

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

**Кафедра инфокоммуникаций  
Отчет по лабораторной работе № 2.15  
«Работа с файлами в языке Python»  
по дисциплине «Основы программной инженерии»**

Выполнил студент группы

ПИЖ-б-о-21-1

Зиберов Александр

«    » февраля 2023 г.

Подпись студента \_\_\_\_\_

Работа защищена

«    » \_\_\_\_\_ 20\_\_ г.

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

Ставрополь, 2023

## Цель работы:

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.

## Выполнение работы:

Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT, рисунок 1.

Ссылка: <https://github.com/afk552/lab2.15>

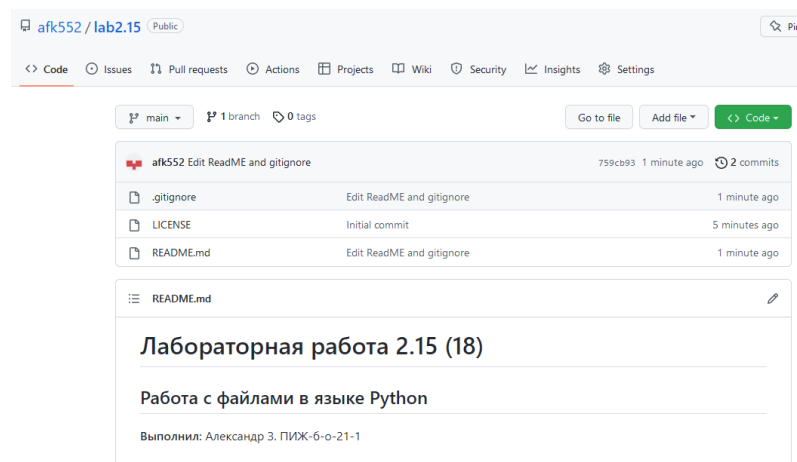


Рисунок 1 – Удаленный репозиторий на GitHub

Дополните файл .gitignore необходимыми правилами для работы с IDE PyCharm, рисунок 2.

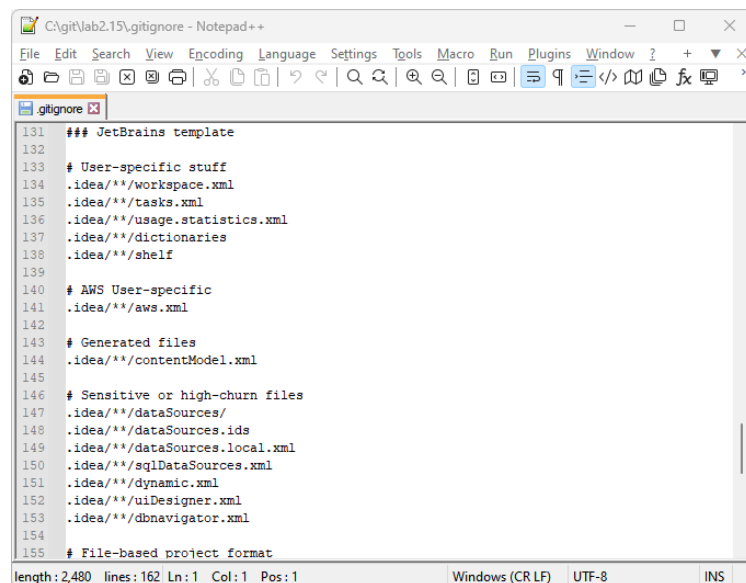
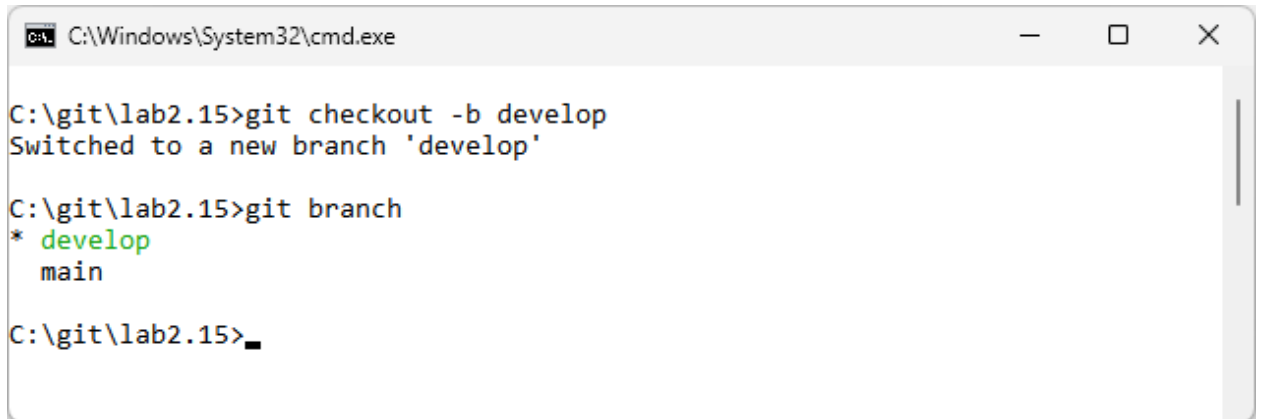


Рисунок 2 – Окно блокнота

Организируйте свой репозиторий в соответствии с моделью ветвления git-flow, рисунок 3.



```
C:\Windows\System32\cmd.exe

C:\git\lab2.15>git checkout -b develop
Switched to a new branch 'develop'

C:\git\lab2.15>git branch
* develop
  main

C:\git\lab2.15>
```

Рисунок 3 – Окно командной строки

Создайте проект PyCharm в папке репозитория, рисунок 4.

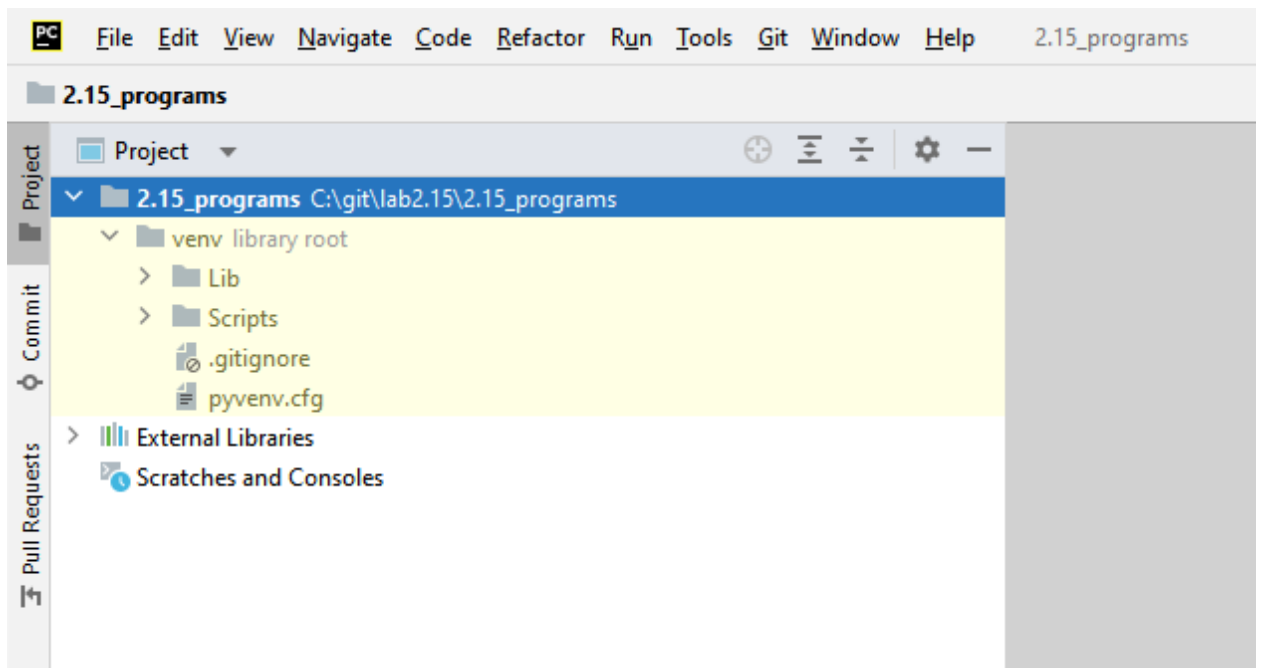


Рисунок 4 – Окно проекта в PyCharm

Самостоятельно подберите или придумайте задачу для работы с изученными функциями модуля `os`. Приведите решение этой задачи.

**Задача:** Разложить аудиофайлы по папкам с именами исполнителей, указанных в имени файла перед знаком `' - '`. Типы звуковых файлов: `mp3`, `wav`, `flac`, `ogg`.

```
audio_files_sorter.py x
2  # -*- coding: utf-8 -*-
3
4  import os
5  import shutil
6
7  # Разложить аудиофайлы по папкам с именами исполнителей,
8  # указанных в имени файла перед знаком ' - '.
9  # Типы звуковых файлов: mp3, wav, flac, ogg.
10
11  if __name__ == "__main__":
12      path = os.getcwd()
13      audio_ext = ["mp3", "wav", "flac", "ogg"]
14      for filename in os.listdir(path):
15          file_ext = filename.split('.')[-1]
16          if file_ext in audio_ext:
17              artist = filename.rpartition(' - ')[0]
18              os.mkdir(artist)
19              # Скопировать/переместить файл через os не получилось
20              shutil.copy(f'{filename}', f'{artist}/{filename}')
21              os.remove(f"{filename}")
22
```

Рисунок 5 – Код программы самостоятельного задания

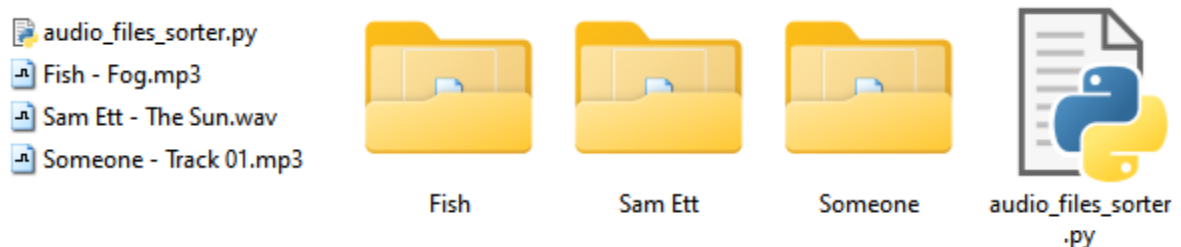
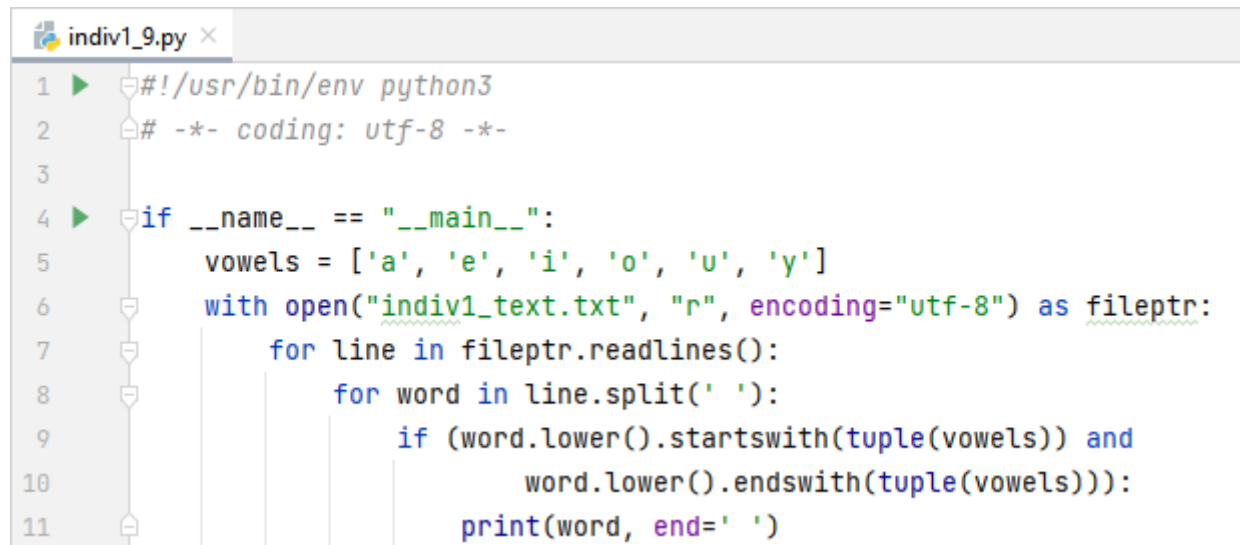


Рисунок 6 – Результат работы программы самостоятельного задания

## Индивидуальное задание.

### Вариант 9

**Задание 1.** Написать программу, которая считывает английский текст из файла и выводит на экран слова текста, начинающиеся и оканчивающиеся на гласные. буквы.



```
1  > #!/usr/bin/env python3
2  > # -*- coding: utf-8 -*-
3
4  > if __name__ == "__main__":
5      vowels = ['a', 'e', 'i', 'o', 'u', 'y']
6      with open("indiv1_text.txt", "r", encoding="utf-8") as fileptr:
7          for line in fileptr.readlines():
8              for word in line.split(' '):
9                  if (word.lower().startswith(tuple(vowels)) and
10                     word.lower().endswith(tuple(vowels))):
11                      print(word, end=' ')
```

Рисунок 7 – Код программы индивидуального задания 1

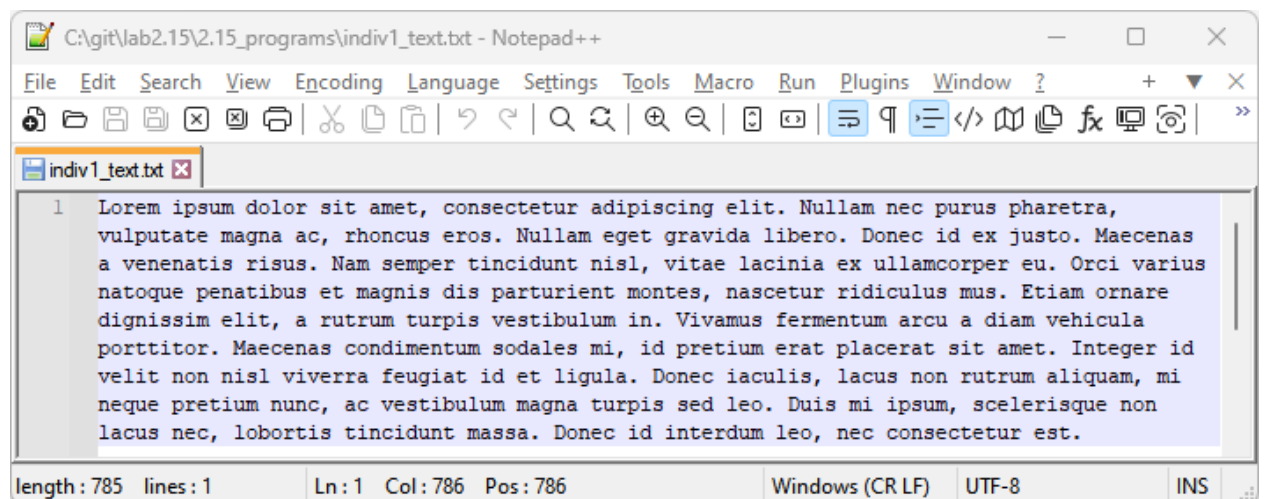


Рисунок 8 – Текстовый файл для программы индивидуального задания 1

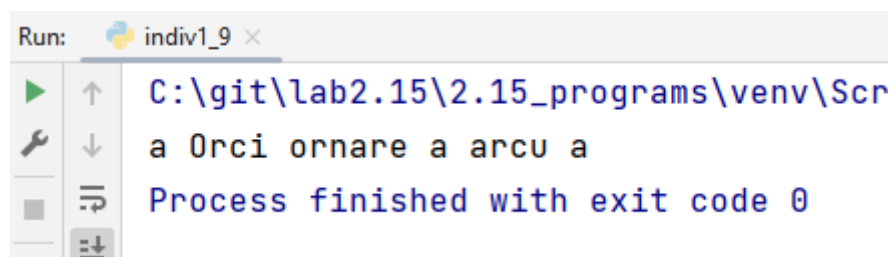
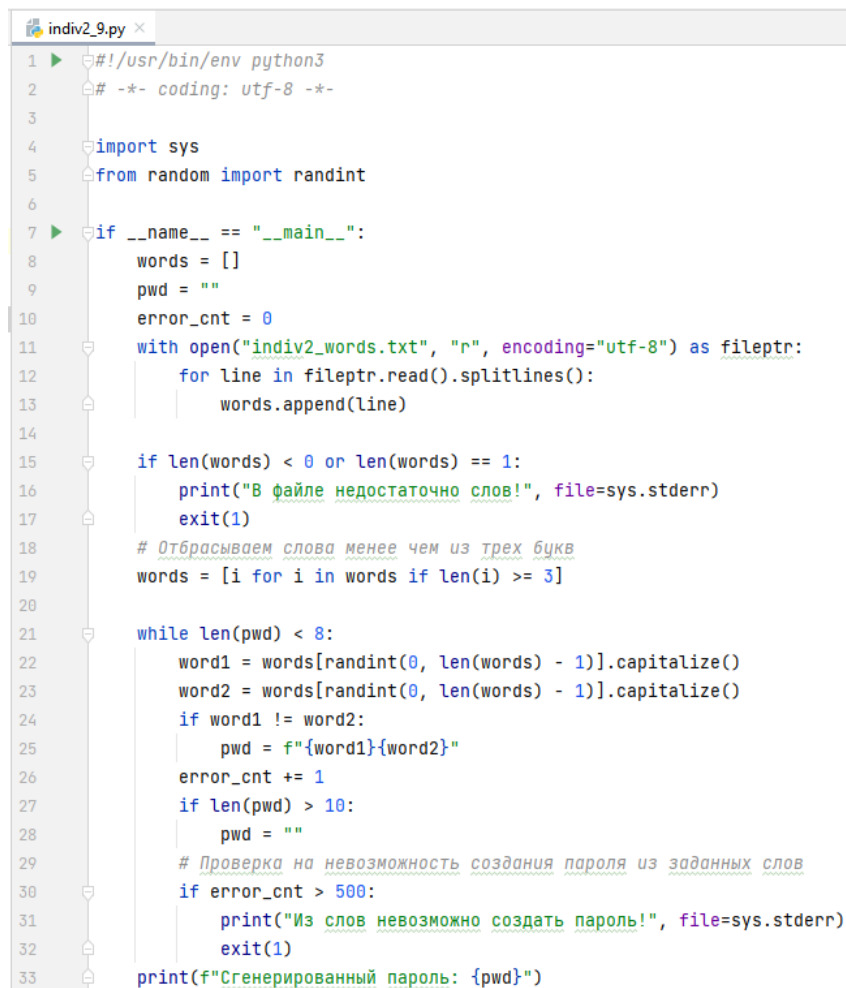


Рисунок 9 – Результат выполнения программы индивидуального задания 1

**Задание 2.** Создание пароля посредством генерирования случайных символов может обернуться сложностью в запоминании полученной относительно надежной последовательности. Некоторые системы создания паролей рекомендуют сцеплять вместе два слова на английском языке, тем самым упрощая запоминание заветного ряда символов – правда, в ущерб его надежности. Напишите программу, которая будет открывать файл со списком слов, случайным образом выбирать два из них и сцеплять вместе для получения итогового пароля. При создании пароля исходите из следующего требования: он должен состоять минимум из восьми символов и максимум из десяти, а каждое из используемых слов должно быть длиной хотя бы в три буквы. Кроме того, сделайте заглавными первые буквы обоих слов, чтобы легко можно было понять, где заканчивается одно и начинается другое. По завершении процесса полученный пароль должен быть отображен на экране.



```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sys
5  from random import randint
6
7  if __name__ == "__main__":
8      words = []
9      pwd = ""
10     error_cnt = 0
11     with open("indiv2_words.txt", "r", encoding="utf-8") as fileptr:
12         for line in fileptr.read().splitlines():
13             words.append(line)
14
15     if len(words) < 0 or len(words) == 1:
16         print("В файле недостаточно слов!", file=sys.stderr)
17         exit(1)
18     # Отбрасываем слова менее чем из трех букв
19     words = [i for i in words if len(i) >= 3]
20
21     while len(pwd) < 8:
22         word1 = words[randint(0, len(words) - 1)].capitalize()
23         word2 = words[randint(0, len(words) - 1)].capitalize()
24         if word1 != word2:
25             pwd = f"{word1}{word2}"
26             error_cnt += 1
27             if len(pwd) > 10:
28                 pwd = ""
29             # Проверка на невозможность создания пароля из заданных слов
30             if error_cnt > 500:
31                 print("Из слов невозможно создать пароль!", file=sys.stderr)
32                 exit(1)
33     print(f"Сгенерированный пароль: {pwd}")
```

Рисунок 10 – Код программы индивидуального задания 2

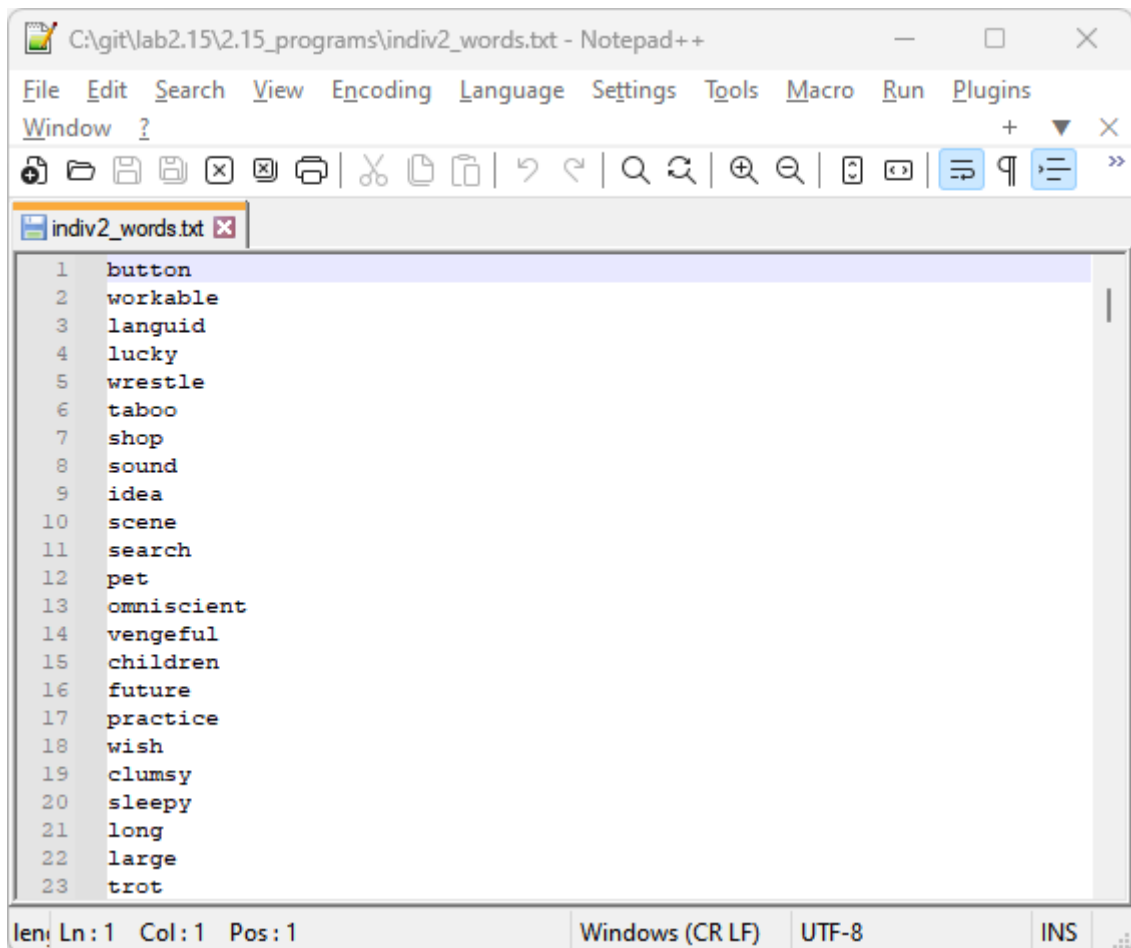


Рисунок 11 – Текстовый файл для программы индивидуального задания 2

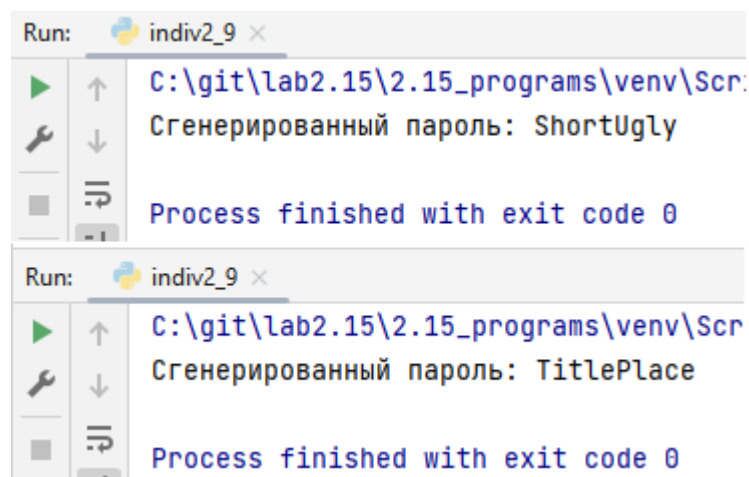


Рисунок 12 – Результат выполнения программы индивидуального задания 2

**Вывод:** В результате выполнения работы была изучена работа с файлами в языке Python, а также некоторые базовые команды для управления ими через модуль os.

## **Контрольные вопросы:**

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

```
fileptr = open("file.txt", "r")
```

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

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

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

```
with open("file.txt", 'r') as f:
```

```
    content = f.read();
```

```
    print(content)
```

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

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

```
fileptr.write(
```

```
    "Python is the modern day language. It makes things so simple.\n"
```

```
    "It is the fastest-growing programming language"
```

```
)
```

### **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?**

Начиная с версии 2.6, язык Python поддерживает протокол менеджеров контекста. Этот протокол гарантирует выполнение завершающих действий (например, закрытие файла) вне зависимости от того, произошло исключение внутри блока кода или нет. Необходимо заметить, что в Python 2.5 также можно использовать протокол, предварительно указав выражения (в Python 2.6 и выше это выражение указывать не нужно).

```
with <Выражение>[ as <Переменная>]:
```

```
<Блок, в котором перехватываем исключения>
```



**7. Изучите самостоятельно документацию Python по работе с файлами. Какие помимо рассмотренных существуют методы записи/чтения информации из файла?**

```
file = open(text.txt')
reader = open('text.txt')
try:
    # Дальнейшая обработка файлов происходит здесь
finally:
    reader.close()
```

Также с помощью Python можно читать данные из других типов файлов, например, csv.

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

os.name - имя операционной системы. Доступные варианты: 'posix', 'nt', 'mac', 'os2', 'ce', 'java'.

os.environ - словарь переменных окружения. Изменяемый (можно добавлять и удалять переменные окружения).

os.getlogin() - имя пользователя, вошедшего в терминал (Unix).

os.getpid() - текущий id процесса.

os.chmod(path, mode, \*, dir\_fd=None, follow\_symlinks=True) - смена прав доступа к объекту (mode - восьмеричное число).

os.chown(path, uid, gid, \*, dir\_fd=None, follow\_symlinks=True) - меняет id владельца и группы (Unix).

os.getcwd() - текущая рабочая директория.

os.link(src, dst, \*, src\_dir\_fd=None, dst\_dir\_fd=None, follow\_symlinks=True) - создаёт жёсткую ссылку.

os.listdir(path=".") - список файлов и директорий в папке.

os.system(command) - исполняет системную команду, возвращает код её завершения (в случае успеха 0).