

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №2
дисциплины «Программирование на языке Python»

Вариант_15_

Выполнила:
Маньшина Дарья Алексеевна
2 курс, группа ИТС-б-о-22-1,
11.03.02 «Инфокоммуникационные
технологии и системы связи»,
направленность (профиль)
«Инфокоммуникационные системы и
сети», очная форма обучения

(подпись)

Руководитель практики:
Воронкин Р. А., канд. тех. наук,
доцент кафедры
инфокоммуникаций

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

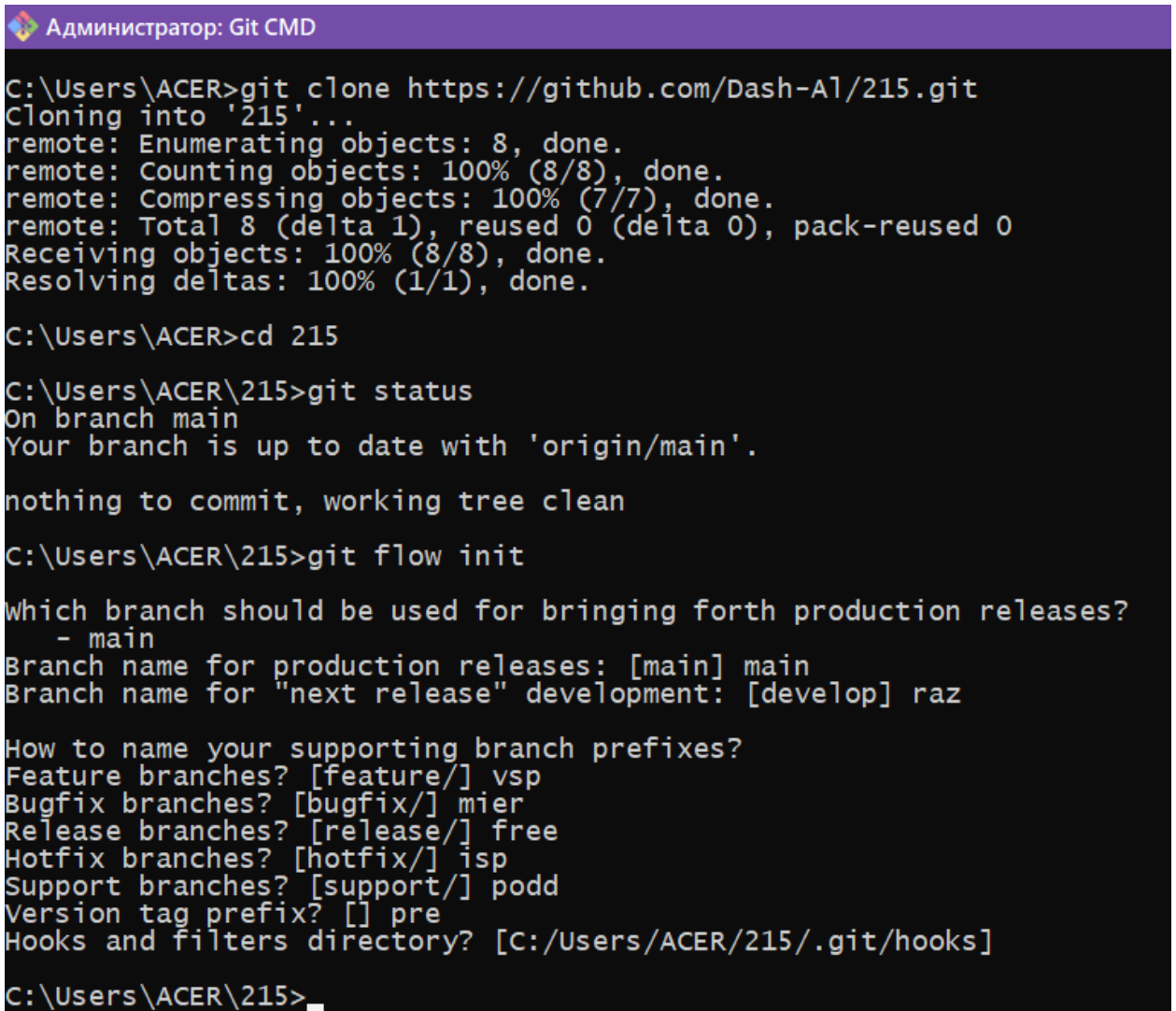
Ставрополь, 2023 г.

Тема: работа с файлами в языке Python.

Цель: приобретение навыков по работе с текстовыми файлами при написании программ с помощью языка программирования Python версии 3.x, изучение основных методов модуля os для работы с файловой системой, получение аргументов командной строки.

Ход работы:

1. Создадим новый репозиторий. Клонировем его и сделаем способ ветвления git-flow.



```
Администратор: Git CMD
C:\Users\ACER>git clone https://github.com/Dash-A1/215.git
Cloning into '215'...
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.

C:\Users\ACER>cd 215

C:\Users\ACER\215>git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean

C:\Users\ACER\215>git flow init

Which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main] main
Branch name for "next release" development: [develop] raz

How to name your supporting branch prefixes?
Feature branches? [feature/] vsp
Bugfix branches? [bugfix/] mier
Release branches? [release/] free
Hotfix branches? [hotfix/] isp
Support branches? [support/] podd
Version tag prefix? [] pre
Hooks and filters directory? [C:/Users/ACER/215/.git/hooks]

C:\Users\ACER\215>_
```

Рисунок 1. Подготовка к лабораторной работе

```
Agusmectpatov: Anaconda Prompt (miniconda2) - conda install -c conda-forge black - conda install -c conda-forge isort - conda install -c conda-forge flake8
Proceed ([y]/n)? y

Downloading and Extracting Packages

Preparing transaction: done
Verifying transaction: done
Executing transaction: done

(215) C:\Users\ACER\215>conda --list
usage: conda-script.py [-h] [-V] command ...
conda-script.py: error: the following arguments are required: command

(215) C:\Users\ACER\215>conda list
# packages in environment at C:\ProgramData\miniconda3\envs\215:
#
# Name                    Version            Build                Channel
black                     23.9.1             py311h1ea47a8_1     conda-forge
bzip2                     1.0.8              he774522_0          conda-forge
ca-certificates           2023.7.22           h56e8100_0          conda-forge
click                     8.1.7              win_pyh7428d3b_0    conda-forge
colorama                  0.4.6              pyhd8ed1ab_0        conda-forge
flake8                    6.1.0              pyhd8ed1ab_0        conda-forge
isort                      5.12.0             pyhd8ed1ab_1        conda-forge
libffi                     3.4.4              hd77b12b_0          conda-forge
mccabe                     0.7.0              pyhd8ed1ab_0        conda-forge
mypy_extensions           1.0.0              pyha778c72_0        conda-forge
openpyxl                   3.0.11             h2b6ff1b_2          conda-forge
packaging                  23.2               pyhd8ed1ab_0        conda-forge
pathspec                   0.11.2             pyhd8ed1ab_0        conda-forge
pip                         23.2.1             py311haa95532_0     conda-forge
platformdirs               3.11.0             pyhd8ed1ab_0        conda-forge
pycodestyle                2.11.1             pyhd8ed1ab_0        conda-forge
pyflakes                   3.1.0              pyhd8ed1ab_0        conda-forge
python                     3.11.5             he1021f5_0          conda-forge
python_abi                 3.11               2_cp311             conda-forge
setuptools                  68.0.0             py311haa95532_0     conda-forge
sqlite                      3.41.2             h2b6ff1b_0          conda-forge
tk                           8.6.12             h2b6ff1b_0          conda-forge
typing_extensions           4.8.0              hd8ed1ab_0          conda-forge
typing_extensions           4.8.0              pyha778c72_0        conda-forge
tzdata                      2023c              h04d1e81_0          conda-forge
vc                           14.2               h21f4f51_1          conda-forge
vs2015_runtime              14.27.29016        h5e58372_2          conda-forge
wheel                       0.41.2             py311haa95532_0     conda-forge
xz                           5.4.2              h8cc25b3_0          conda-forge
zlib                        1.2.13             h8cc25b3_0          conda-forge

(215) C:\Users\ACER\215>conda env export > environment.yml
```

Рисунок 2. Установка пакетов

2. Проработка нескольких примеров из лабораторной работы.

Пример 1. Запись файла.

Программа:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
# open the file2.txt in append mode. Create a new file if no such file exists.
if __name__ == "__main__":
    fileptr = open("file2.txt", "w")
    # appending the content to the file
    fileptr.write(
        "Python is the modern day language. It makes things so simple.\n"
        "It is the fastest-growing programming language"
    )
    # closing the opened the file
    fileptr.close()
```

Результат:

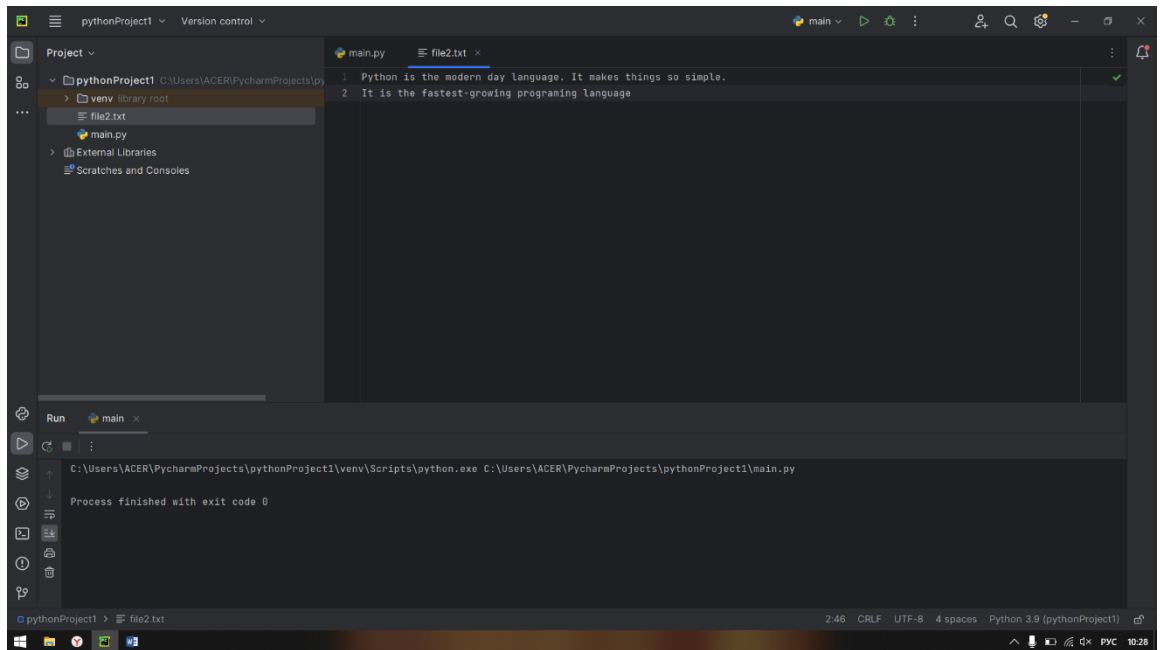


Рисунок 3. Результат работы примера №1

Пример 2.

Программа:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
# open the file.txt in write mode.
if __name__ == "__main__":
    fileptr = open("file2.txt", "a")
    # overwriting the content of the file
    fileptr.write(" Python has an easy syntax and user-friendly interaction.")
    # closing the opened file
    fileptr.close()
```

Результат:

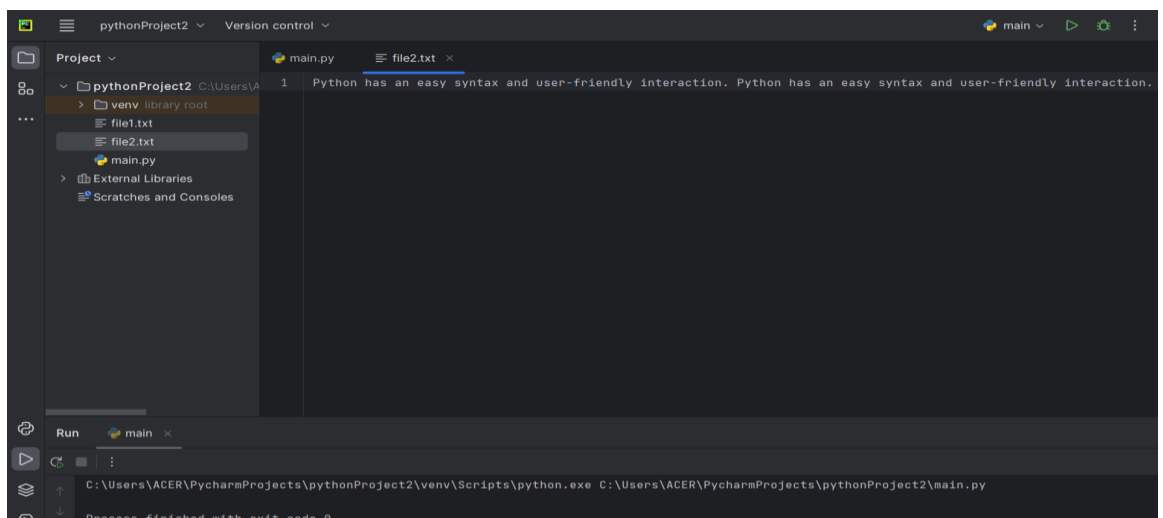


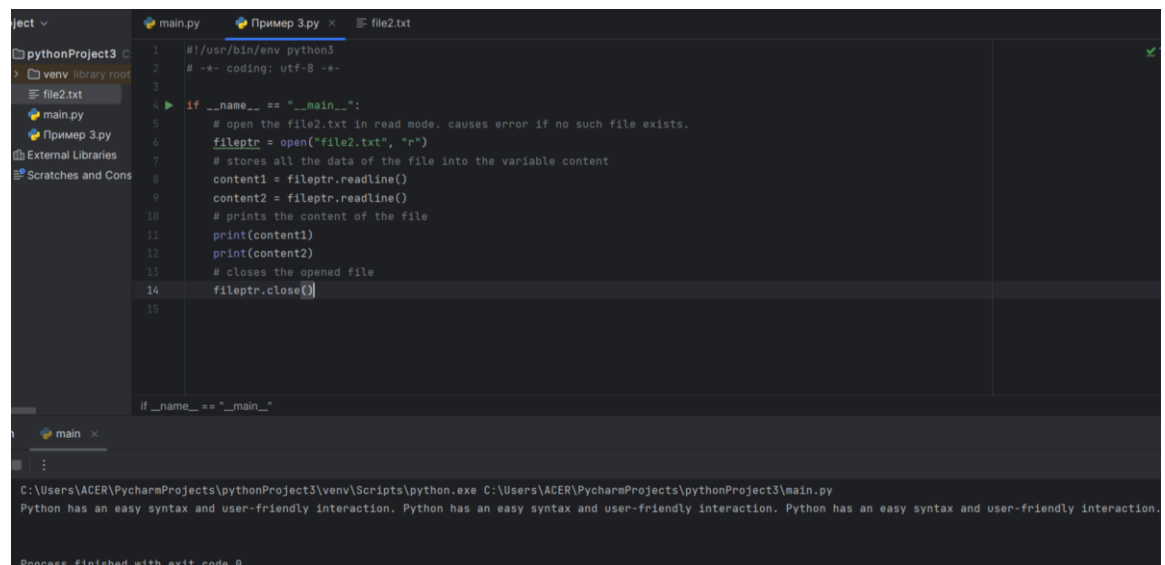
Рисунок 4. Результат работы примера №2

Пример 3. Чтение строк с помощью метода readline().

Программа:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
if __name__ == "__main__":
    # open the file2.txt in read mode. causes error if no such file exists.
    fileptr = open("file2.txt", "r")
    # stores all the data of the file into the variable content
    content1 = fileptr.readline()
    content2 = fileptr.readline()
    # prints the content of the file
    print(content1)
    print(content2)
    # closes the opened file
    fileptr.close()
```

Результат:



```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  if __name__ == "__main__":
5      # open the file2.txt in read mode. causes error if no such file exists.
6      fileptr = open("file2.txt", "r")
7      # stores all the data of the file into the variable content
8      content1 = fileptr.readline()
9      content2 = fileptr.readline()
10     # prints the content of the file
11     print(content1)
12     print(content2)
13     # closes the opened file
14     fileptr.close()
15
16 if __name__ == "__main__":
```

main

C:\Users\ACER\PycharmProjects\pythonProject3\venv\Scripts\python.exe C:\Users\ACER\PycharmProjects\pythonProject3\main.py
Python has an easy syntax and user-friendly interaction. Python has an easy syntax and user-friendly interaction. Python has an easy syntax and user-friendly interaction.

Process finished with exit code 0

Рисунок 5. Результат работы примера №3

3. Выполнение индивидуальных заданий. Вариант 15.

Задание 1.

Условие: написать программу, которая считывает текст из файла и выводит на экран сначала вопросительные, а затем восклицательные предложения.

Программа:

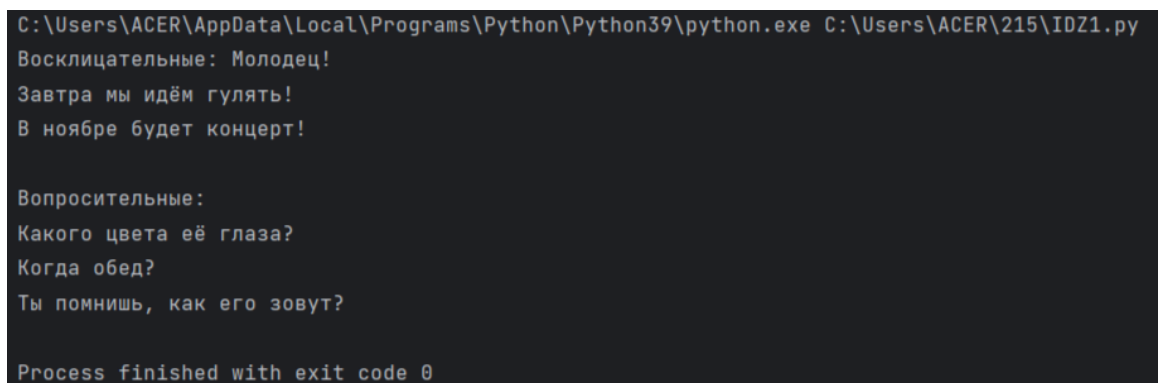
```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
# Условие задания №1
```

```

# написать программу, которая считывает текст из файла и выводит на экран сначала
вопросительные,
# а затем восклицательные предложения.
if __name__ == "__main__":
    sequence = ""
    Vos = "#Восклицательные"
    Vop = "#Вопросительные"
    with open("Zadanie1.txt", 'r', encoding='utf-8') as file:
        text = file.read()
        file.close()
        for i in text:
            sequence += i
            if i == '!':
                Vos+=sequence #Записываем предложение в хранилище
                sequence=""
            elif i == '?':
                Vop+=sequence
                sequence=""
            elif i == '.':
                sequence=""
        print("Восклицательные:", Vos)
        print()
        print("Вопросительные:", Vop)

```

Результат:



```

C:\Users\ACER\AppData\Local\Programs\Python\Python39\python.exe C:\Users\ACER\215\IDZ1.py
Восклицательные: Молодец!
Завтра мы идём гулять!
В ноябре будет концерт!

Вопросительные:
Какого цвета её глаза?
Когда обед?
Ты помнишь, как его зовут?

Process finished with exit code 0

```

Рисунок 6. Результат индивидуального задания №1

Задание 2.

Условие: при написании функций хорошей практикой считается предварение ее блоком комментариев с описанием назначения функции, ее входных параметров и возвращаемого значения. Но некоторые разработчики просто не пишут комментарии к своим функциям. Другие честно собираются написать их когда-нибудь в будущем, но руки так и не доходят.

Напишите программу, которая будет проходить по файлу с исходным кодом на Python и искать функции, не снабженные блоком комментариев. Можно принять за аксиому, что строка, начинающаяся со слова `def`, следом за которым идет пробел, будет считаться началом функции. И если функция документирована, предшествующая строчка должна начинаться со знака `#`. Перечислите названия всех функций, не снабженных комментариями, вместе с именем файла и номером строки, с которой начинается объявление функции. Одно или несколько имен файлов с кодом на языке Python пользователь должен передать в функцию в качестве аргументов командной строки. Для файлов, которые не существуют или не могут быть открыты, должны выдаваться соответствующие предупреждения, после чего должна быть продолжена обработка остальных файлов.

Программа:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
import sys
import os
if __name__ == "__main__":
    def find_functions_without_comments(file_path):
        if not os.path.exists('Zadanie2.txt'):
            print(f'Файл {\'Zadanie2.txt\'} не существует')
            return

        try:
            with open('Zadanie2.txt', 'r') as file:
                lines = file.readlines()

            functions = []
            current_function = None

            for i, line in enumerate(lines):
                line = line.strip()

                if not line.startswith("#") and line.startswith("def"):
                    current_function = line
                elif line.startswith("#") and current_function is not None:
                    functions.append((current_function, i))
                    current_function = None

            if len(functions) > 0:
                print(f'Функции без комментариев в файле {\'Zadanie2.txt\'}:')
                for function, line_number in functions:
                    print(f'{line_number}: {function}')
            else:
                print(f'Все функции в файле {\'Zadanie2.txt\'} имеют комментарии')
```

```
except Exception as e:  
    print(f"Файл обработки ошибок {'Zadanie2.txt'}: {str(e)}")
```

```
if __name__ == "__main__":  
    if len(sys.argv) < 2:  
        print("Путь к файлу не указан")  
    else:  
        for i in range(1, len(sys.argv)):  
            find_functions_without_comments(sys.argv[i])
```

Результат:

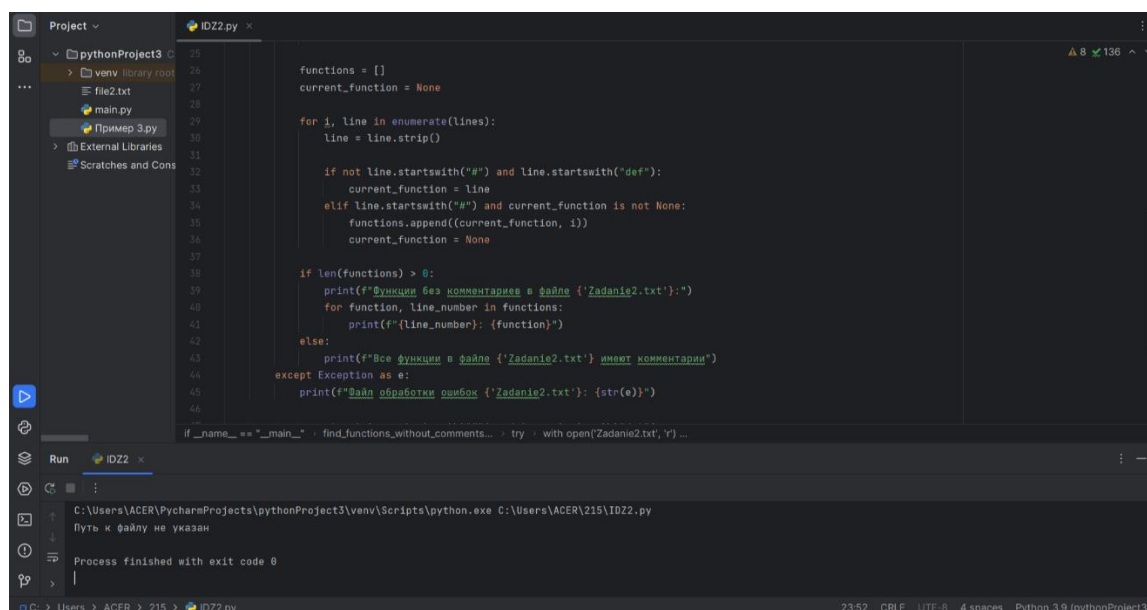


Рисунок 7. Результат индивидуального задания №2

Вывод: в ходе лабораторной работы приобрела навыков по работе с текстовыми файлами при написании программ с помощью языка программирования Python версии 3.x, изучила основные методы модуля os для работы с файловой системой, получение аргументов командной строки.

Контрольные вопросы

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

file object = open(<file-name>, r)

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

file object = open(<file-name>, w)

3. Как прочитать данные из файла в языке Python?

```
fileobj.read(<count>)
```

4. Как записать данные в файл в языке Python?

```
fileptr = open("file2.txt", "w")
```

5. Как закрыть файл в языке Python?

```
fileobject.close()
```

6. Изучите самостоятельно работу конструкции `with ... as`. Каково ее назначение в языке Python? Где она может быть использована еще, помимо работы с файлами?

Оператор `with` полезен в случае манипулирования файлами. 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?

`r` - только для чтения.

`w` - только для записи. Создаст новый файл, если не найдет с указанным именем.

`rb` - только для чтения (бинарный).

`wb` - только для записи (бинарный). Создаст новый файл, если не найдет с указанным именем.

`r+` - для чтения и записи.

`rb+` - для чтения и записи (бинарный).

`w+` - для чтения и записи. Создаст новый файл для записи, если не найдет с указанным именем.

`wb+` - для чтения и записи (бинарный). Создаст новый файл для записи, если не найдет с указанным именем.

`a` - откроет для добавления нового содержимого. Создаст новый файл для записи, если не найдет с указанным именем.

`a+` - откроет для добавления нового содержимого. Создаст новый файл для чтения записи, если не найдет с указанным именем.

`ab` - откроет для добавления нового содержимого (бинарный). Создаст новый файл для записи, если не найдет с указанным именем.

`ab+` - откроет для добавления нового содержимого (бинарный). Создаст новый файл для чтения записи, если не найдет с указанным именем.

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

Создание директорий:

```
import os  
os.mkdir(r"D:\folder")
```

Удаление файлов и директорий:

```
import os  
os.remove(r"D:\test.txt")
```

```
import os  
os.rmdir(r"D:\folder")
```

Запуск на исполнение:

```
import os  
os.startfile(r"D:\test.txt")
```

Получение имени файла и директории:

```
import os
```

```
print(os.path.basename("D:/test.txt"))  
test.txt
```

```
import os  
print(os.path.dirname("D:/folder/test.txt"))  
D:/folder
```

Вычисление размера:

```
import os  
print(os.path.getsize("D:\\test.txt"))
```

Переименование:

```
import os  
os.rename(r"D:\folder", r"D:\catalog")
```