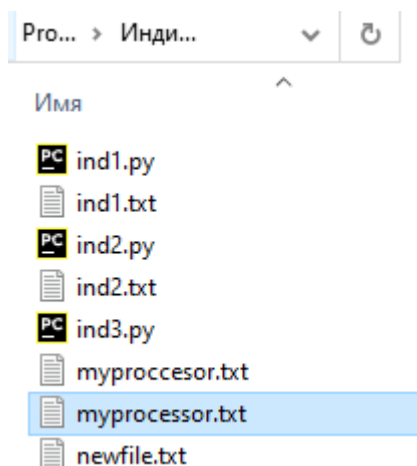


Рисунок 7 – Созданный файл в папке с заданиями

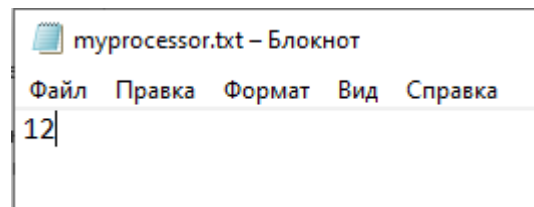


Рисунок 8 – Результат выполнения программы

### Контрольные вопросы:

#### 1. Как открыть файл в языке Python только для чтения?

`file object = open(<file-name>, <access-mode>, <buffering>)` `access-mode = r` - открывает файл в режиме только для чтения.

Указатель файла существует в начале.

Файл по умолчанию открывается в этом режиме, если не передан режим доступа.

#### 2. Как открыть файл в языке Python только для записи?

`file object = open(<file-name>, <access-mode>, <buffering>)`

`access-mode = w` - только для записи. Он перезаписывает файл, если он существовал ранее, или создает новый, если файл с таким именем не существует. Указатель имеется в начале файла.

#### 3. Как прочитать данные из файла в языке Python?

Чтобы прочитать файл с помощью сценария Python, Python предоставляет метод `read()`.

Метод `read()` считывает строку из файла. Он может читать данные как в текстовом, так и в двоичном формате.

Python упрощает чтение файла построчно с помощью метода `readline()`. Метод `readline()` читает строки файла с самого начала, т. е. если мы используем его два раза, мы можем получить первые две строки файла.

#### 4. Как записать данные в файл в языке Python?

Чтобы записать текст в файл, нам нужно открыть файл с помощью метода `open` с одним из следующих режимов доступа.

'w': он перезапишет файл, если какой-либо файл существует. Указатель файла находится в начале файла.

'a': добавит существующий файл. Указатель файла находится в конце файла. Он создает новый файл, если файл не существует.

## **5. Как закрыть файл в языке Python?**

После того, как все операции будут выполнены с файлом, мы должны закрыть его с помощью нашего скрипта Python, используя метод `close()` .

Любая незаписанная информация уничтожается после вызова метода `close()` для файлового объекта.

## **6. Изучите самостоятельно работу конструкции `with ... as`.**

**Каково ее назначение в языке Python? Где она может быть использована еще, помимо работы с файлами?**

Конструкция `with ... as` используется для оборачивания выполнения блока инструкций менеджером контекста. Используется в сценарии, когда пара операторов должна выполняться с блоком кода между ними.

Преимущество использования оператора `with` заключается в том, что он обеспечивает гарантию закрытия файла независимо от того, как закрывается вложенный блок.

## **7. Изучите самостоятельно документацию Python по работе с**

**файлами. Какие помимо рассмотренных существуют методы записи/чтения информации из файла?**

**Функция `write()`**

Функция `write()` используется для записи в файлы Python, открытые в режиме записи.

С помощью метода `writelines()` можно записать в файл итерируемую последовательность.

## **8. Какие существуют, помимо рассмотренных, функции модуля `os` для работы с файловой системой?**

Функцию `os.replace()` можно использовать для перемещения файлов или каталогов.

Функция `os.listdir()` возвращает список, который содержит имена файлов в папке.

`os.walk()` — это генератор дерева каталогов. Он будет перебирать все переданные составляющие.

Метод `os.path.join()` был использован для объединения текущего пути с именем файла/папки.

Для получения информации о файле в ОС используется функция `os.stat()`, которая выполняет системный вызов `stat()` по выбранному пути.

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