**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**Факультет прикладной математики-процессов управления**

**Программа бакалавриата**

**“Большие данные и распределенная цифровая платформа”**

**ОТЧЕТ**

**по лабораторной работе №3**

**по дисциплине «Функциональное программирование»**

**на тему**

**«Система рекомендаций книг»**

**Студент гр. 23Б15-пу**

**Сурин И.С.**

**Преподаватель**

**Киямов Ж. У.**

**Санкт-Петербург**

**2024 г.**

Оглавление

1. [Цель работы 3](#_Toc185850987)
2. [Описание задачи 3](#_Toc185850988)
3. [Теоретическая часть 4](#_Toc185850989)
4. [Основные шаги программы 4](#_Toc185850990)
5. [Описание программы 5](#_Toc185850991)
6. [Рекомендации пользователя 8](#_Toc185850992)
7. [Рекомендации программиста 9](#_Toc185850993)
8. [Исходный код программы 9](#_Toc185850994)
9. [Контрольный пример 9](#_Toc185850995)
10. [Вывод 11](#_Toc185850996)
11. [Источники 11](#_Toc185850997)

**Цель работы**Разработка графического интерфейса для системы рекомендаций книг с использованием Python и PyQt5. Программа должна предоставлять пользователю следующие возможности:

* Просмотр списка рекомендованных книг.
* Фильтрацию и сортировку списка по различным критериям.
* Добавление книг в персональный список "Прочитать".

# **Описание задачи**

**1. Создание функционального интерфейса**

Основная цель — разработать удобный графический интерфейс для пользователей, позволяющий эффективно работать с рекомендованными книгами. Для этого будет реализовано несколько ключевых элементов:

* **Отображение списка рекомендованных книг:** Программа должна загружать и показывать список книг, хранящийся в файле books.json. Каждая книга будет содержать информацию о названии, авторе, жанре, году публикации и описании.
* **Фильтрация книг:** Пользователь должен иметь возможность фильтровать книги по следующим критериям:
  + Жанр (например, "Science Fiction", "Fantasy").
  + Год публикации (например, книги, изданные в определённом временном диапазоне).

Для этого будут предоставлены выпадающие списки или текстовые поля для ввода, с помощью которых пользователь сможет выбрать фильтры.

* **Сортировка списка:** Книги должны быть отсортированы по нескольким параметрам:
  + Рейтинг соответствия (приоритетным будет отображение книг, наиболее подходящих пользователю, например, через заранее заданные критерии).
  + Алфавитный порядок (по названиям или авторам).
  + Год публикации (по возрастанию или убыванию).

**2. Работа с пользовательскими действиями**

* **Добавление книги в список "Прочитать":** Каждая книга будет иметь кнопку для добавления в персональный список пользователя. Этот список будет сохраняться и отображаться в отдельном разделе программы.
* **Отображение и управление списком "Прочитать":** Пользователь сможет просматривать книги, добавленные в его список "Прочитать", и управлять этим списком (удалять книги, отмечать их как прочитанные и т. д.).

# **Теоретическая часть**

**Работа с PyQt5**  
PyQt5 используется для создания графического интерфейса, который позволяет создать кросс-платформенное приложение для отображения и управления списком книг. Этот фреймворк предоставляет инструменты для обработки пользовательских действий, динамического обновления данных и визуализации информации, что позволяет эффективно взаимодействовать с программой.

**Фильтрация и сортировка данных**  
Реализация фильтрации и сортировки в программе осуществляется через обработку данных с использованием моделей PyQt (например, QTableWidget). Для фильтрации используются жанры и год публикации, а для сортировки — различные параметры, такие как год публикации или алфавитный порядок. Это позволяет пользователю быстро находить нужную информацию в больших наборах данных.

**Структура данных и взаимодействие**  
Для хранения информации о книгах используется формат JSON, который легко обрабатывается в Python. Каждая книга представлена как словарь с полями: название, авторы, жанр, год публикации и описание. Персональный список "Прочитать" пользователя сохраняется отдельно, что позволяет отслеживать добавленные книги и управлять ими.

# **Основные шаги программы**

**Запуск приложения**  
При запуске программы открывается главное окно, которое содержит:

* + Таблицу со списком рекомендованных книг (данные загружаются из файла books.json).
  + Поля для фильтрации по жанру и году публикации.
  + Выпадающий список для выбора способа сортировки.
  + Таблицу для отображения списка книг, добавленных в раздел "Прочитать".

**Фильтрация книг**  
Пользователь может выбрать жанр и указать минимальный год публикации. Система фильтрует книги в реальном времени, обновляя таблицу с учетом введенных фильтров.

**Сортировка**  
При выборе параметра сортировки (по году публикации, алфавиту или рейтингу) список книг автоматически сортируется в соответствии с выбранным критерием.

**Добавление книги в список "Прочитать"**  
При нажатии на кнопку "Добавить в список" выбранная книга перемещается в персональный список "Прочитать", который отображается в отдельной таблице.

**Управление списком "Прочитать"**  
Пользователь может просматривать свой список.

# **Описание программы**

Программная реализация выполнена на Python 3.12.1 с использованием библиотеки **Tkinter** для создания графического интерфейса.

Таблица 1 main.py

|  |  |  |  |
| --- | --- | --- | --- |
| Функция | Описание | Тип входных данных | Тип выходных данных |
| \_\_init\_\_() | Конструктор класса, инициализирует окно приложения, загружает данные книг из файла и вызывает методы для построения пользовательского интерфейса. | str (путь к данным) | None (создает объект приложения) |
| Create\_widgets() | Основной метод для создания пользовательского интерфейса. Добавляет виджеты, такие как таблицы, кнопки и поля ввода, и устанавливает их расположение. | None | None |
| get\_recommendations() | Обрабатывает введенные фильтры (жанры, авторы, ключевые слова, минимальный год), применяет их к списку книг, сортирует результаты и отображает их в таблице рекомендаций. | None | None |
| add\_to\_read() | Добавляет выбранные пользователем книги из таблицы рекомендаций в список "Прочитать". | list[str] (выбранные строки в таблице) | None |
| update\_to\_read\_list() | Обновляет содержимое таблицы "Прочитать" после добавления новых книг. | list[dict] (данные книг в списке "Прочитать") | None |
| save\_recommendations() | Сохраняет текущий список рекомендаций в JSON-файл, выбранный пользователем через диалоговое окно. | None | None |
| load\_books() | Загружает данные книг из JSON-файла. | str (путь к файлу) | list[dict] (список книг) |
| |  | | --- | | calculate\_rating() |  |  | | --- | |  | | Рассчитывает рейтинг книги на основе предпочтений пользователя. | dict (данные книги), dict (предпочтения) | int (рейтинг) |
| recommend\_books() | Возвращает отсортированный список книг на основе рейтинга, рассчитанного по предпочтениям. | list[dict] (данные книг), dict (предпочтения) | list[dict] (рекомендации с рейтингами) |

# **Рекомендации пользователя**

Запуск программы: Для запуска используйте файл main.py.

Убедитесь, что файл с данными о книгах (books.json) находится в одной папке с программой.

Фильтрация: Укажите жанры, авторов, ключевые слова и минимальный год публикации в соответствующих полях ввода.

Нажмите кнопку "Рекомендовать", чтобы применить фильтры и получить список рекомендаций.

Сортировка: Список книг автоматически сортируется по рейтингу соответствия предпочтениям пользователя.

Добавление книги: Выберите книгу в таблице с рекомендациями. Выберите путь и имя файла для сохранения в формате JSON.

# **Рекомендации программиста**

Установка необходимых библиотек:Для работы программы требуется библиотека Tkinter (входит в стандартный пакет Python). Убедитесь, что файл с данными книг (books.json) корректен и содержит необходимые поля:

title (название книги), author (автор), genre (жанр), first\_publish\_year (год первой публикации), description (описание).

Проверка версии Python: Код протестирован и совместим с Python 3.7 и выше.Рекомендуется использовать Python 3.10 или более новую версию для стабильной работы.

# **Исходный код программы**

<https://github.com/Ignatio27/Func_prog_corse_2_spbu>

# **Контрольный пример**

1. **Запуск программы:** для запуска программы используйте модуль main.py**.**

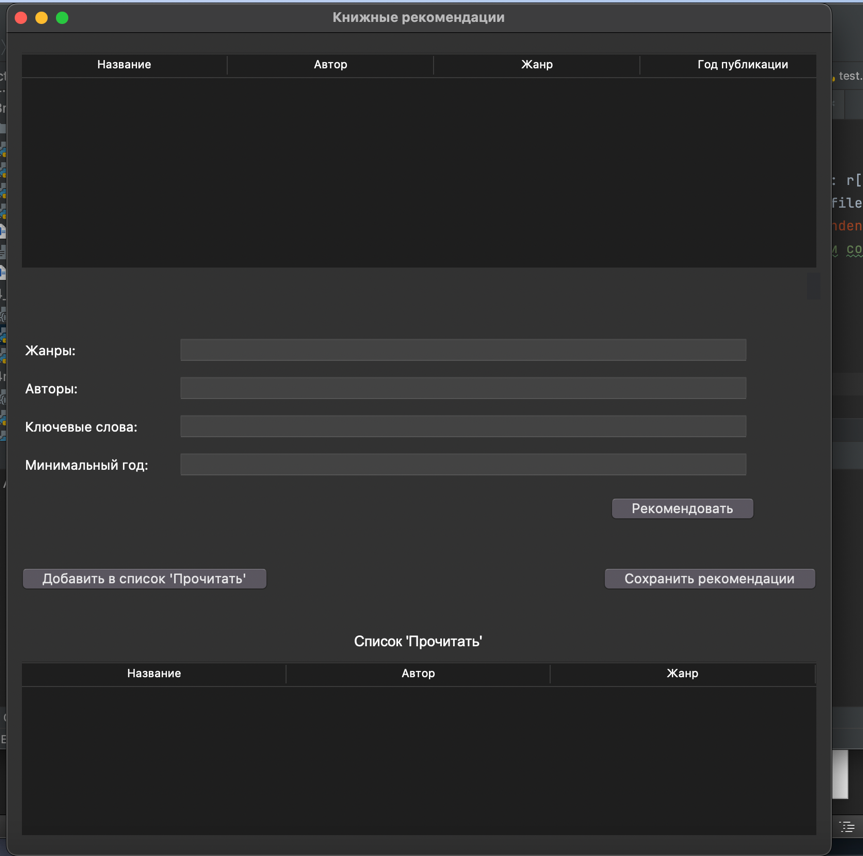


Рис. 1 Запуск программы

2. Применение фильтров и сортировки

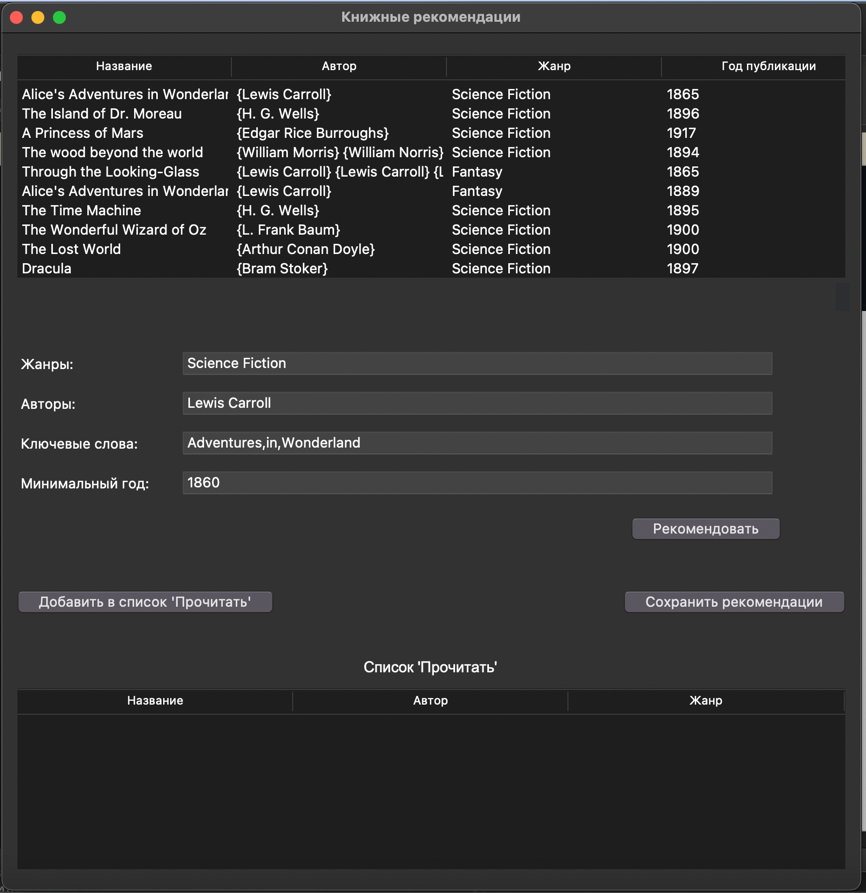


Рис.2 Сгенерированные и отсортированные рекомендации

3. Добавление книги в список "Прочитать":

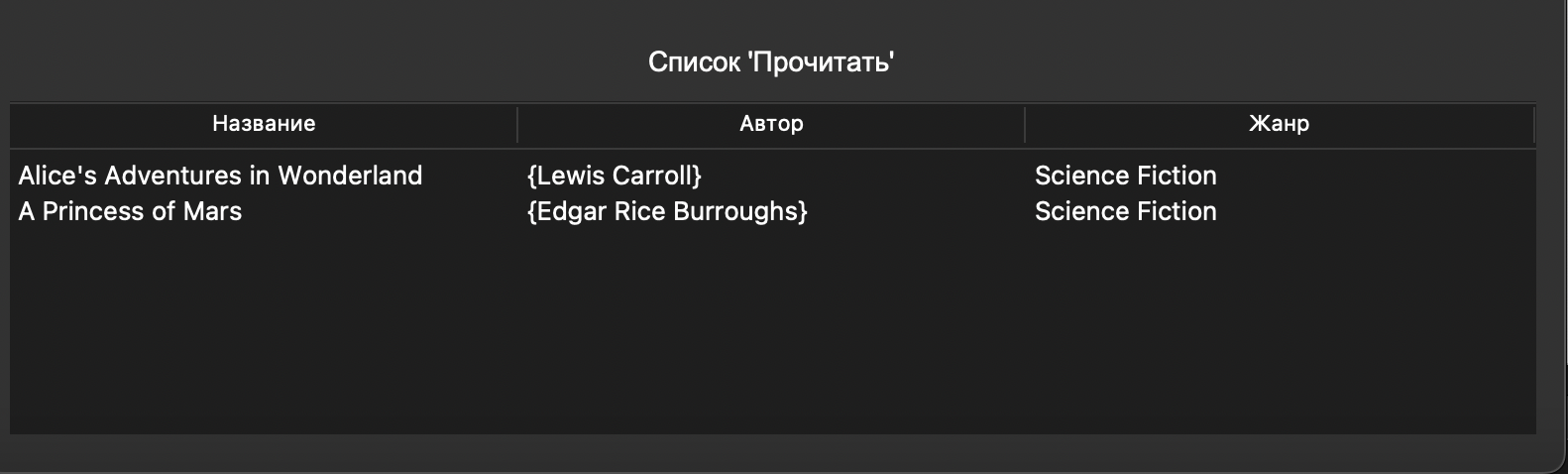


Рис. 3 Работа со списком «Прочитать»

4. Сохранение рекомендаций:

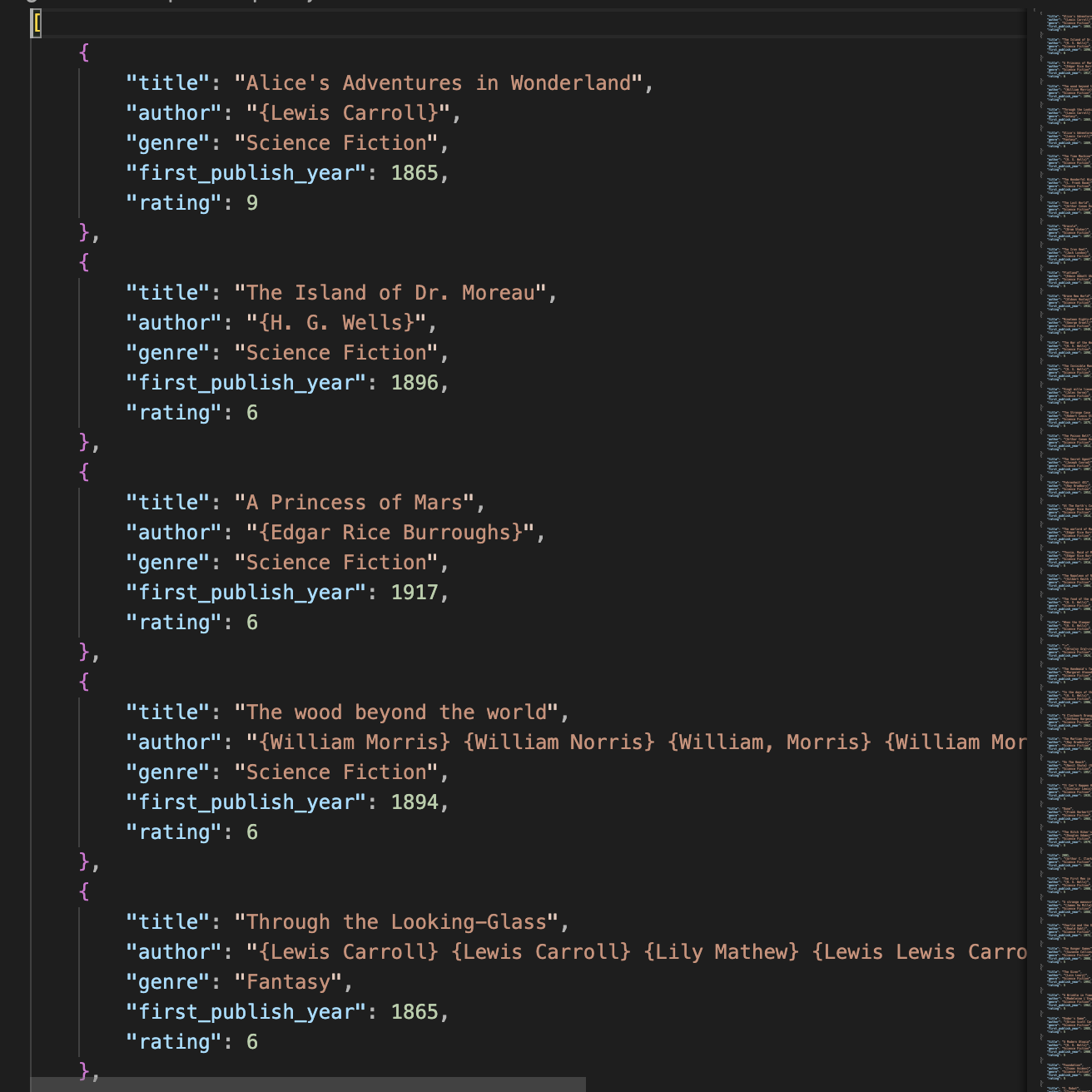


Рис. 4 Список рекомендаций в формате json

# **Вывод**

Реализация интерфейса для системы рекомендаций книг продемонстрировала эффективность использования PyQt5 при создании графических приложений. Программа обеспечивает удобные инструменты для фильтрации, сортировки и управления пользовательскими списками, предоставляя пользователю интуитивный и комфортный опыт взаимодействия.

# **Источники**

1. **PyQt5 //** https://pypi.org/project/PyQt5/ (дата обращения: 7.12.2024)