1. Министерство образования и науки Российской Федерации
2. Санкт-Петербургский Политехнический Университет Петра Великого
3. —
4. Институт кибербезопасности и защиты информации

**ЛАБОРАТОРНАЯ РАБОТА № 3**

1. «Разработка текстового редактора»
2. по дисциплине «Объектно-ориентированное программирование»
3. Выполнил
4. студент гр. 5131001/20503 Черникова В. М.

<*подпись*>

1. Преподаватель
2. асс. преподавателя Кубрин Г. С.

<*подпись*>

1. Санкт-Петербург
2. 2024

# Цель работы

Целью работы – ознакомление с основными шаблоны проектирования программного обеспечения, получение навыка разработки приложений на базе архитектурного шаблона проектирования – Model View Controller.

# Задачи

В рамках лабораторной работы необходимо разработать VIM-подобный текстовый редактор на языке программирования Python 3, удовлетворяющий следующим требованиям:

1. Программа должна быть разработана с использованием архитектурного шаблона проектирования MVC.

2. На этапе проектирования программы должна быть разработана UML-диаграмма классов.

3. Программа должна иметь TUI – Text User Interface (текстовый пользовательский интерфейс). Для разработки TUI необходимо использовать встроенный Python-модуль curses с применением шаблона проектирования «адаптер» для интеграции модуля в основную логикой программы.

• интерфейс должен поддерживать отображение курсора и строки состояния (в качестве примера интерфейса следует рассмотреть оригинальную программу VIM);

• строка состояния должна отображать: текущий режим работы редактора (редактирование, ввод команды, окно помощи, поиск и прочие), имя файла, номер текущей строки и общее количество строк.

4. Программа должна работать с 1-байтовой кодировкой текста.

5. Для работы программы с текстом необходимо использовать Pythonобёртку над собственным классом MyString, разработанную в рамках выполнения первой лабораторной работы. В случае необходимости расширения либо корректировки поведения класса-обёртки MyString необходимо использовать шаблон проектирования «декоратор».

6. Программа должна иметь следующие режимы работы:

A. Режим навигации и редактирования

B. Режим ввода команд.

C. Режим ввода.

D. Режим поиска.

7. Проект должен содержать Unit-тесты для базового функционала программы, реализованный с помощью фреймворка pytest.

# Ход работы

Работа состояла из нескольких этапов:

1. Разработка адаптера для библиотеки curses;
2. Проектирование архитектуры приложения на базе шаблона MVC с использованием UML-диаграммы классов и разработка каркаса приложения с использованием шаблона MVC;
3. Разработка Model, View, Controller.

Итак, в результате прохождения этих шагов, были получены следующие результаты.

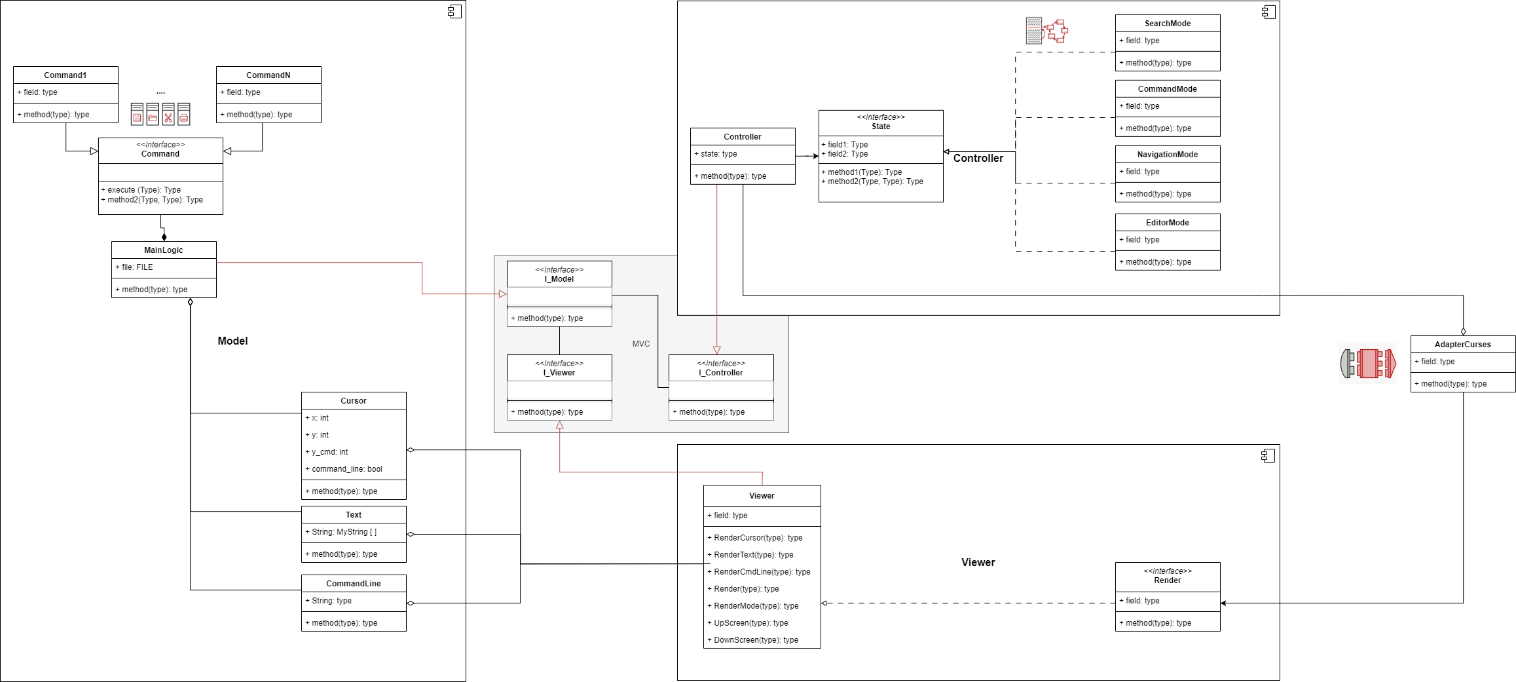
Первым делом, была разработана UML диаграмма классов, состоящая из фреймворка, адаптера для библиотеки curses и реализации классов Model, View, Controller. 

Рисунок - UML-диаграмма проекта

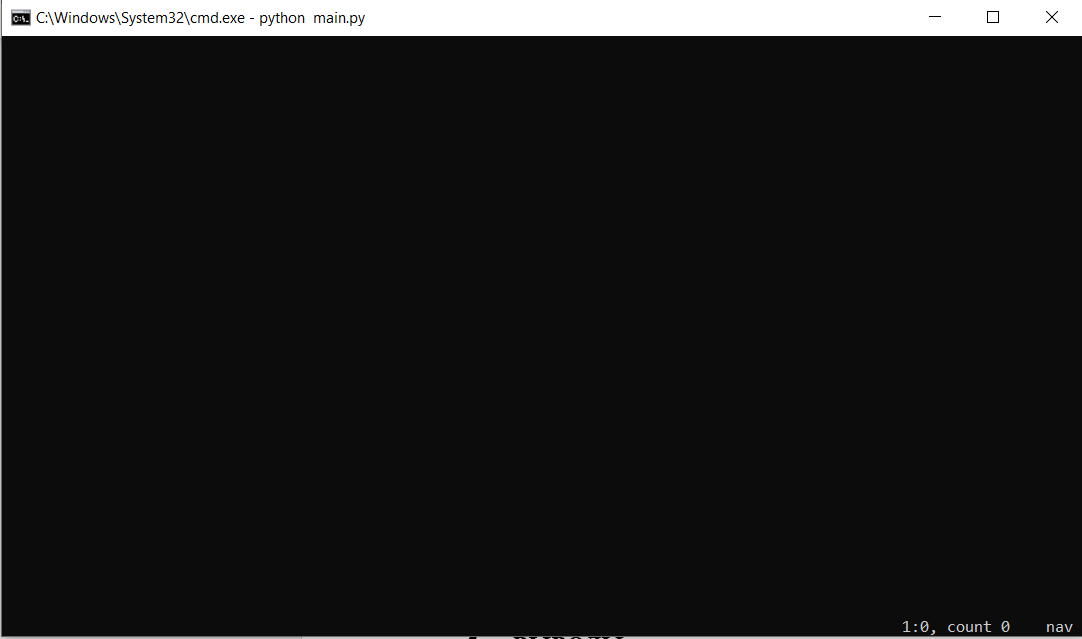
|  |  |
| --- | --- |
| **Название класса** | **Функционал** |
| Model | Отвечает за бизнес-логику программы. |
| Command | Родительский класс для паттерна команда |
| Controller | Управляет режимами редактора и передает модели имя команды, которую надо выполнить |
| Viewer | Отвечает за экран пользователя |
| CursesWrapper | Адаптер для библиотеки curses |
| State | Родительский класс для состояний |
| NavigationMode, FindMode, EditorMode, CommandMode | Классы, соответствующие своим состояниям |
| MyState | Перечисление для отслеживания состояний |
| Buffer | Класс для хранения текста и работы с ним |
| IModel | Интерфейс модели |
| IController | Интерфейс контроллера |
| IViewer | Интерфейс визуализатора |

После выполнения основной части работы было написано несколько тестов, чтобы проверить работу логики игры и класса событий.

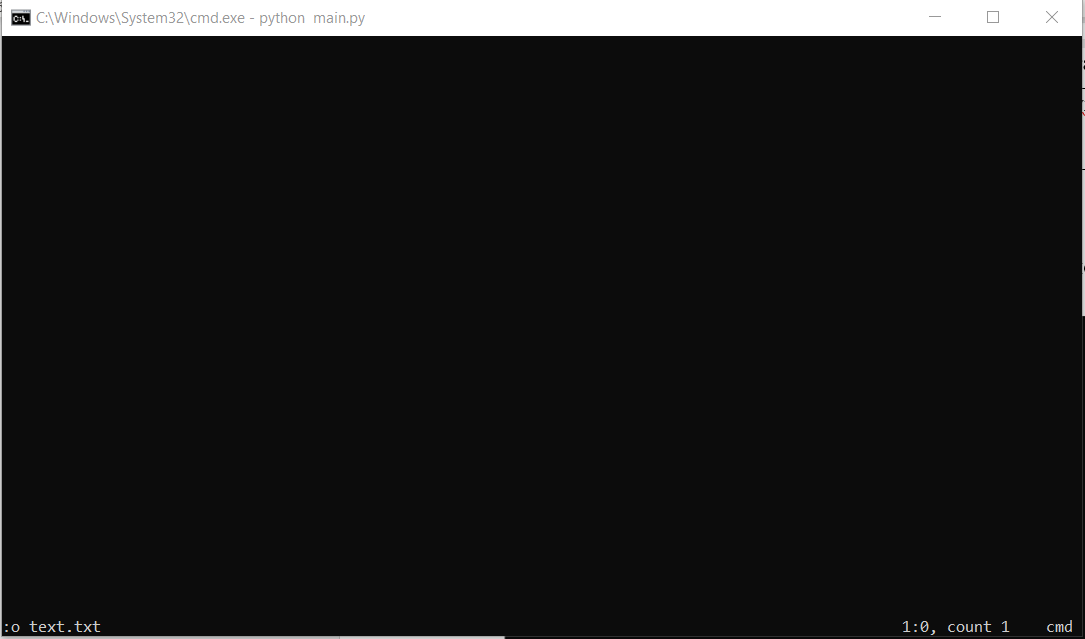
|  |  |
| --- | --- |
| **Название теста** | **Требования теста** |
| test\_text | При написании текста он должен появляться на экране и правильно двигаться курсор |
| test\_cursor | Проверка функции передвижения курсора |
| test\_init\_cursor | При открытии нового документа курсор остается на первой строке |
| test\_print\_and\_cursor | При написании текста должен правильно двигаться курсор при вызове комнд движения курсора |
| test\_beg\_lin | Проверка функции MoveCursorToBeginFile |
| test\_0 | Проверка поиска текста в файле |
| test\_del | Проверка DeleteTextSymbol и DeleteLine |

# Пример работы программы

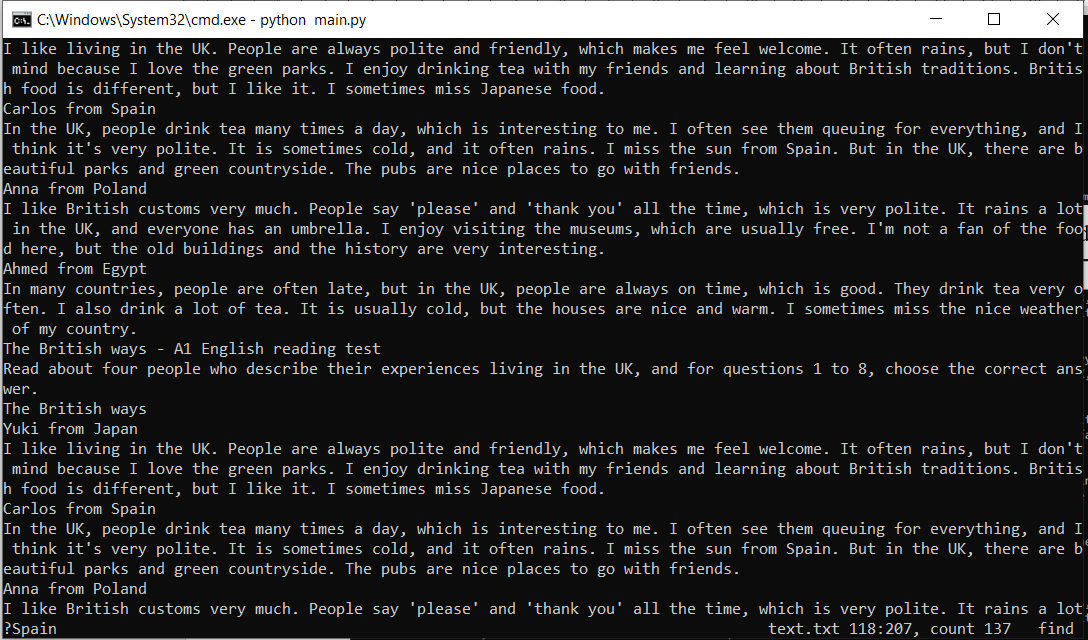
При запуске программы редактор открывается в навигационном режиме.

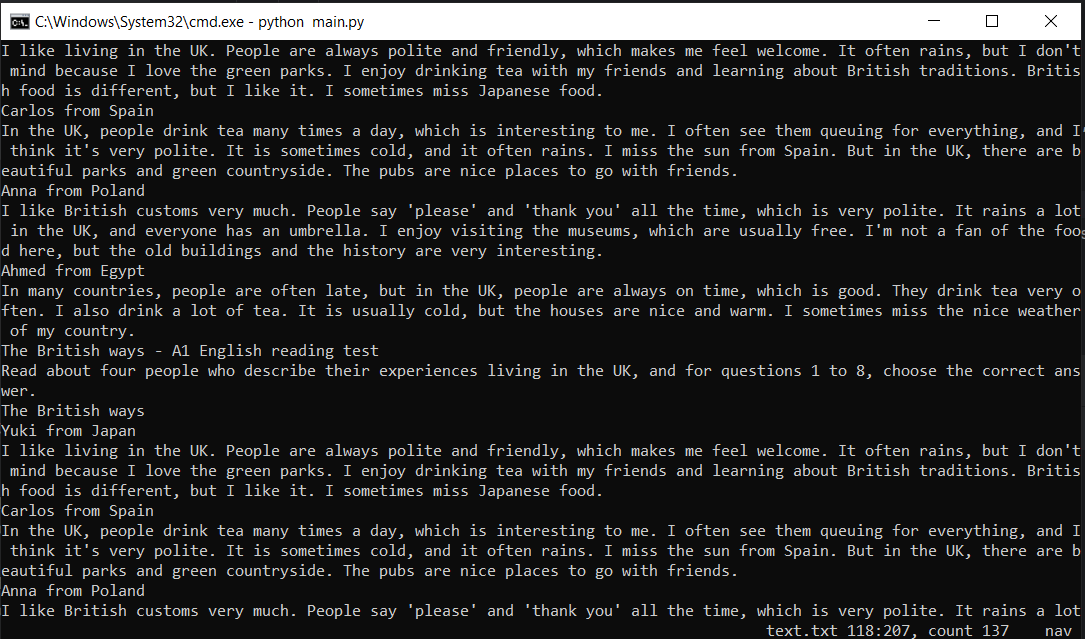


На последней строке справа можно видеть строку состояния из позиции курсора в документе, общее количество строк и режим редактора. Далее откроем любой существующий файл.

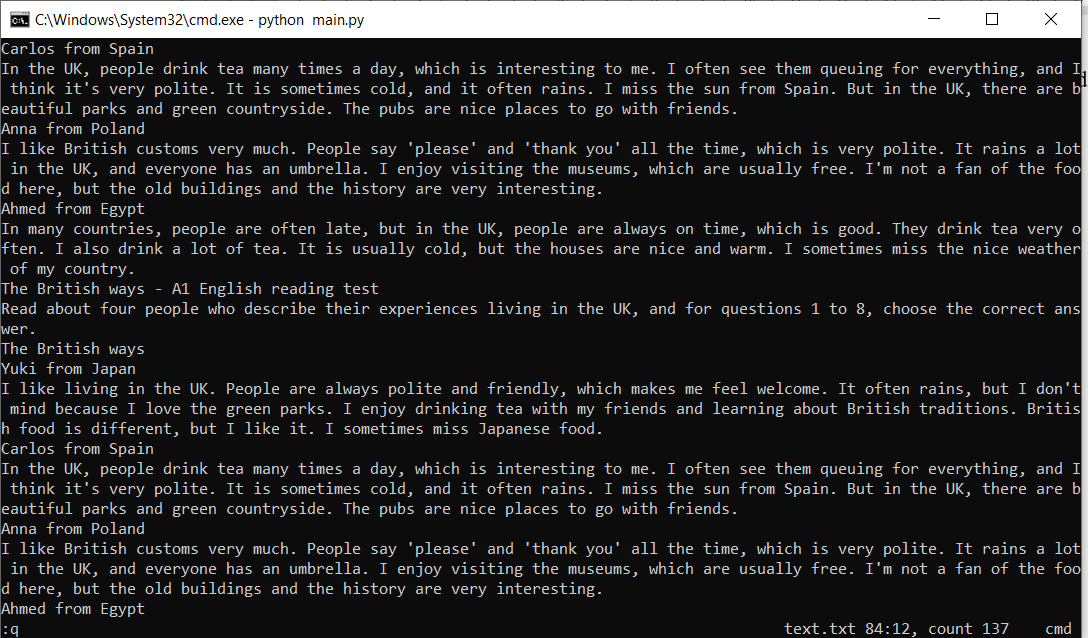


Давайте найдем каждое вхождение подстроки “Spain” с помощью find mode.

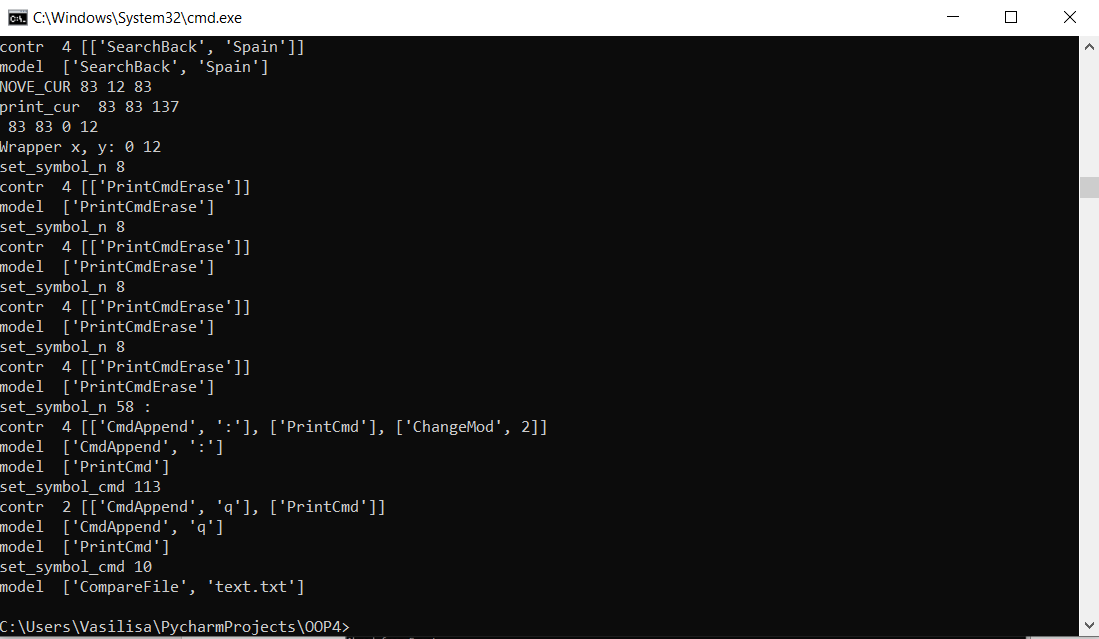




Далее выйдем из программы с помощью команды “:q”



И в командной строке можно увидеть логи редактора.



# выводы

В результате выполнения лабораторной работы были получены создания MVC - архитектуры. Также был получен опыт работы с библиотекой curses и были написаны тесты для проверки работоспособности. В лабораторной работе были использованы паттерны состояния, команды и адаптер.