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:  Пинаева Диана Олеговна  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) | |
|  | | Проверил:  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.Создала общедоступный репозиторий на GitHub, в котором использована лицензия MIT и язык программирования Python, также добавила файл .gitignore с необходимыми правилами. Клонировала свой репозиторий на свой компьютер.

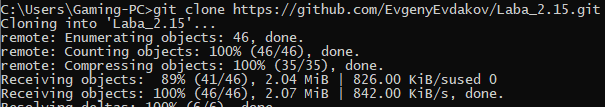


Рисунок 1. Клонирование репозитория

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

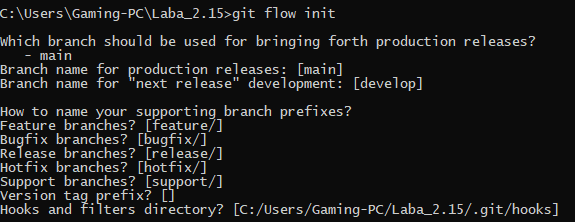


Рисунок 2. Модель ветвления git-flow

Задание 3.Создала виртуальное окружение conda и активировала его, также установила необходимые пакеты isort, black, flake8.

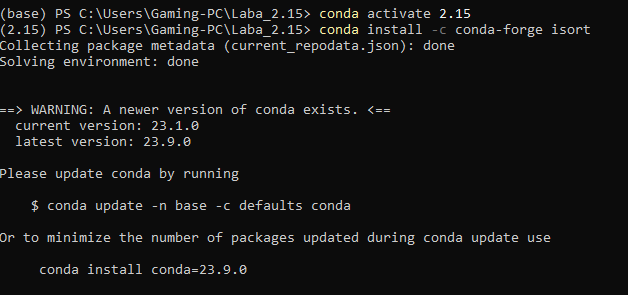


Рисунок 3. Создание виртуального окружения и установка пакетов

Задание 4.Создала проект PyCharm в папке репозитория. Приступила к работе с примером. Добавила 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() .

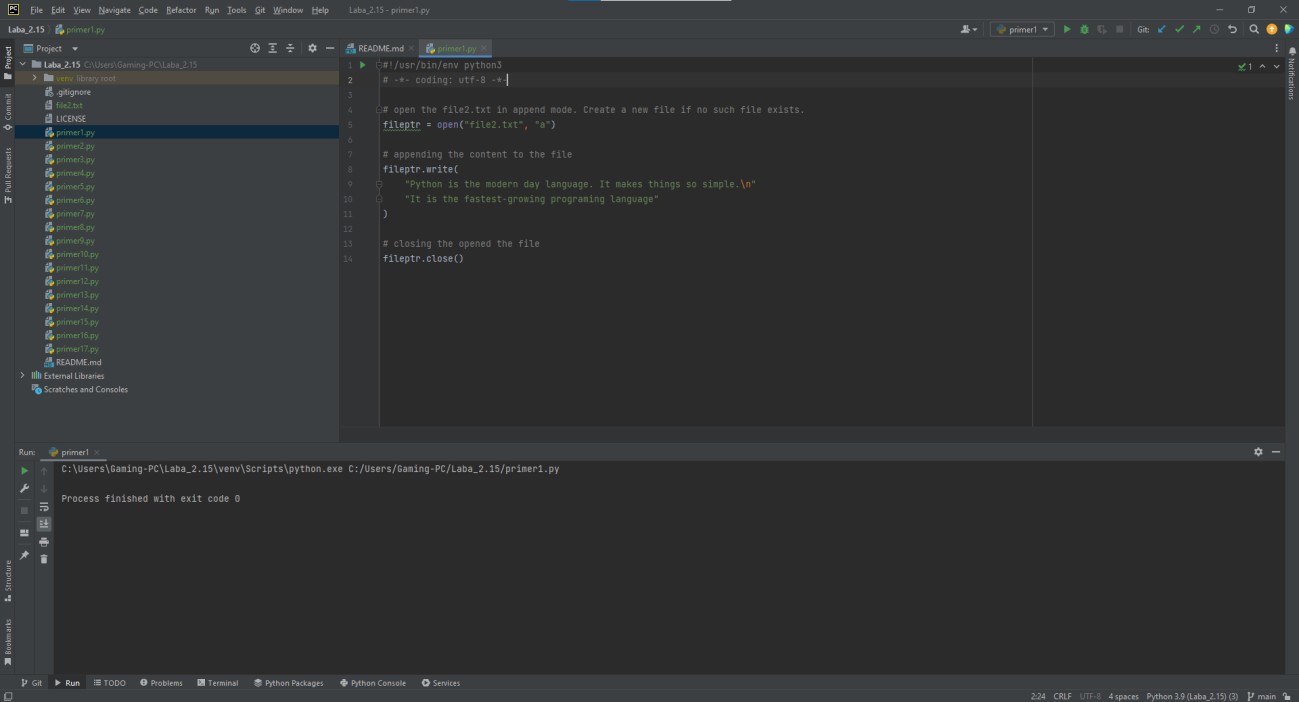


Рисунок 4. Реализация примера 1

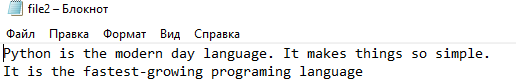


Рисунок 5. Результат примера 1 Создала новый файл под названием primer2.py.

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

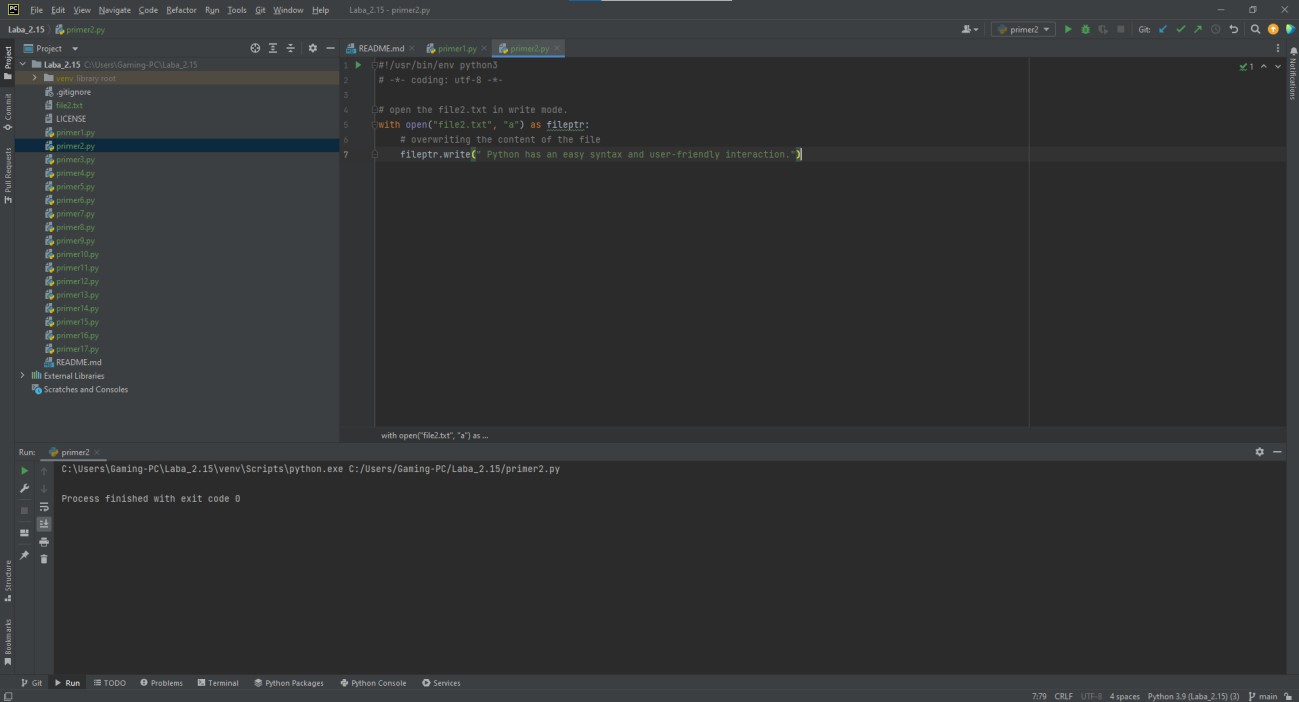


Рисунок 6. Реализация второго примера

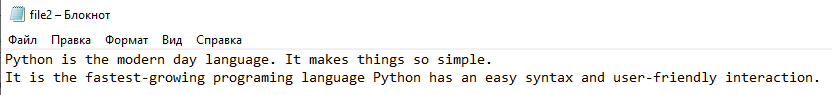


Рисунок 7. Результат второго примера Создала новый файл под названием primer3.py

Условие примера:чтение строк с помощью метода readline()

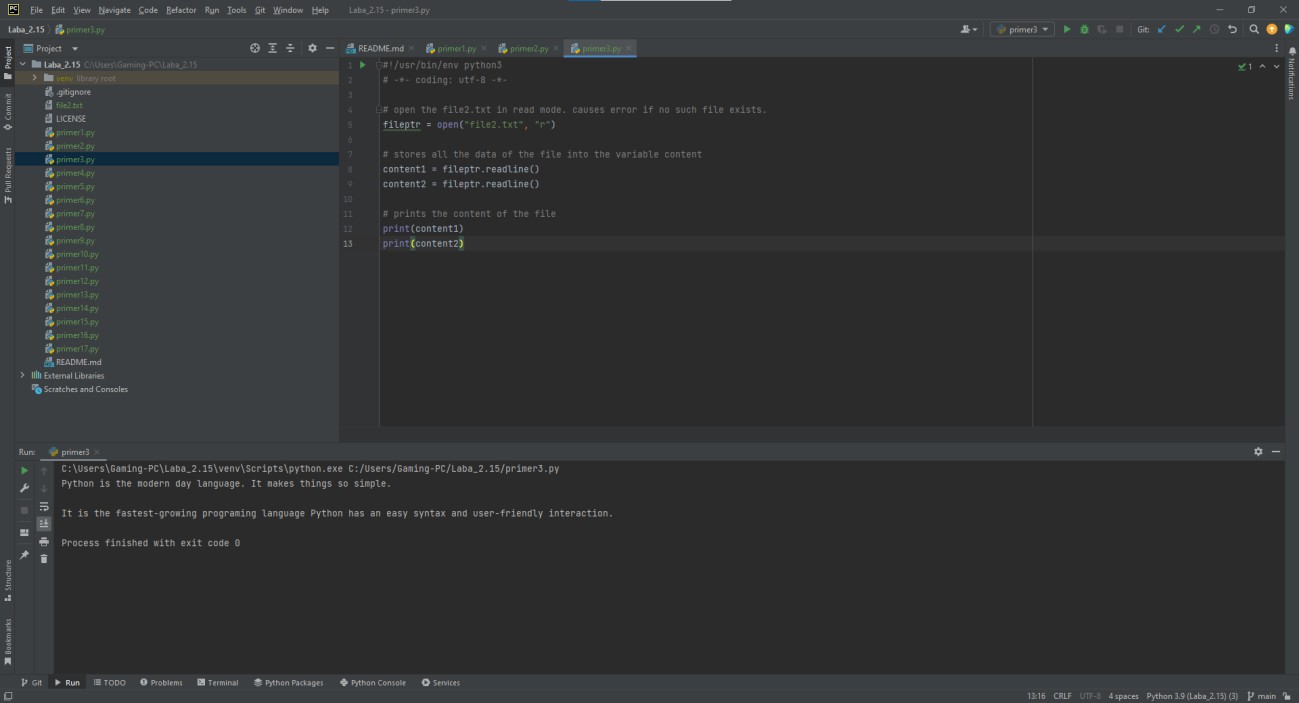


Рисунок 8. Реализация третьего примера Создала новый файл под названием primer4.py

Условие примера: чтение строк с помощью функции readlines()

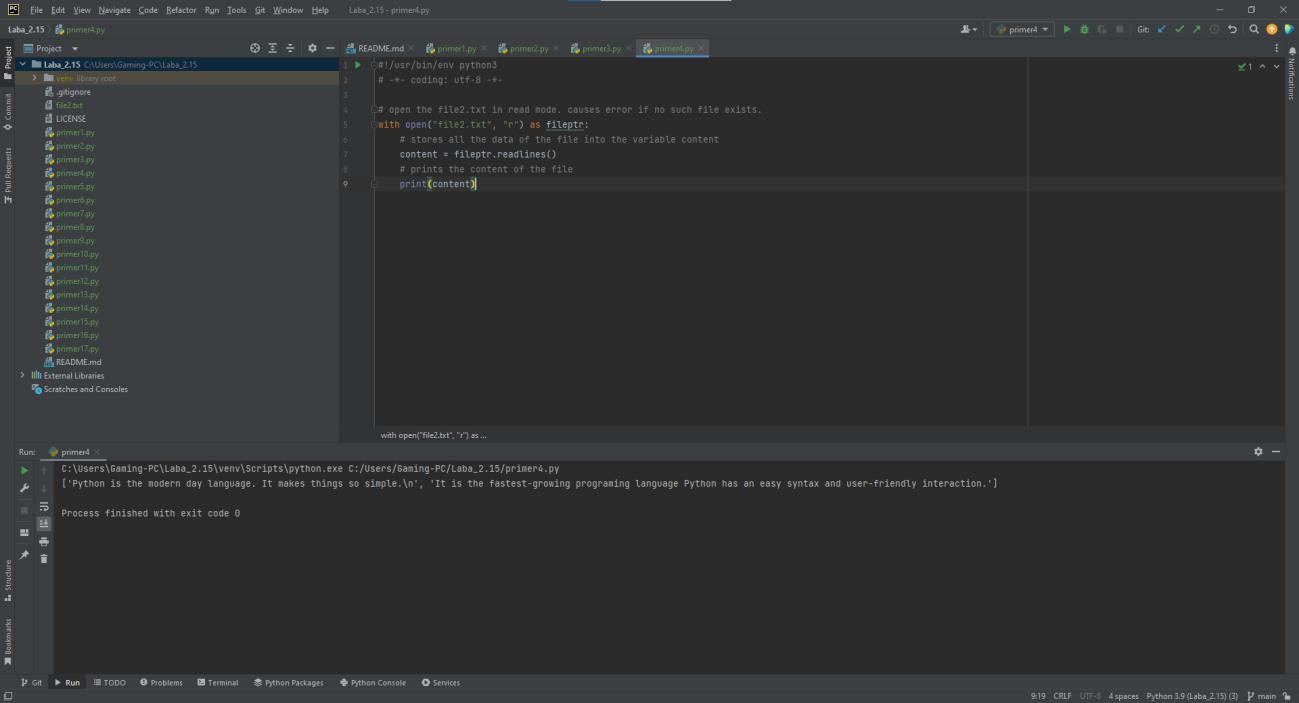


Рисунок 9. Реализация четвертого примера Создала новый файл под названием primer5.py

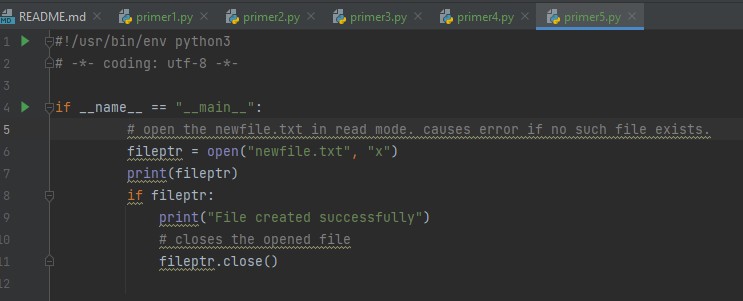


Рисунок 10. Реализация пятого примера Создала новый файл под названием primer6.py

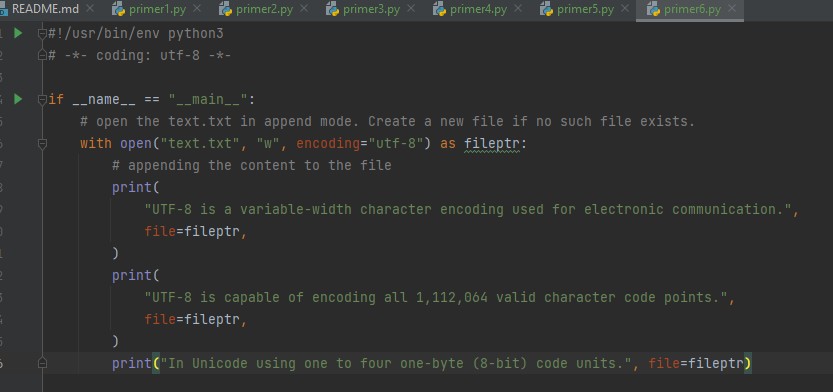


Рисунок 11. Реализация шестого примера

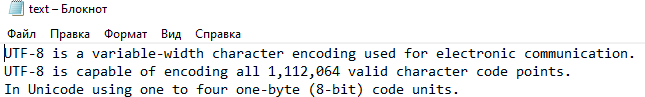


Рисунок 12. Результат шестого примера Создала новый файл под названием primer7.py

Условие примера: Написать программу, которая считывает текст из файла и выводит на экран только предложения, содержащие запятые. Каждое предложение в файле записано на отдельной строке.

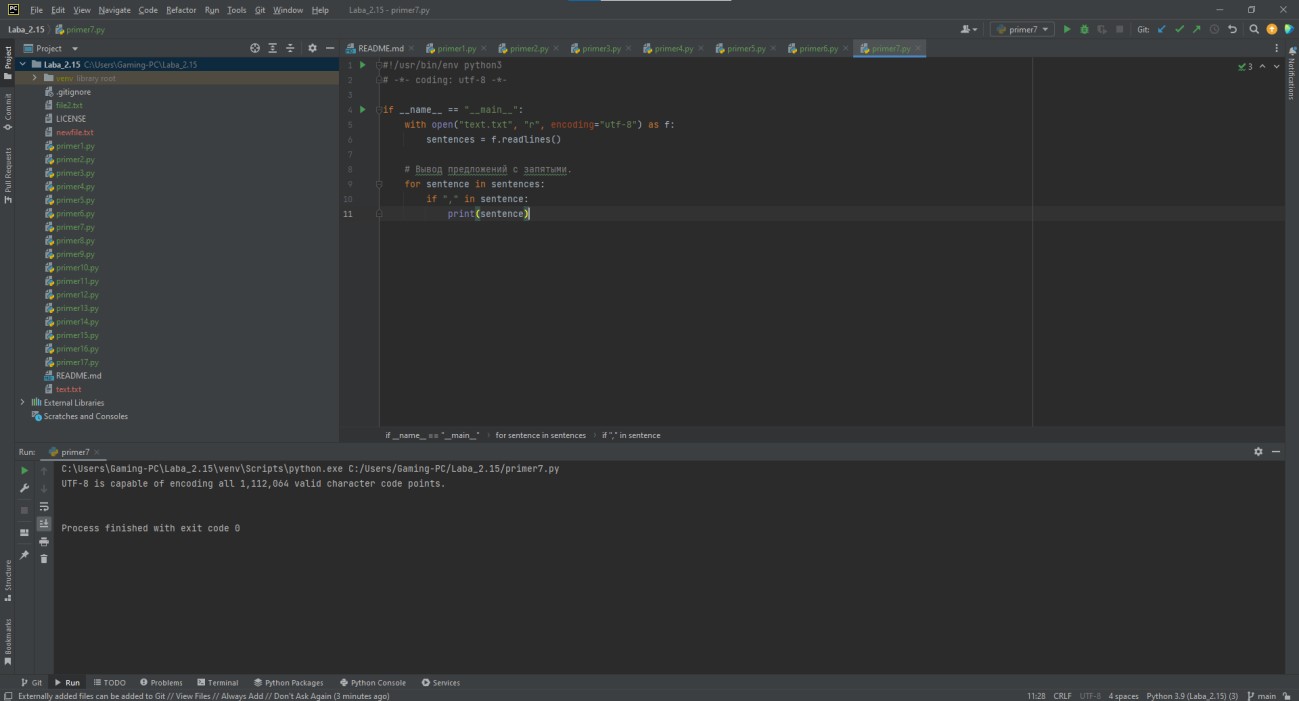


Рисунок 13. Реализация седьмого примера Создала новый файл под названием primer8.py

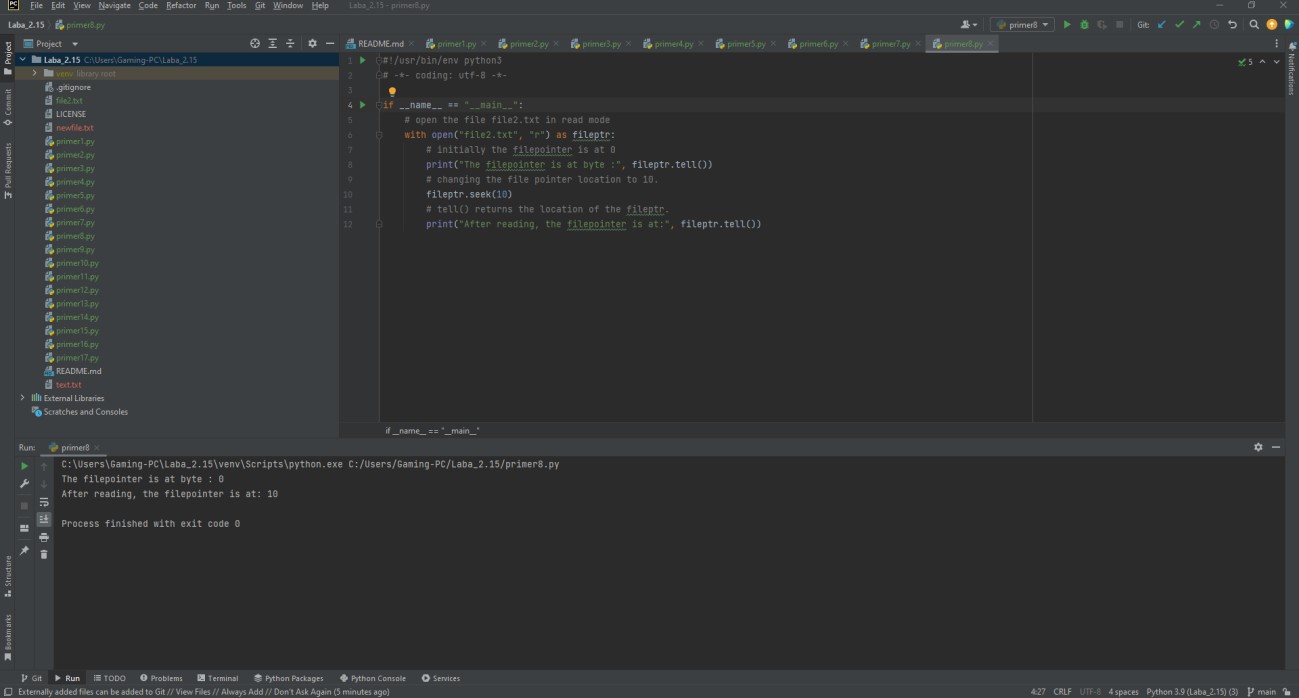


Рисунок 14. Реализация восьмого примера Создал новый файл под названием primer9.py

Условие примера:Приведенный выше код переименует файл file2.txt в file3.txt в текущей рабочей папке.

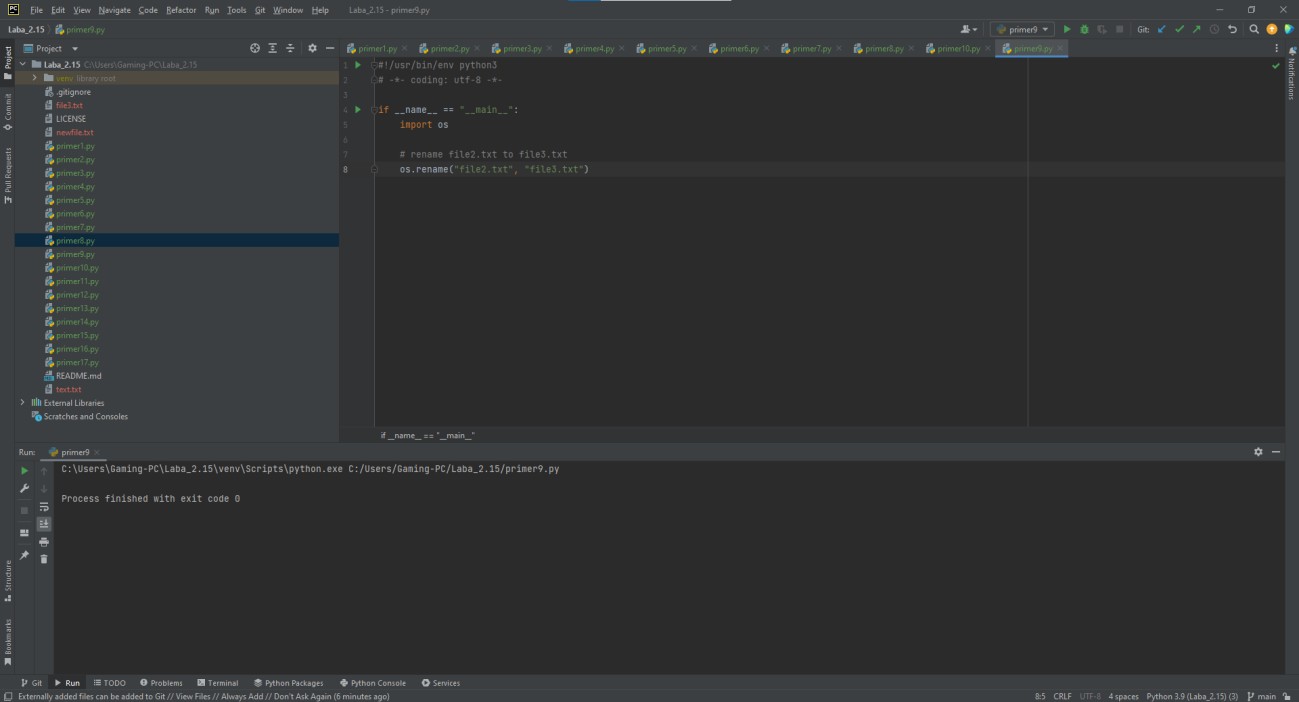


Рисунок 15. Реализация девятого примера

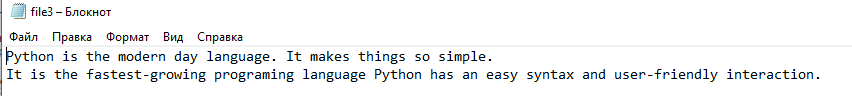


Рисунок 16. Результат девятого примера Создала новый файл под названием primer10.py

Условие примера:Приведенный выше код удалит файл file3.txt в текущей рабочей папке.

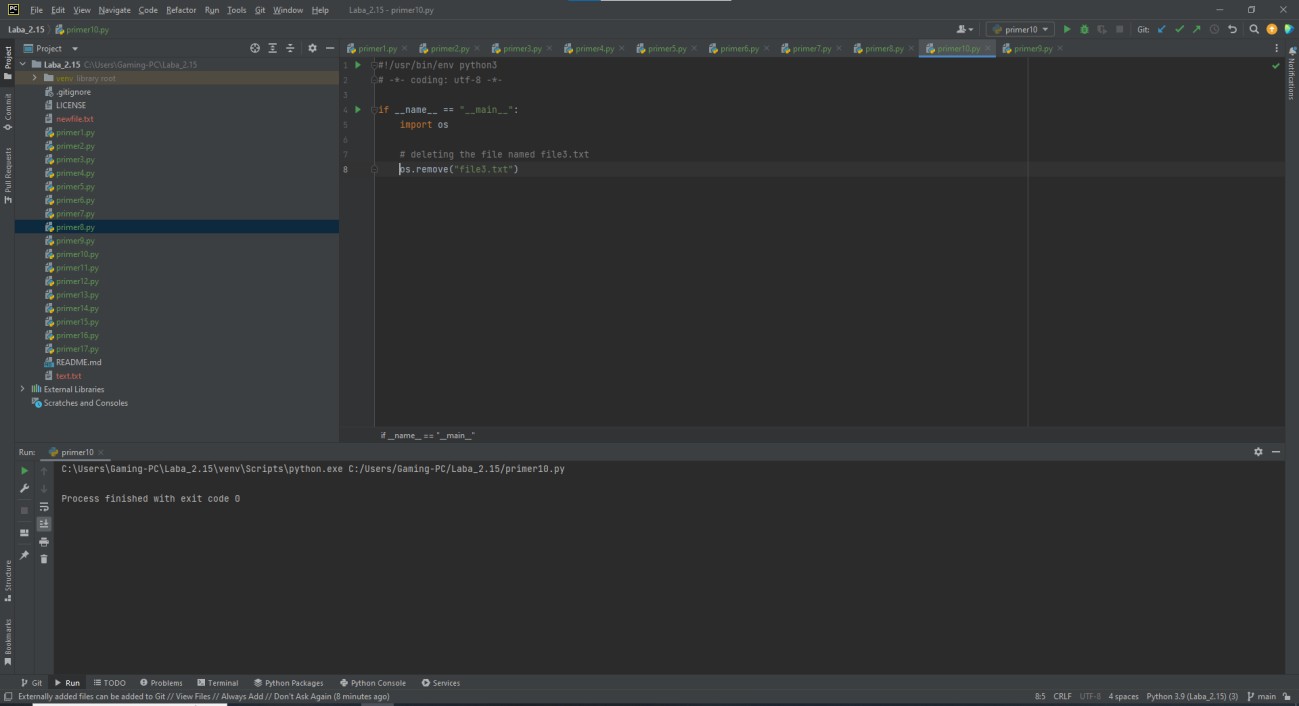


Рисунок 16. Реализация десятого примера Создала новый файл под названием primer11.py

Условие примера:Приведенный выше код создаст каталог new в текущей рабочей папке.

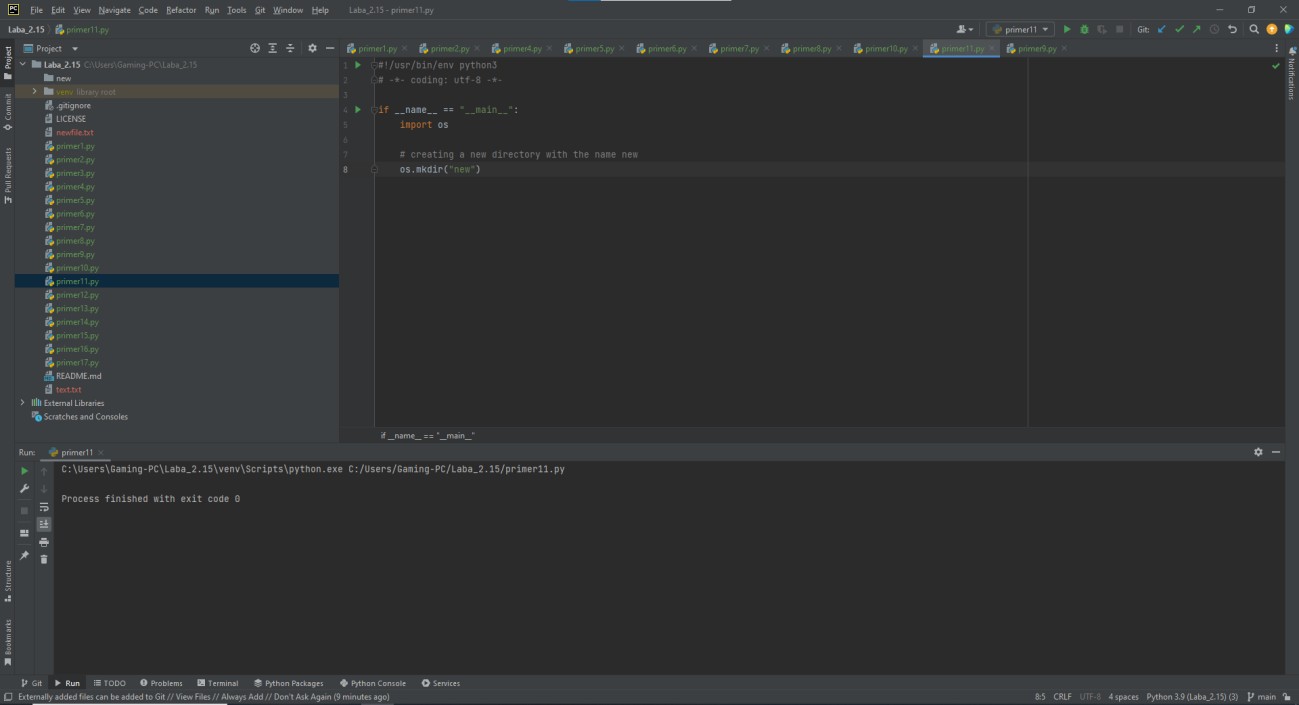


Рисунок 18. Реализация одиннадцатого примера

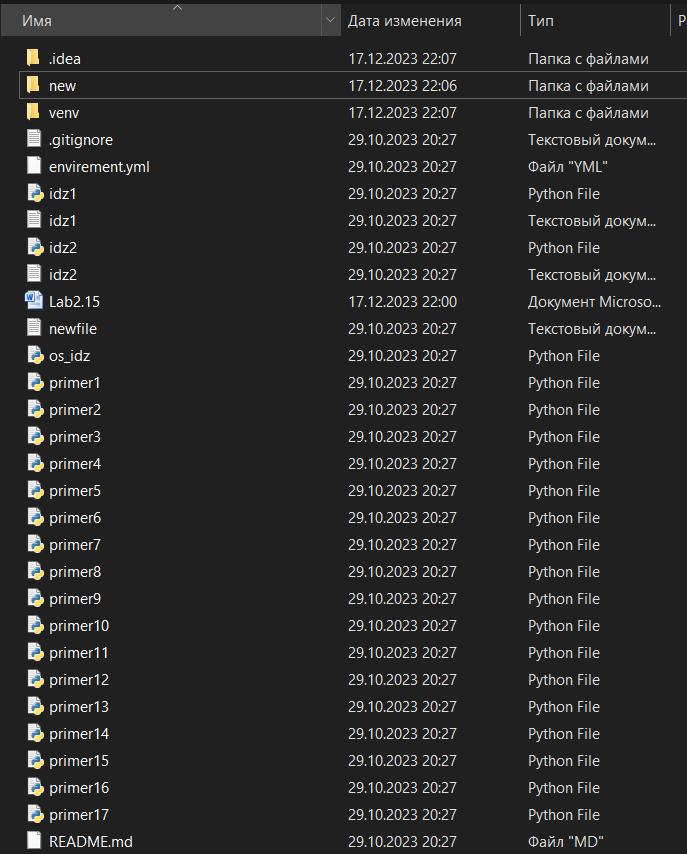


Рисунок 19. Результат программы Создала новый файл под названием primer12.py

Условие примера:Приведенный код выведет имя текущего рабочего каталога.

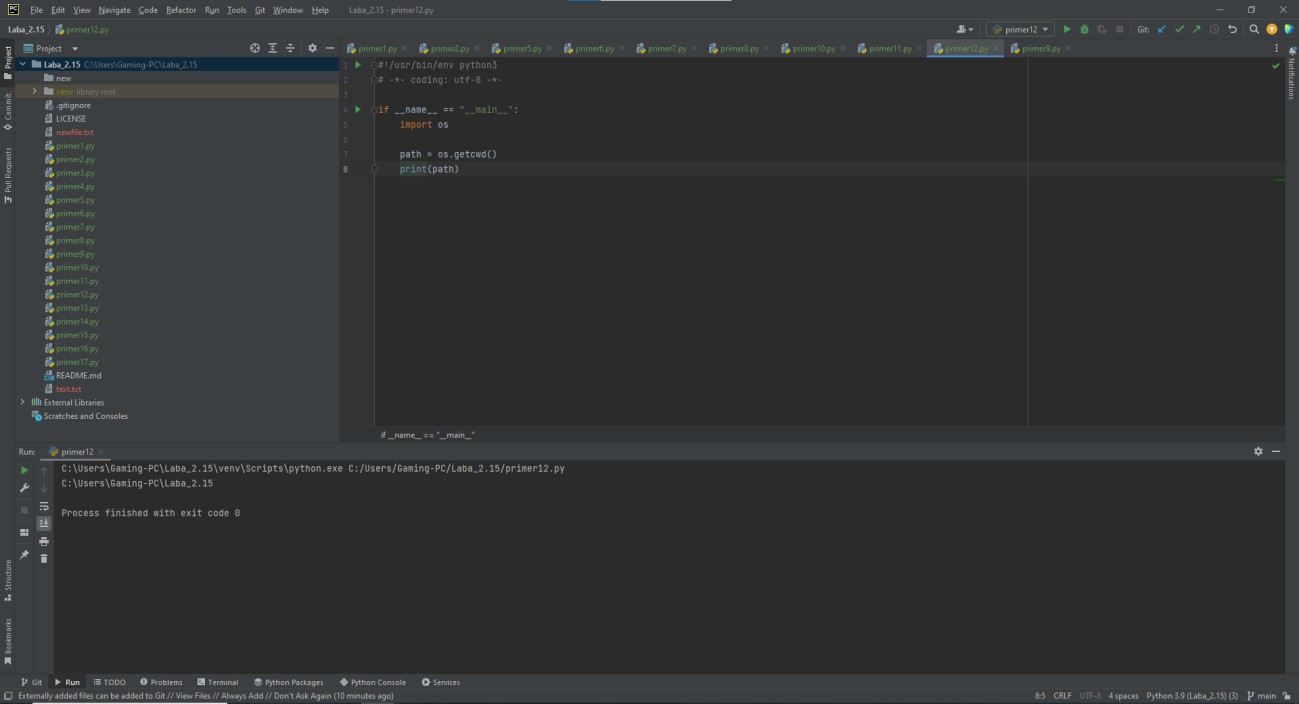


Рисунок 20. Реализация двенадцатого примера Создала новый файл под названием primer13.py

Условие примера:Приведенный код для операционной системы Windows сменит текущий каталог на C:\Windows и выведет новое имя текущего каталога.

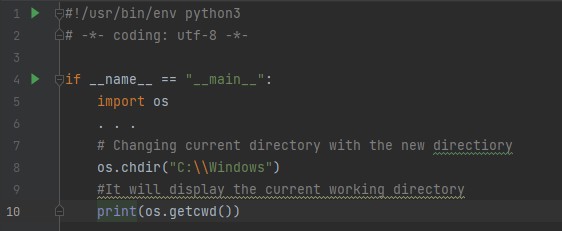


Рисунок 21. Реализация тринадцатого примера Создала новый файл под названием primer14.py

Условие примера: Данный код удалит ранее созданный каталог new при условии, что он не пуст и находится в текущей рабочей папке.

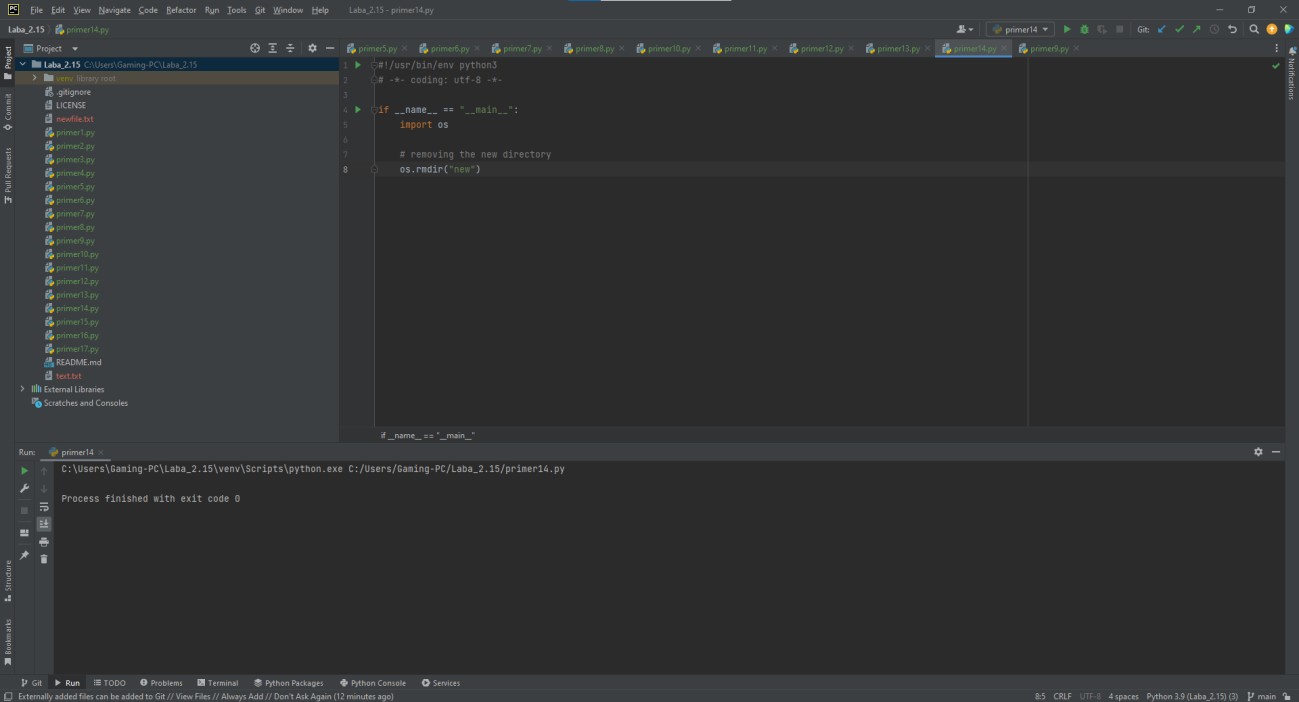


Рисунок 22. Реализация четырнадцатого примера Создала новый файл под названием primer15.py

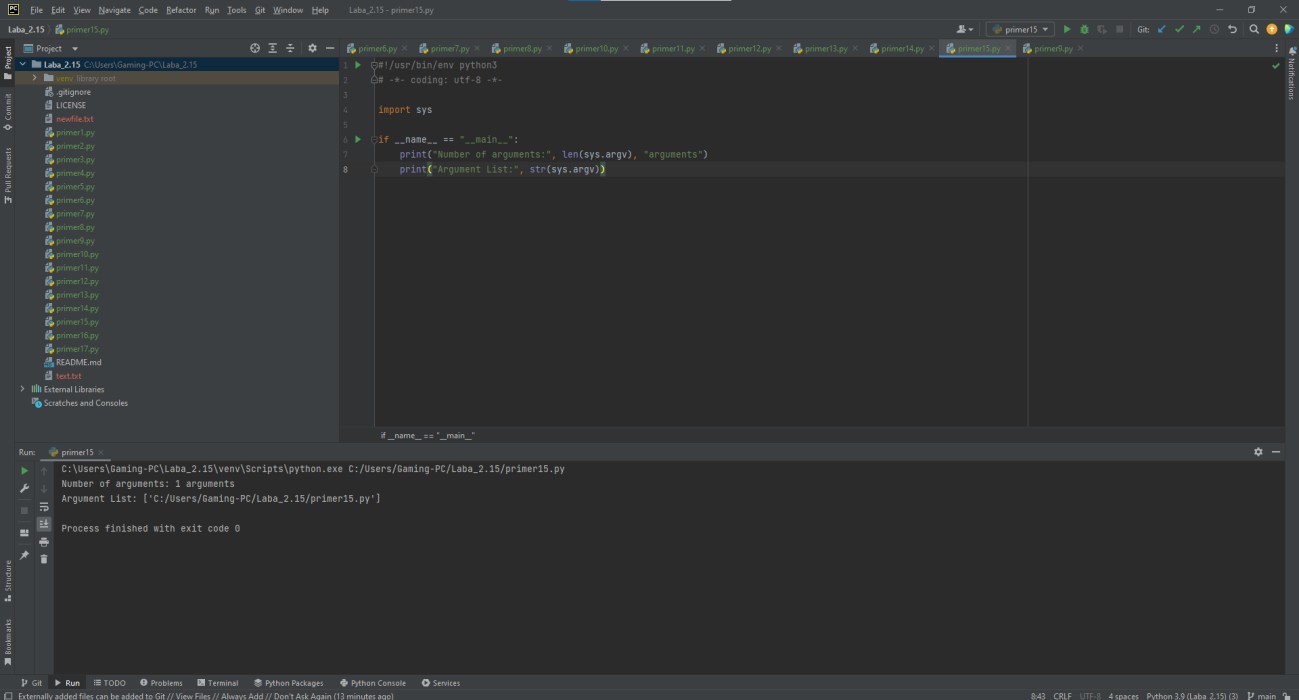


Рисунок 23. Реализация пятнадцатого примера Создала новый файл под названием primer16.py

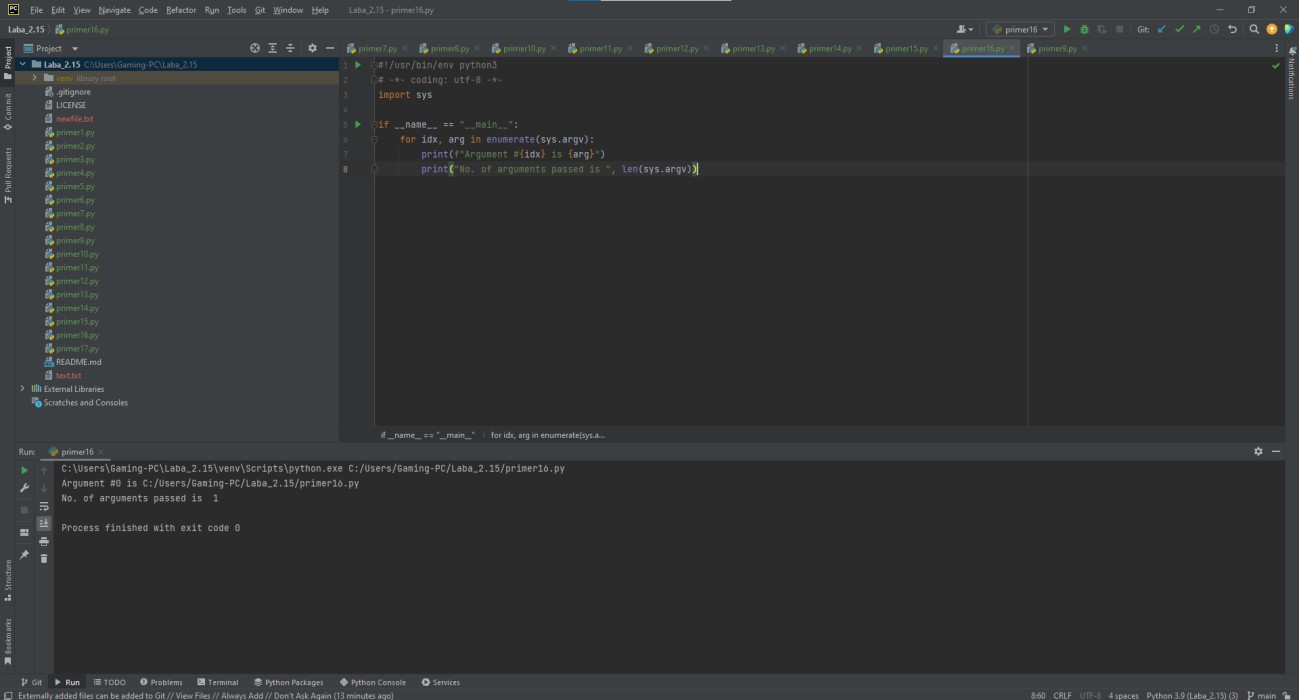


Рисунок 24. Реализация шестнадцатого примера Создала новый файл под названием primer17.py

Условие примера:Написать программу для генерации пароля заданной длины. Длина пароля должна передаваться как аргумент командной строки сценария.

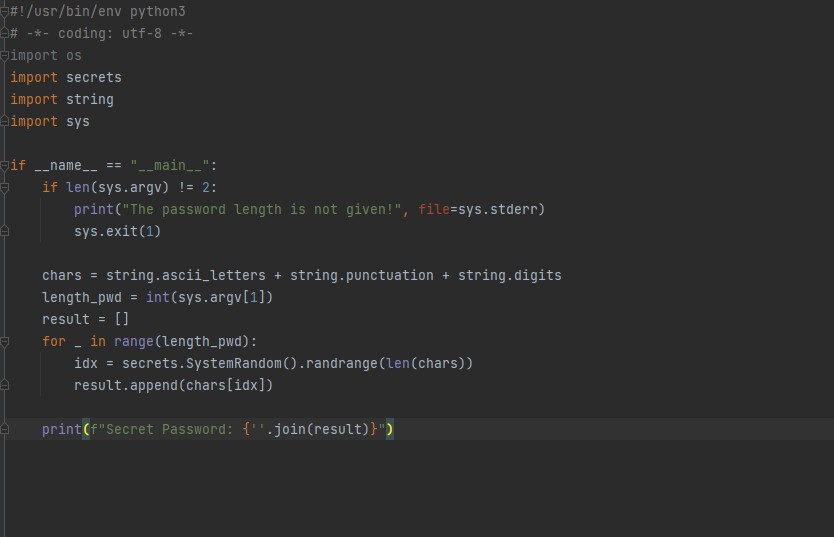


Рисунок 25. Реализация семнадцатого примера

# Задание 5.

Создала 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.

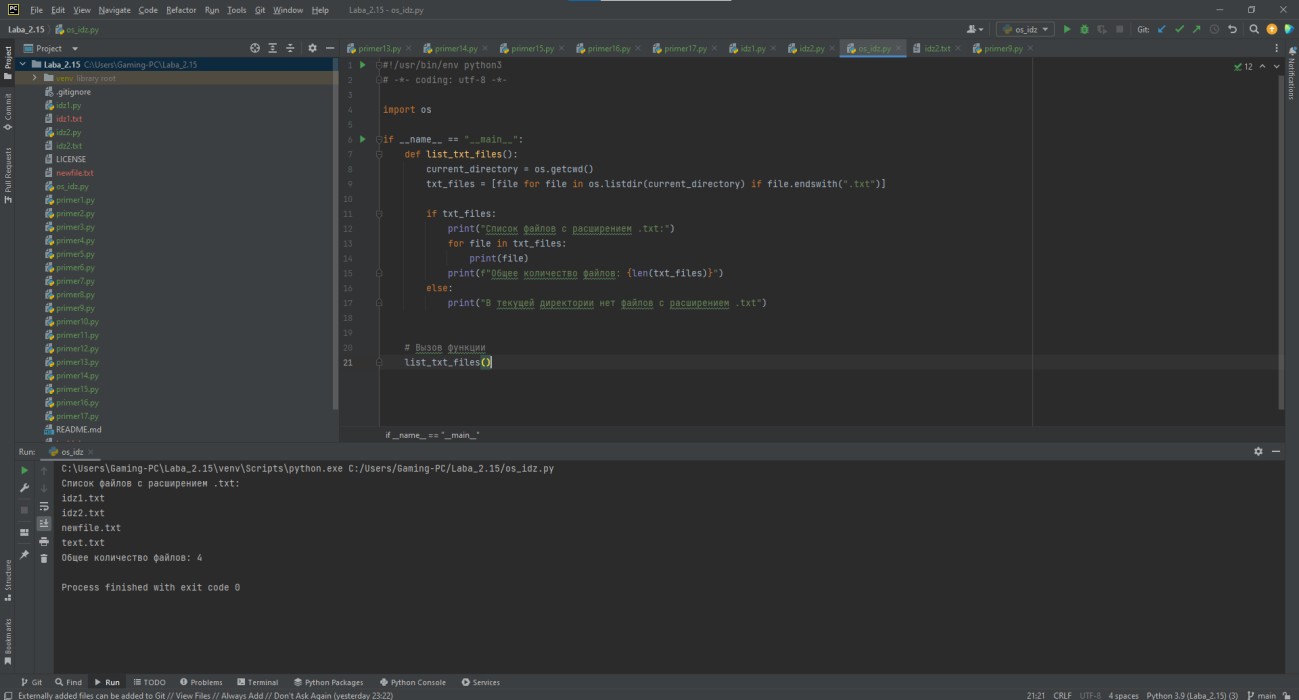


Рисунок 26. Программа индивидуального задания

# Индивидуальное задание

# Вариант 10

Создала 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.

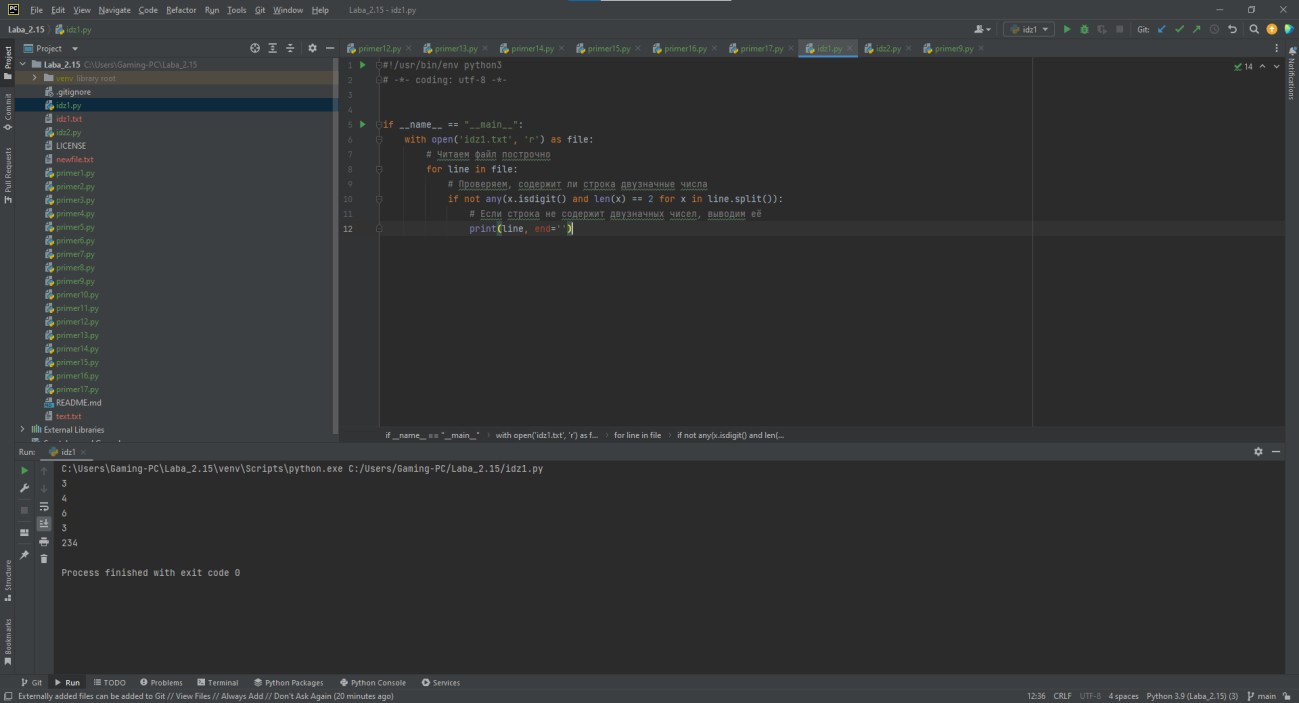


Рисунок 27. Реализация первого индивидуального задания

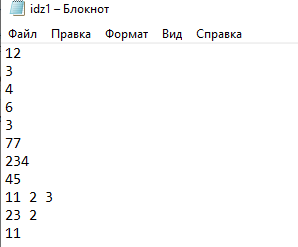


Рисунок 28. Результат программы Создала новый файл под названием idz2.py.

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

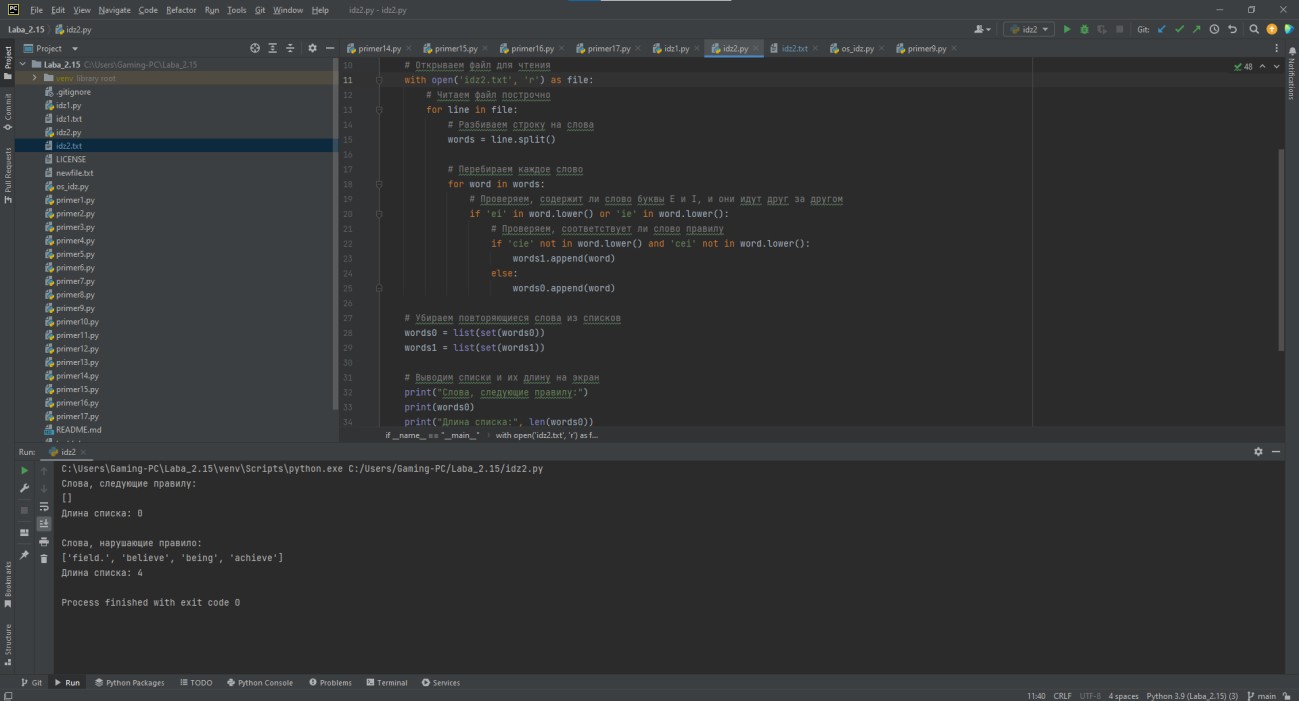


Рисунок 29. Реализация второго индивидуального задания

# Задание 6.

После выполнения работы на ветке develop, слила ее с веткой main и отправила изменения на удаленный сервер. Создала файл envirement.yml и деактивировала виртуальное окружение.



Рисунок 29. Деактивация виртуального окружения Ссылка: <https://github.com/Diana-372/Lab2.15.git>

Ответы на контрольные вопросы:

# Как открыть файл в языке Python только для чтения?

‘r’ - Открывает файл только для чтения.

# Как открыть файл в языке Python только для записи?

‘w’ – Открывает файл только для записи.

# Как прочитать данные из файла в языке Python?

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

# Как записать данные в файл в языке Python?

open('file.txt', 'w') открывает файл для записи. Параметр 'w' указывает режим открытия файла (в данном случае, для записи).

Контекстный менеджер with автоматически закрывает файл после использования блока кода внутри него.

write() используется для записи строки в файл. writelines() записывает список строк в файл.

Метод print() с параметром file=file записывает текст в файл. json.dump() записывает данные в формате JSON в файл.

# 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.

# 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.

# 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.

# 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:** приобрела навыки 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, получил аргументы командной строки.