

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 №2.15**  
**дисциплины «Основы кроссплатформенного программирования»**

Выполнил:  
Евдаков Евгений Владимирович  
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., 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:**

**Задание 1.** Создал общедоступный репозиторий на GitHub, в котором использована лицензия MIT и язык программирования Python, также добавил файл .gitignore с необходимыми правилами. Клонировал свой репозиторий на свой компьютер.

### **Рисунок 1. Клонирование репозитория**

**Задание 2.** Организовал свой репозиторий в соответствии с моделью ветвления git-flow, появилась новая ветка develop в которой буду выполнять дальнейшие задачи.

### **Рисунок 2. Модель ветвления git-flow**

**Задание 3.** Создал проект PyCharm в папке репозитория. Приступил к работе с примером. Добавил новый файл primer1.py.

**Условие примера:** файл file2.txt не существует, необходимо создать программу которая будет создавать новый файл и записывать его содержимое с помощью функции write() .

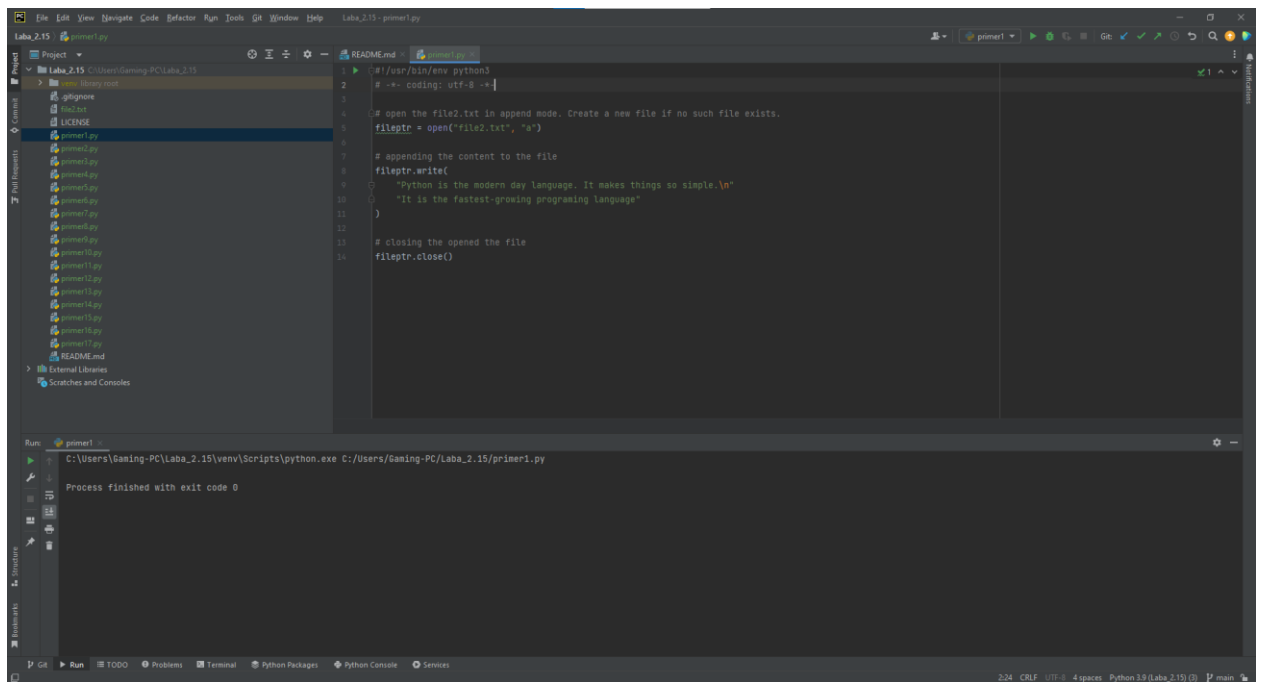



Рисунок 3. Реализация примера 1

 file2 – Блокнот

Файл Правка Формат Вид Справка

Python is the modern day language. It makes things so simple.  
It is the fastest-growing programming language

Рисунок 4. Результат примера 1

Создал новый файл под названием primer2.py.

**Условие примера:** необходимо открыть файл в режиме и добавить содержимое в существующий файл file2.txt.

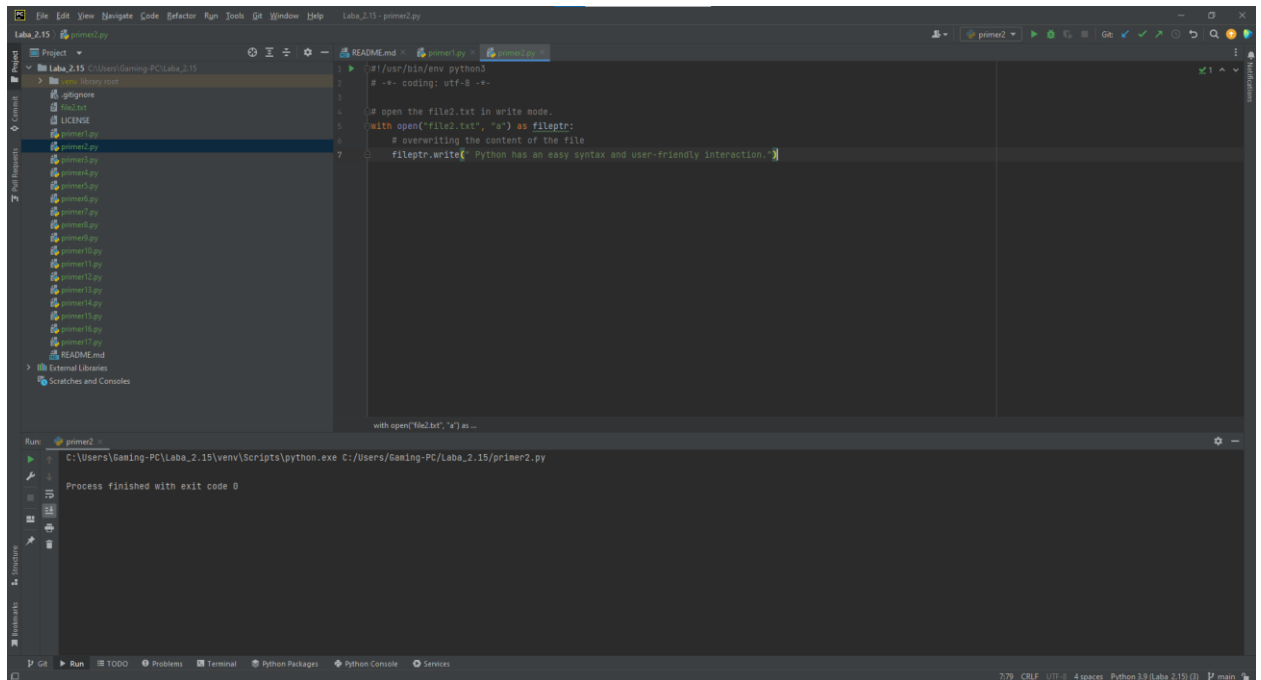


Рисунок 5. Реализация второго примера

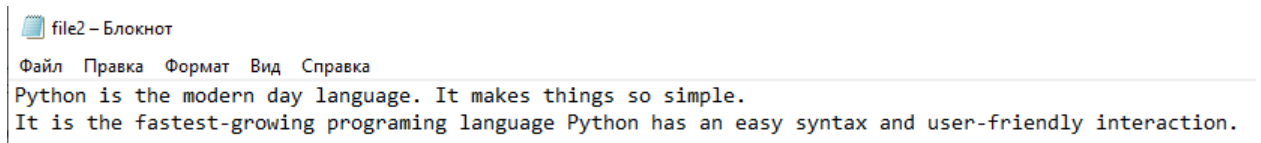


Рисунок 6. Результат второго примера

Создал новый файл под названием primer3.py

**Условие примера:** чтение строк с помощью метода readline()

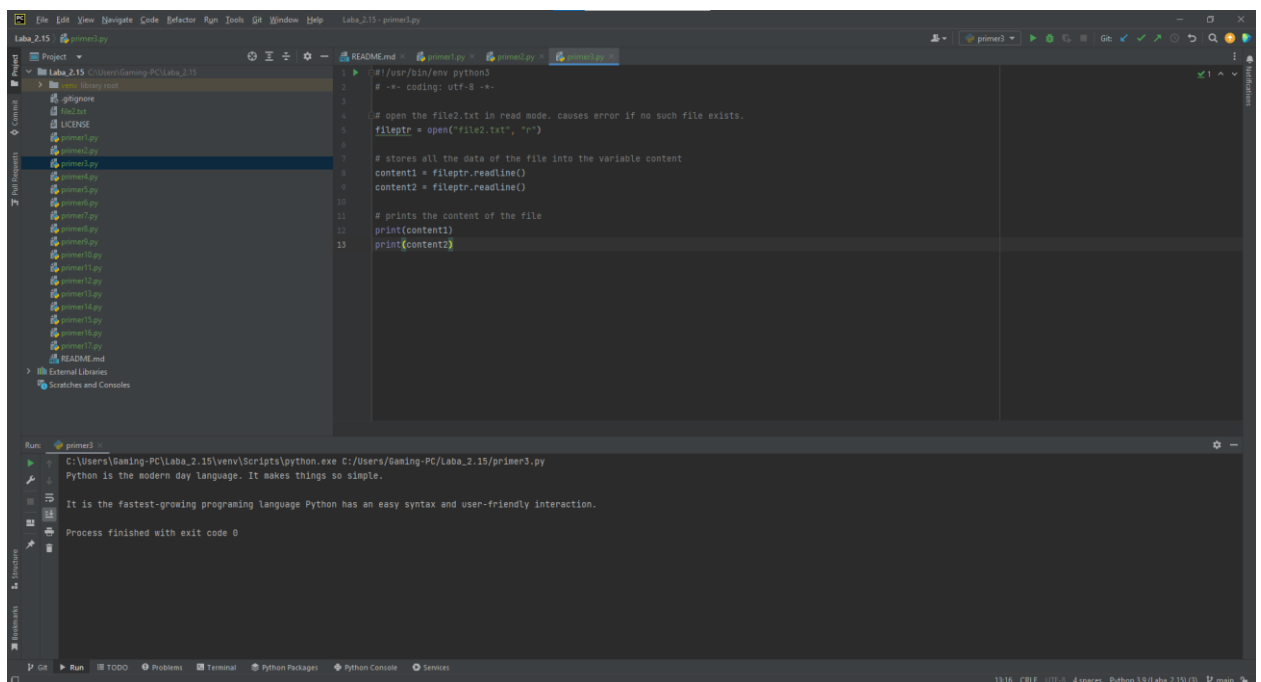


Рисунок 7. Реализация третьего примера

Создал новый файл под названием primer4.py

**Условие примера:** чтение строк с помощью функции `readlines()`

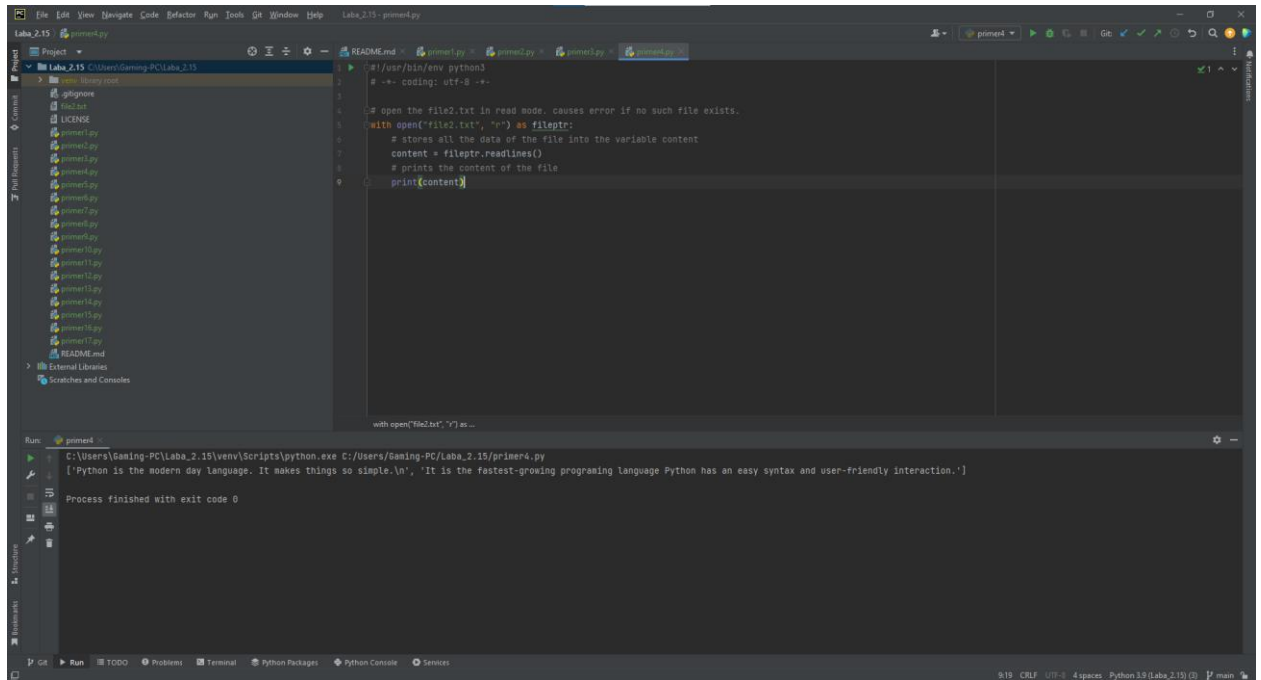


Рисунок 8. Реализация четвертого примера

Создал новый файл под названием `primer5.py`

**Условие примера:**

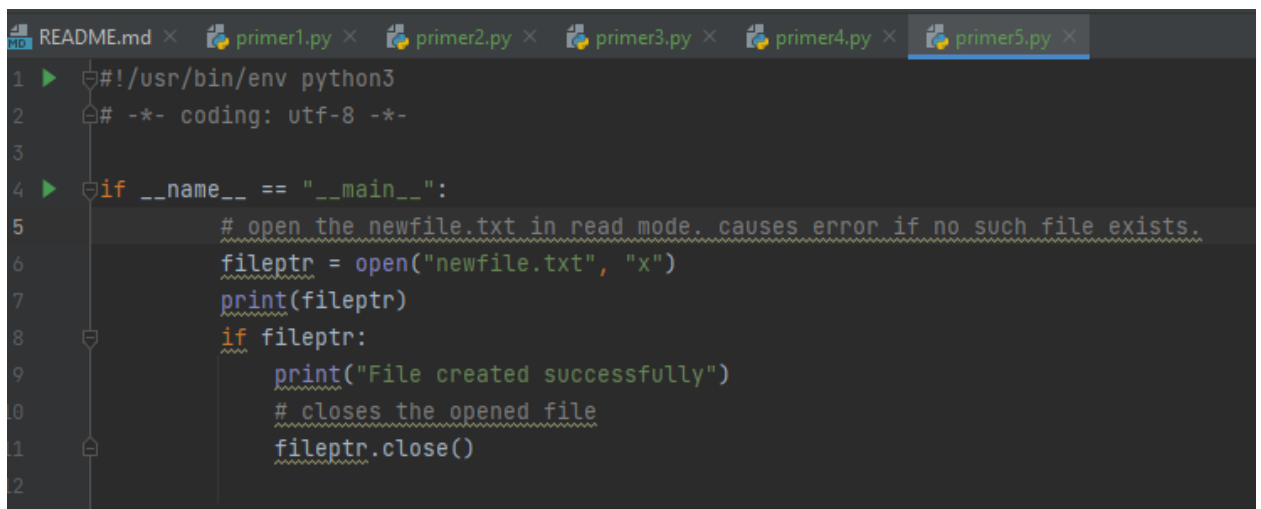
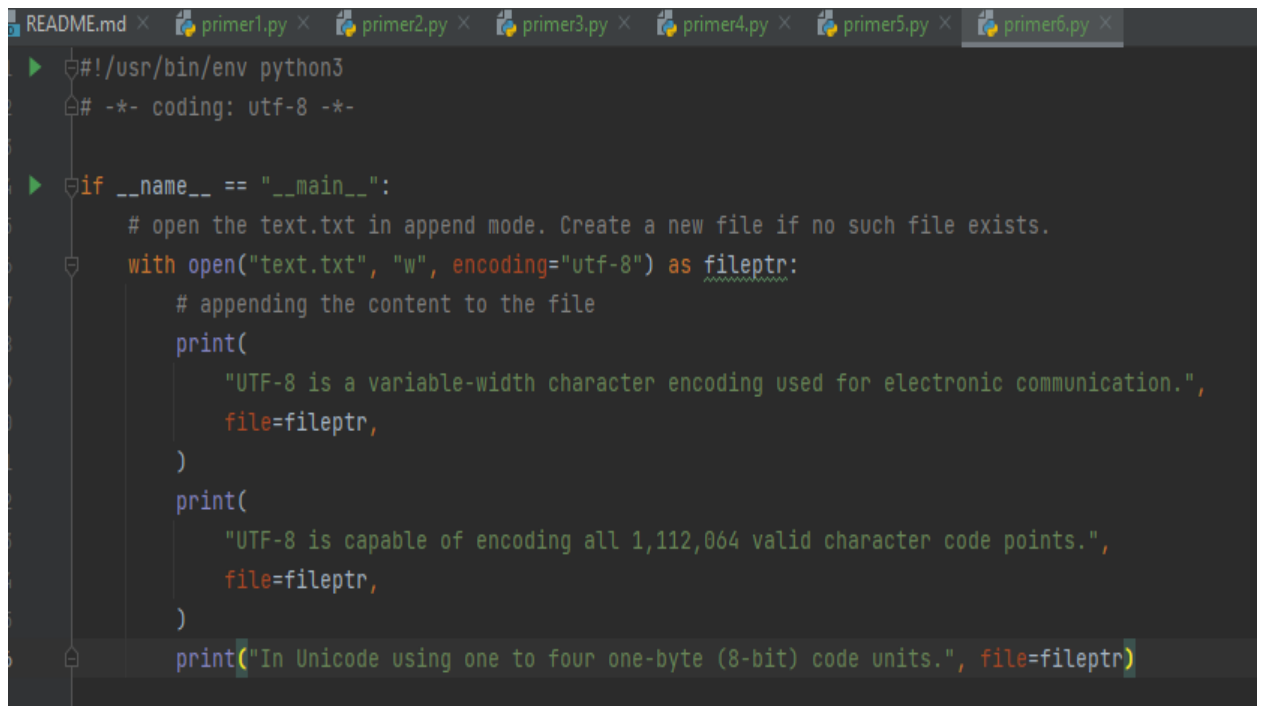


Рисунок 9. Реализация пятого примера


Создал новый файл под названием `primer6.py`

**Условие примера:**



```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 if __name__ == "__main__":
5     # open the text.txt in append mode. Create a new file if no such file exists.
6     with open("text.txt", "a", encoding="utf-8") as fileptr:
7         # appending the content to the file
8         print(
9             "UTF-8 is a variable-width character encoding used for electronic communication.",
10             file=fileptr,
11         )
12         print(
13             "UTF-8 is capable of encoding all 1,112,064 valid character code points.",
14             file=fileptr,
15         )
16         print("In Unicode using one to four one-byte (8-bit) code units.", file=fileptr)
```

Рисунок 10. Реализация шестого примера

 text – Блокнот

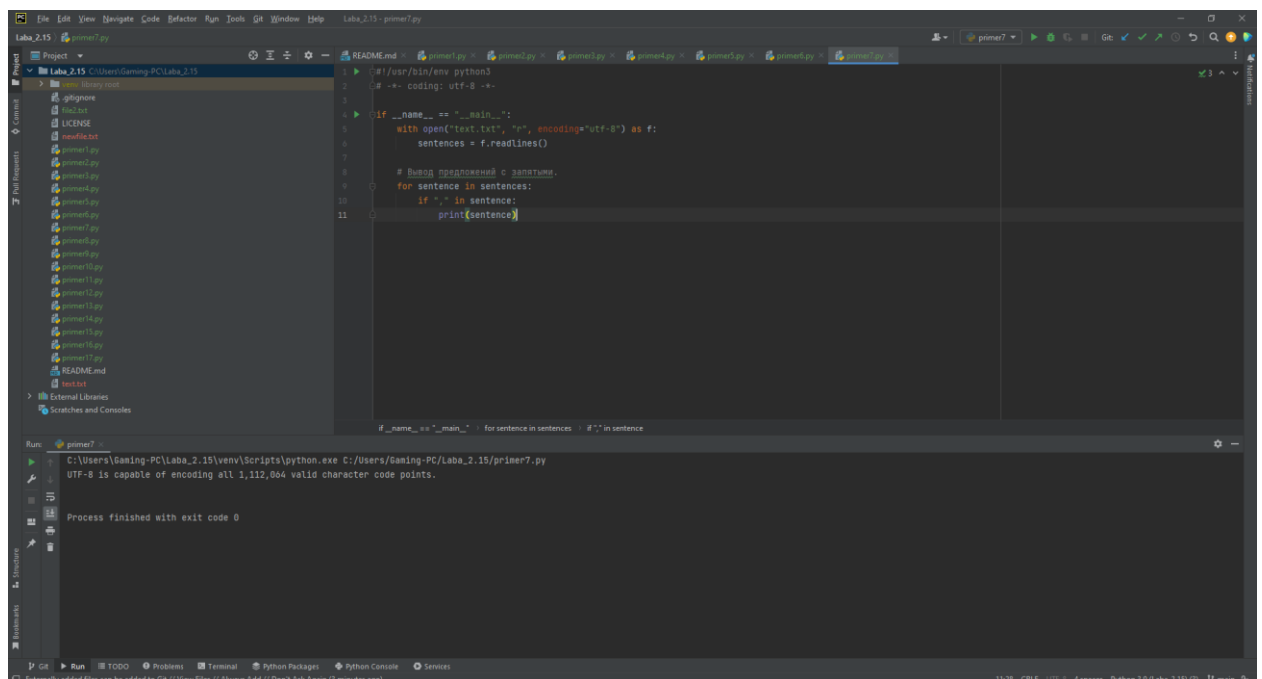
Файл Правка Формат Вид Справка

UTF-8 is a variable-width character encoding used for electronic communication.  
UTF-8 is capable of encoding all 1,112,064 valid character code points.  
In Unicode using one to four one-byte (8-bit) code units.

Рисунок 11. Результат шестого примера

Создал новый файл под названием primer7.py

**Условие примера:**



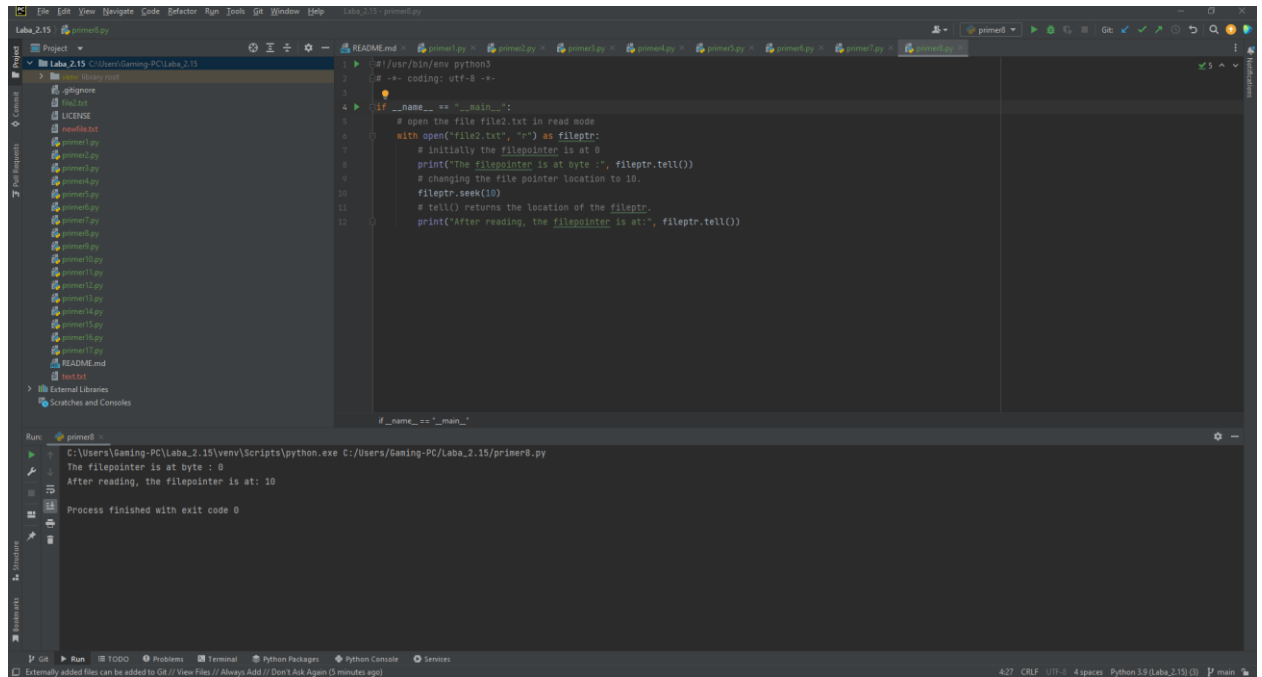
```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 if __name__ == "__main__":
5     with open("text.txt", "r", encoding="utf-8") as f:
6         sentences = f.readlines()
7
8     # Вывод предложений с запятой.
9     for sentence in sentences:
10         if "," in sentence:
11             print(sentence)
```

Run: primer7.py  
C:\Users\Gaming-PC\Lab2.15\venv\scripts\python.exe C:\Users\Gaming-PC\Lab2.15\primer7.py  
UTF-8 is capable of encoding all 1,112,064 valid character code points.  
Process finished with exit code 0

## Рисунок 12. Реализация седьмого примера

Создал новый файл под названием primer8.py

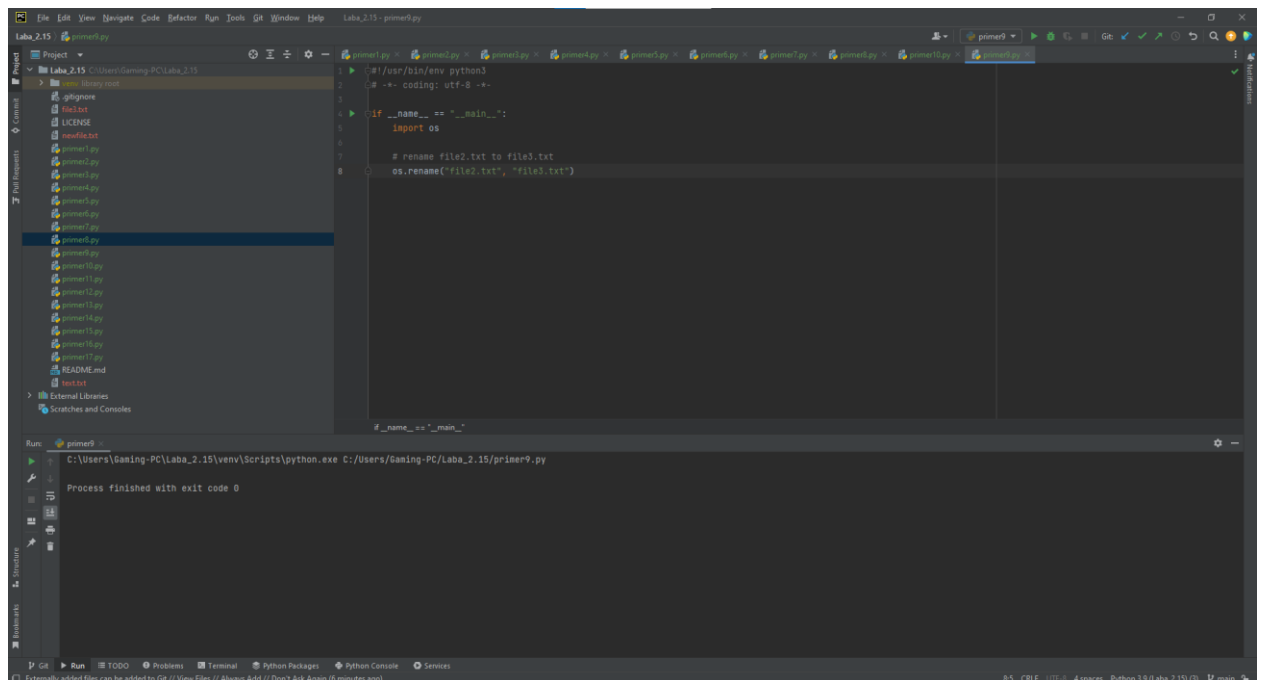
Условие примера:



## Рисунок 13. Реализация восьмого примера

Создал новый файл под названием primer9.py

Условие примера:



## Рисунок 14. Реализация девятого примера

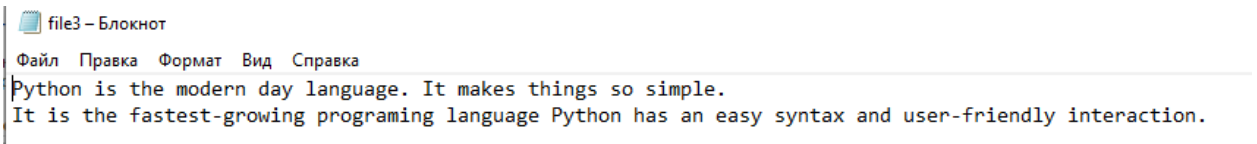


Рисунок 15. Результат девятого примера

Создал новый файл под названием primer10.py

Условие примера:

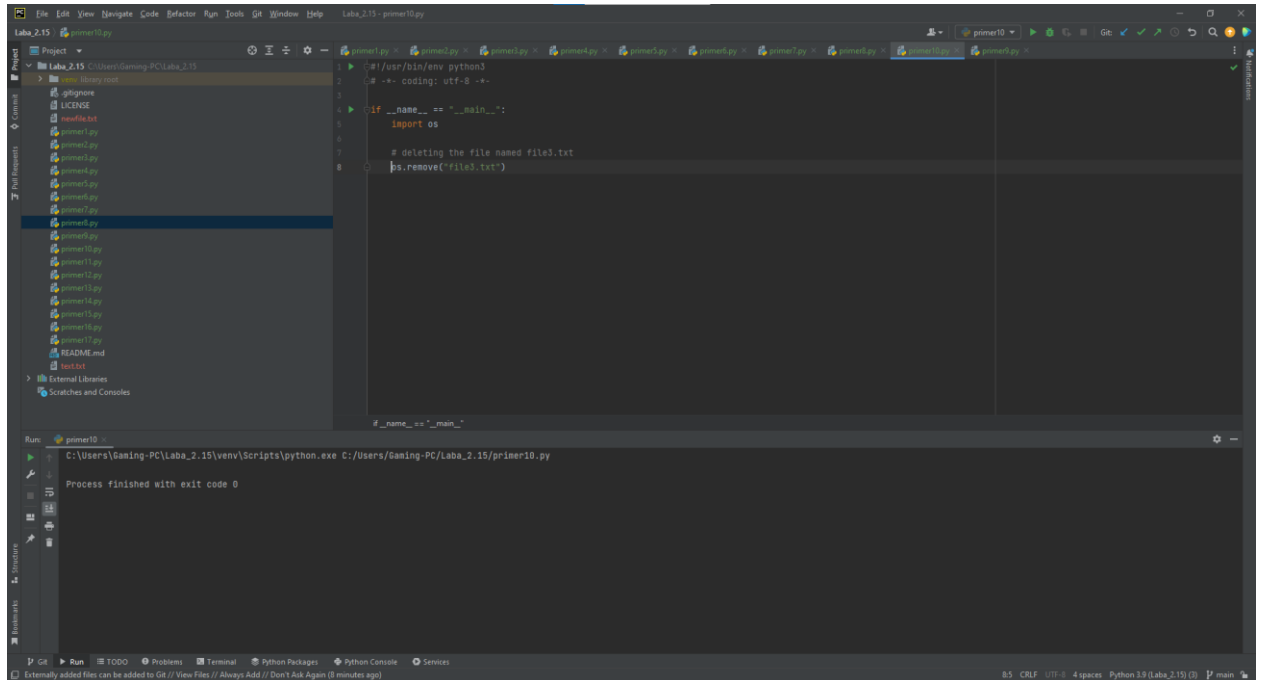


Рисунок 16. Реализация десятого примера

Создал новый файл под названием primer11.py

Условие примера:

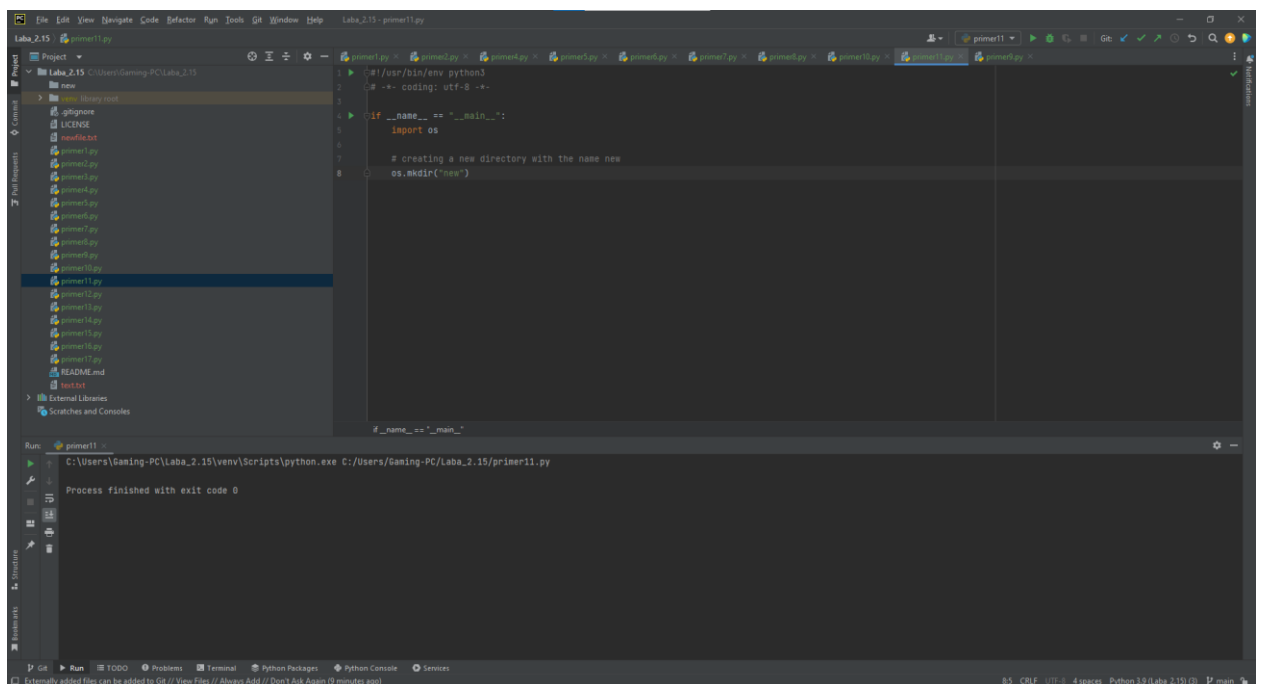




Рисунок 17. Реализация одиннадцатого примера

Имя	Дата изменения	Тип	Размер
.idea	08.10.2023 23:31	Папка с файлами	
new	08.10.2023 23:31	Папка с файлами	
venv	08.10.2023 23:16	Папка с файлами	
.gitignore	08.10.2023 23:06	Текстовый докум...	4 КБ
LICENSE	08.10.2023 23:06	Файл	2 КБ
newfile	08.10.2023 23:22	Текстовый докум...	0 КБ
primer1	08.10.2023 23:14	JetBrains PyChar...	1 КБ
primer2	08.10.2023 23:17	JetBrains PyChar...	1 КБ
primer3	08.10.2023 23:18	JetBrains PyChar...	1 КБ
primer4	08.10.2023 23:20	JetBrains PyChar...	1 КБ
primer5	08.10.2023 23:22	JetBrains PyChar...	1 КБ
primer6	08.10.2023 23:25	JetBrains PyChar...	1 КБ
primer7	08.10.2023 23:26	JetBrains PyChar...	1 КБ
primer8	08.10.2023 23:27	JetBrains PyChar...	1 КБ
primer9	08.10.2023 23:28	JetBrains PyChar...	1 КБ
primer10	08.10.2023 23:30	JetBrains PyChar...	1 КБ
primer11	08.10.2023 23:31	JetBrains PyChar...	1 КБ
primer12	08.10.2023 23:12	JetBrains PyChar...	0 КБ
primer13	08.10.2023 23:12	JetBrains PyChar...	0 КБ
primer14	08.10.2023 23:13	JetBrains PyChar...	0 КБ
primer15	08.10.2023 23:13	JetBrains PyChar...	0 КБ
primer16	08.10.2023 23:13	JetBrains PyChar...	0 КБ
primer17	08.10.2023 23:13	JetBrains PyChar...	0 КБ
README	08.10.2023 23:06	Исходный файл ...	1 КБ
text	08.10.2023 23:25	Текстовый докум...	1 КБ

Рисунок 18. Результат программы

Создал новый файл под названием primer12.py

Условие примера:

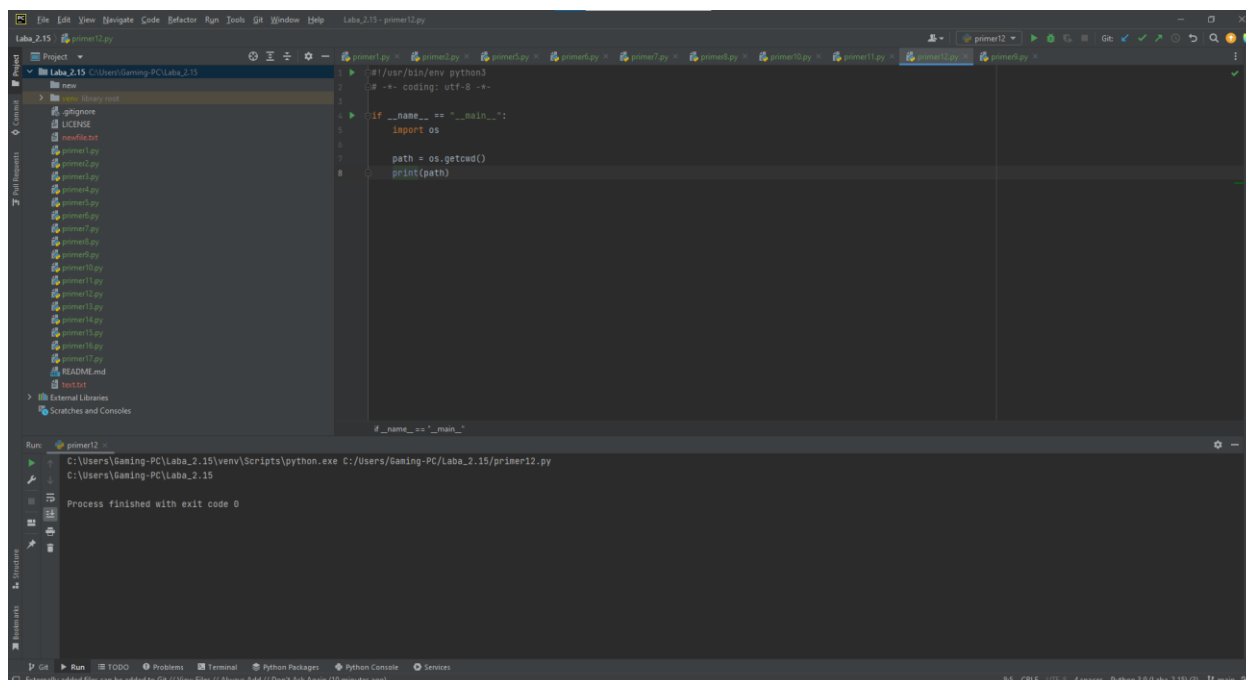


Рисунок 19. Реализация двенадцатого примера

Создал новый файл под названием primer13.py

Условие примера:

```

1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 ▶ if __name__ == "__main__":
5     import os
6     . . .
7     # Changing current directory with the new directory
8     os.chdir("C:\\Windows")
9     #It will display the current working directory
10    print(os.getcwd())

```

Рисунок 20. Реализация тринадцатого примера

Создал новый файл под названием primer14.py

**Условие примера:**

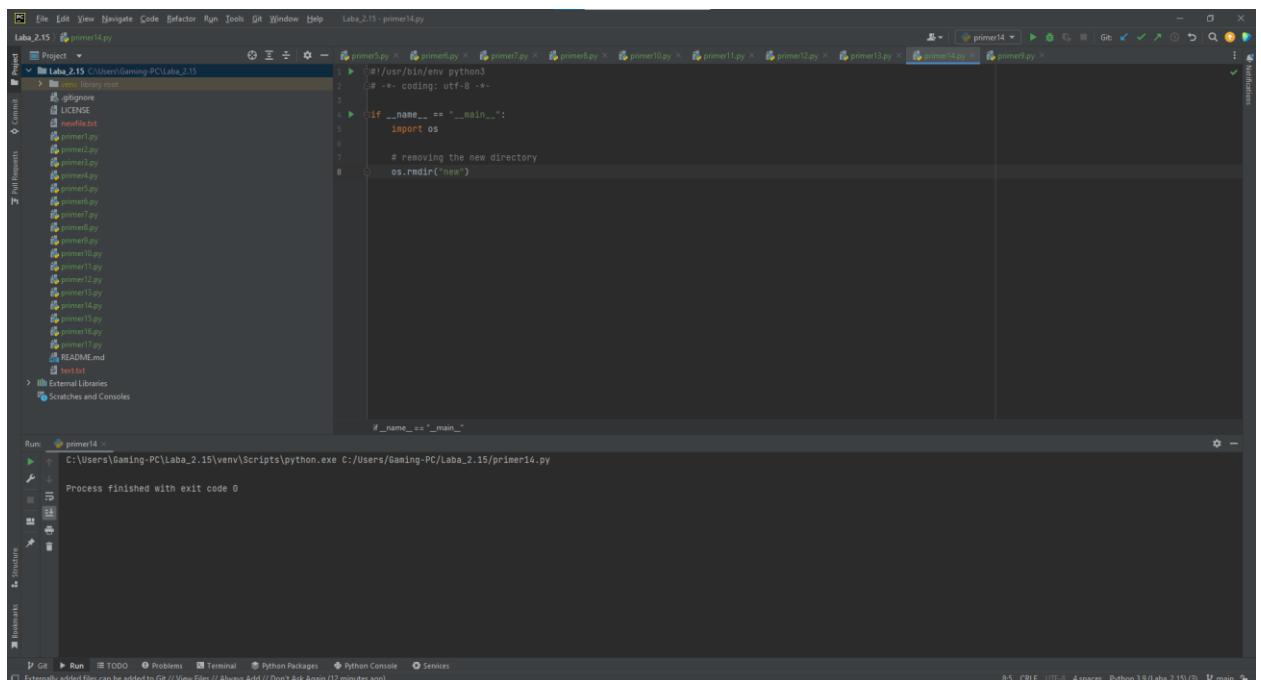


Рисунок 21. Реализация четырнадцатого примера

Создал новый файл под названием primer15.py

**Условие примера:**

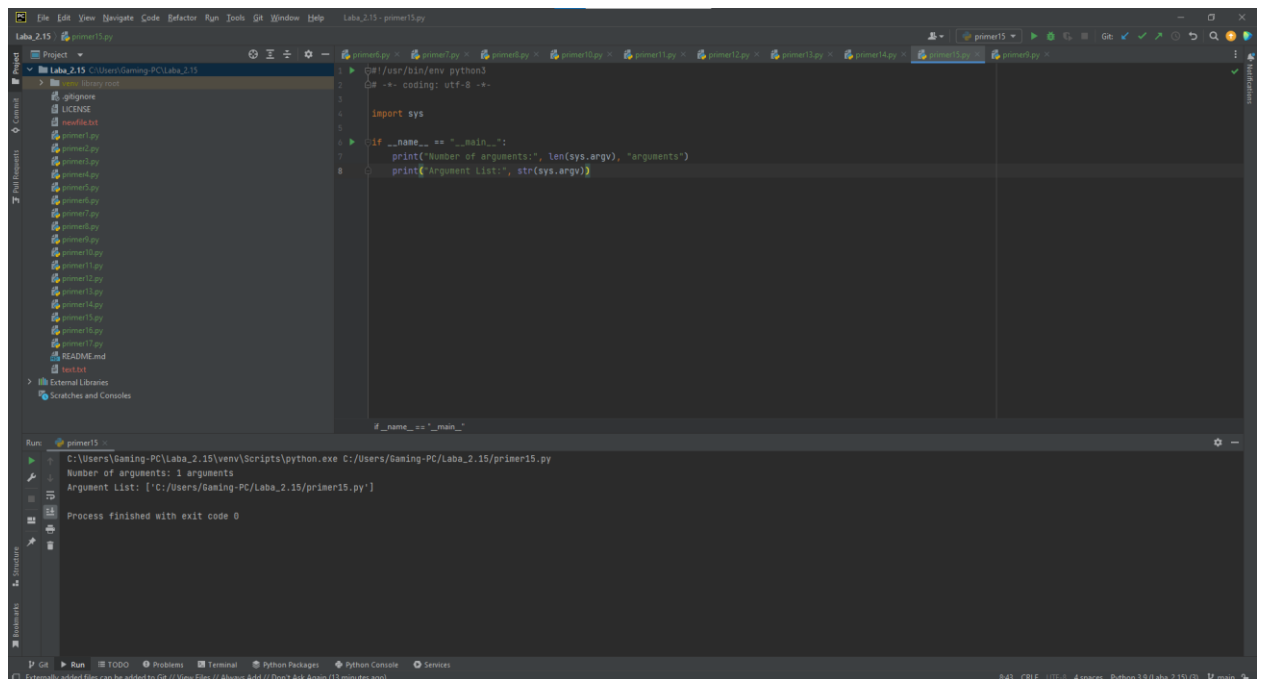


Рисунок 22. Реализация пятнадцатого примера

Создал новый файл под названием primer16.py

Условие примера:

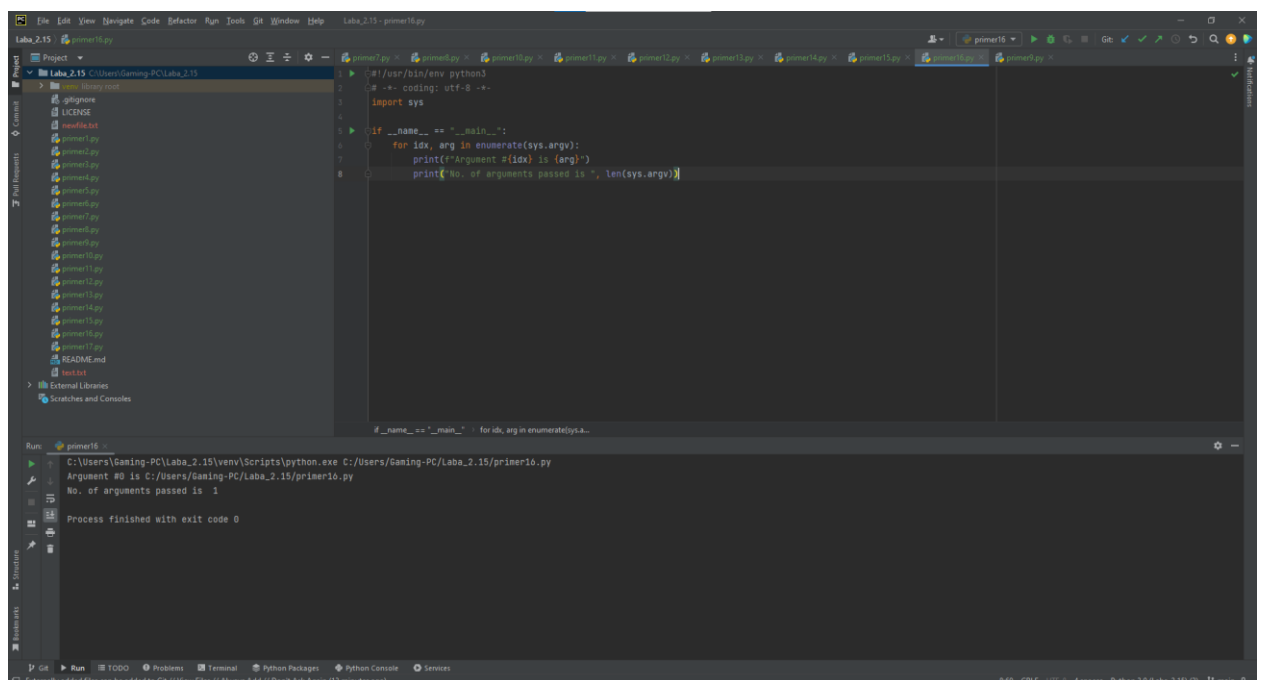


Рисунок 23. Реализация шестнадцатого примера

Создал новый файл под названием primer17.py

Условие примера:

```

#!/usr/bin/env python3
# -*- coding: utf-8 -*-
import os
import secrets
import string
import sys

if __name__ == "__main__":
    if len(sys.argv) != 2:
        print("The password length is not given!", file=sys.stderr)
        sys.exit(1)

    chars = string.ascii_letters + string.punctuation + string.digits
    length_pwd = int(sys.argv[1])
    result = []
    for _ in range(length_pwd):
        idx = secrets.SystemRandom().randrange(len(chars))
        result.append(chars[idx])

    print(f"Secret Password: {''.join(result)}")

```

Рисунок 24. Реализация семнадцатого примера

## Задание 6.

Создал новый файл под названием os\_idz.py

Условие примера: самостоятельно подобрать или придумать задачу для работы с изученными функциями модуля os. Привести решение этой задачи.

```

1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 import os
5
6 if __name__ == "__main__":
7     def list_txt_files():
8         current_directory = os.getcwd()
9         txt_files = [file for file in os.listdir(current_directory) if file.endswith('.txt')]
10
11         if txt_files:
12             print("Список файлов с расширением .txt:")
13             for file in txt_files:
14                 print(file)
15             print(f"Общее количество файлов: {len(txt_files)}")
16         else:
17             print("В текущей директории нет файлов с расширением .txt")
18
19     # Вывод функции
20     list_txt_files()
21
22 if __name__ == "__main__":

```

Run: C:\Users\Gaming-PC\Labo\_2.15\venv\Scripts\python.exe C:\Users\Gaming-PC\Labo\_2.15\os\_idz.py

Список файлов с расширением .txt:  
 id1.txt  
 id2.txt  
 newfile.txt  
 text.txt  
 Общее количество файлов: 4  
 Process finished with exit code 0

Рисунок 25. Реализация os программы

## Задание 7.

## Индивидуальное задание

## Вариант 10

Создал новый файл под названием idz1.py.

**Условие задания:** Написать программу, которая считывает текст из файла и выводит на экран только строки, не содержащие двузначных чисел.

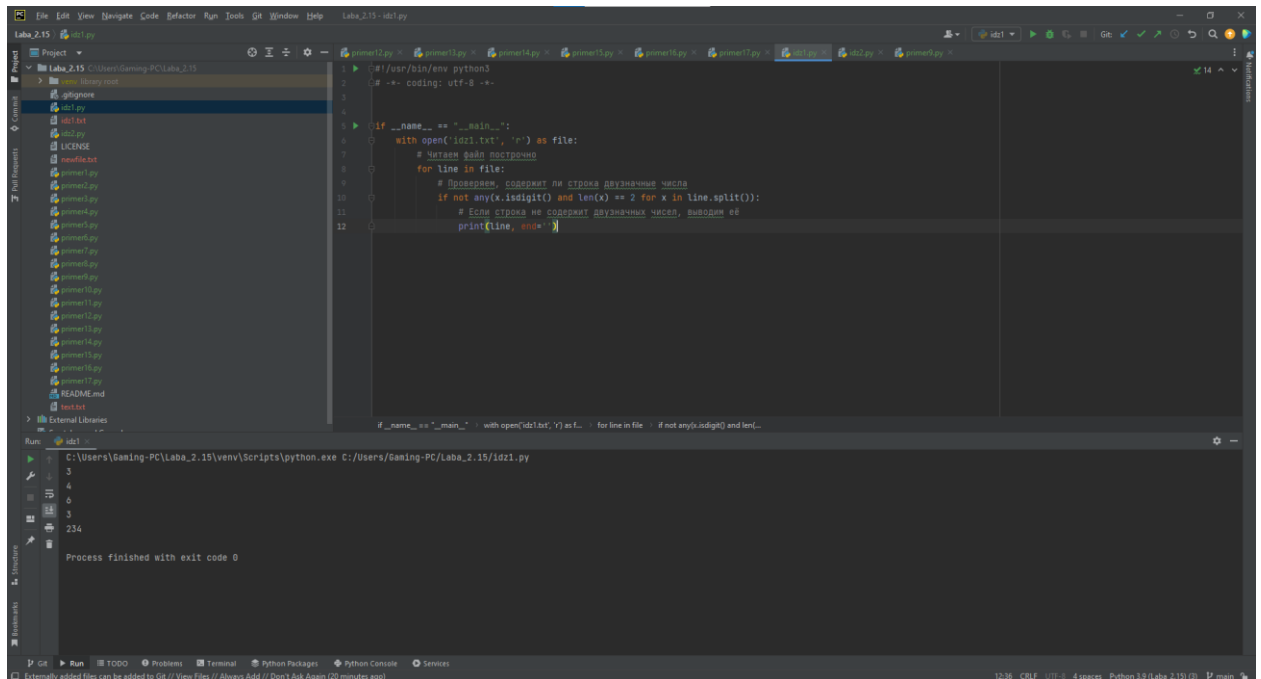


Рисунок 26. Реализация первого индивидуального задания

Рисунок 6. Программа индивидуального задания

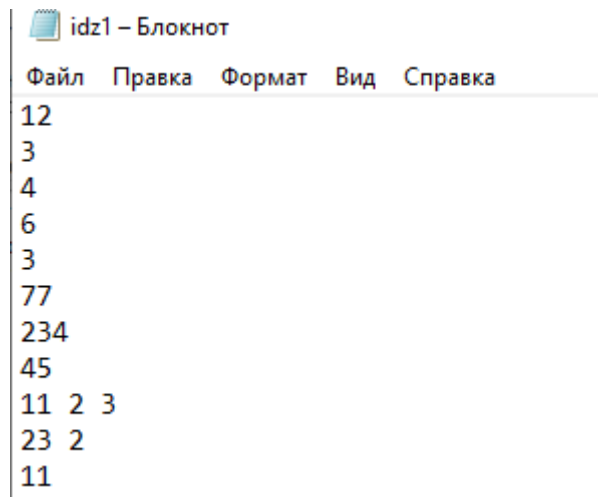


Рисунок 27. Результат программы

Создал новый файл под названием individual2.py.

**Условие задания:** Ученикам, желающим запомнить правила написания слов в английском языке, часто напоминают следующее рифмованное одностопное: «I before E except after C» (I перед E, если не после

С). Это правило позволяет запомнить, в какой последовательности писать буквы I и E, идущие в слове одна за другой, а именно: буква I должна предшествовать букве E, если непосредственно перед ними не стоит буква C. Если стоит – порядок гласных будет обратным. Примеры слов, на которые действует это правило: believe, chief, fierce, friend, ceiling и receipt. Но есть и исключения из этого правила, и одним из них является слово weird (странный). Напишите программу, которая будет построчно обрабатывать текстовый файл. В каждой строке может присутствовать много слов, а может и не быть ни одного. Слова, в которых буквы E и I не соседствуют друг с другом, обработке подвергать не следует. Если же такое соседство присутствует, необходимо проверить, соответствует ли написание анализируемого слова указанному выше правилу. Создайте и выведите на экран два списка. В первом должны располагаться слова, следующие правилу, а во втором – нарушающие его. При этом списки не должны содержать повторяющиеся слова. Также отобразите на экране длину каждого списка, чтобы пользователю было понятно, сколько слов в файле не отвечает правилу.

```

10 # Открываем файл для чтения
11 with open('id2.txt', 'r') as file:
12     # Читаем файл построчно
13     for line in file:
14         # Разбиваем строку на слова
15         words = line.split()
16
17         # Перебираем каждое слово
18         for word in words:
19             # Проверим, содержит ли слово буквы E и I, и они идут друг за другом
20             if 'ei' in word.lower() or 'ie' in word.lower():
21                 # Проверим, соответствует ли слово правилу
22                 if 'ci' not in word.lower() and 'ce' not in word.lower():
23                     words1.append(word)
24                 else:
25                     words0.append(word)
26
27 # Удаляем повторяющиеся слова из списков
28 words0 = list(set(words0))
29 words1 = list(set(words1))
30
31 # Выводим списки и их длину на экран
32 print('Слова, следующие правилу:')
33 print(words0)
34 print('Слова, нарушающие правило:')
35 print(words1)
36
37 if __name__ == '__main__': with open('id2.txt', 'r') as f:

```

Run

```

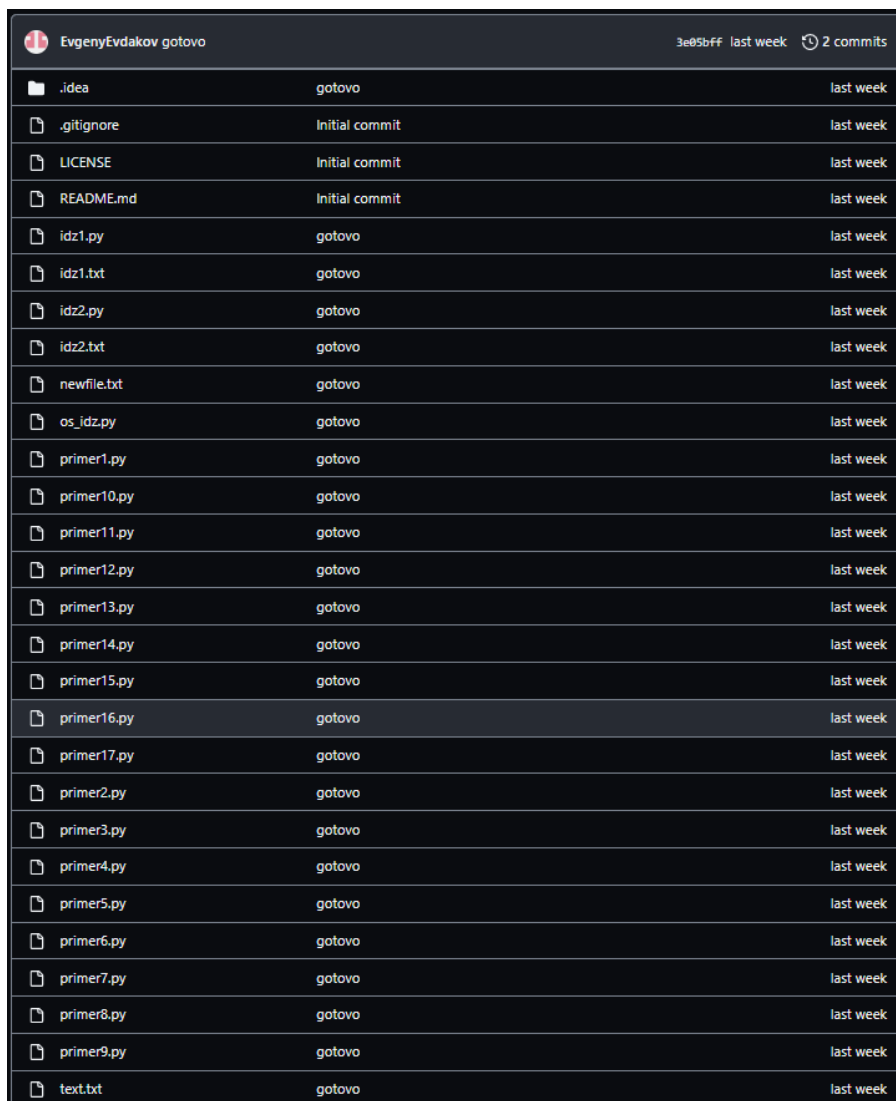
C:\Users\Gaming-PC\Labo_2.15\venv\Scripts\python.exe C:/Users/Gaming-PC/Labo_2.15/id2.py
Слова, следующие правилу:
[]
Длина списка: 0
Слова, нарушающие правило:
['field.', 'believe', 'being', 'achieve']
Длина списка: 4
Process finished with exit code 0

```

Рисунок 28. Реализация второго индивидуального задания

## Задание 8.

После выполнения работы на ветке develop, слил ее с веткой main и отправил изменения на удаленный сервер.



EvgenyEvdakov gotovo		3e05bfff last week 2 commits
.idea	gotovo	last week
.gitignore	Initial commit	last week
LICENSE	Initial commit	last week
README.md	Initial commit	last week
idz1.py	gotovo	last week
idz1.txt	gotovo	last week
idz2.py	gotovo	last week
idz2.txt	gotovo	last week
newfile.txt	gotovo	last week
os_idz.py	gotovo	last week
primer1.py	gotovo	last week
primer10.py	gotovo	last week
primer11.py	gotovo	last week
primer12.py	gotovo	last week
primer13.py	gotovo	last week
primer14.py	gotovo	last week
primer15.py	gotovo	last week
primer16.py	gotovo	last week
primer17.py	gotovo	last week
primer2.py	gotovo	last week
primer3.py	gotovo	last week
primer4.py	gotovo	last week
primer5.py	gotovo	last week
primer6.py	gotovo	last week
primer7.py	gotovo	last week
primer8.py	gotovo	last week
primer9.py	gotovo	last week
text.txt	gotovo	last week

Рисунок 29. Удаленный репозиторий

Ссылка: [https://github.com/EvgenyEvdakov/Laba\\_2.15](https://github.com/EvgenyEvdakov/Laba_2.15)

### Ответы на контрольные вопросы:

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

‘r’ - Открывает файл только для чтения.

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

‘w’ – Открывает файл только для записи.

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

Функция read() используется для чтения содержимого файла после открытия его в режиме чтения ( r ).

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

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

Метод файла `file.close()` закрывает открытый файл. Закрытый файл больше не может быть прочитан или записан. Любая операция, которая требует, чтобы файл был открыт, вызовет исключение `ValueError` после того, как файл был закрыт.

#### **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` в Python предназначена для управления контекстами ресурсов, таких как файлы, сетевые соединения, базы данных и другие, чтобы гарантировать их корректное открытие, использование и закрытие. Она гарантирует, что ресурсы будут правильно освобождены после завершения блока кода. Конструкция `with ... as` обычно используется с контекстными менеджерами, которые определяют методы `__enter__` и `__exit__` для выполнения действий до и после использования ресурсов.

Кроме работы с файлами, `with ... as` может быть использована для работы с сетевыми соединениями (например, с помощью библиотеки `socket`), управления транзакциями в базах данных (с библиотекой `sqlite3`), манипуляций с ресурсами, требующими блокировки, и многими другими ситуациями, где важна правильная инициализация и освобождение ресурсов.

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

Помимо рассмотренных методов (`read()`, `write()`, `readline()`, `writelines()` и т. д.) для чтения и записи информации в файлы, существует ряд других методов в Python для более специфических задач:

- `readlines()`: Этот метод читает все строки из файла и возвращает их в виде списка строк.



- `seek()`: Метод `seek()` используется для перемещения указателя файла в заданную позицию. Это полезно, например, чтобы перейти к определенному месту в файле.

- `tell()`: Метод `tell()` возвращает текущую позицию указателя файла.

- `flush()`: Метод `flush()` записывает непрошедшие данные на диск, но не закрывает файл.

- `truncate()`: Метод `truncate()` урезает файл до указанного размера.

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

Модуль `os` в Python предоставляет множество функций для работы с файловой системой. Помимо рассмотренных ранее функций, некоторые другие функции модуля `os` включают:

- `os.rename(src, dst)`: Используется для переименования файла или директории.

- `os.remove(path)`: Удаляет файл.

- `os.rmdir(path)`: Удаляет пустую директорию.

- `os.removedirs(path)`: Удаляет директории рекурсивно.

- `os.listdir(path)`: Возвращает список файлов и директорий в указанной директории.

- `os.makedirs(path)`: Создает директорию или директории, включая промежуточные.

- `os.path`: Этот модуль предоставляет функции для работы с путями к файлам и директориям.

Это лишь несколько примеров функций модуля `os`. Модуль `os` предоставляет множество других функций для работы с файловой системой и системными вызовами в операционной системе.

**Вывод:** приобрел навыки по работе с текстовыми файлами при написании программ с помощью языка программирования Python версии 3.x, изучил основные методы модуля `os` для работы с файловой системой, получил аргументы командной строки.