Федеральное государственное бюджетное профессиональное образовательное учреждение

Торжокский политехнический колледж

Федерального агентства по государственным резервам

Индивидуальное задание по МДК.01.01

«Разработка программных модулей»

**Оценка где???**

Выполнили студенты группы П-30

Смирнова Е.В.

Курчаков А.Д.

2025

**СОДЕРЖАНИЕ**

# **Раздел 1. Практическая значимость разрабатываемого ПС**

Данная программа разрабатывается для управления библиотекой книг. Она позволяет автоматизировать процессы добавления, удаления и поиска книг, а также отслеживать, какие книги выданы и кому. Это помогает библиотекарям эффективно управлять библиотечным фондом и предоставлять быстрый и удобный доступ к литературе для пользователей библиотеки.

Если бы не было данной программы все операции выполнялись бы вручную. Это потребовало бы значительных временных затрат и повышало риск ошибок. Без централизованной базы данных поиск нужных книг занимал бы гораздо больше времени, особенно в больших библиотеках с обширным фондом. Также ведение записей вручную сопряжено с риском утраты данных из-за человеческих ошибок, случайного стирания или повреждения бумажных носителей.

Задачи:

1. Создать полуавтоматическую систему управления базой данных «Библиотека».

2. Разработать удобный интерфейс для работы с базой данных «Библиотека».

3. Реализовать механизм проверки правильности добавления, поиска, удаления, записи и возврата книг.

Сложности:

* Работа с файлами – нужно корректно загружать и сохранять книги, чтобы не потерять данные.
* Простота использования – интерфейс должен быть интуитивным, даже для преподавателей без технических навыков.

В данный момент процесс добавления новой книги происходит следующим образом:

* библиотекарь вручную вписывает данные о книге в журнал или базу данных.

Процесс удаления книги происходит следующим образом:

* библиотекарю необходимо найти книгу в журнале или базе данных и вручную удалить ее.

Процесс поиска книги по автору происходит следующим образом:

* библиотекарь должен просмотреть все записи о книгах вручную, чтобы найти книгу нужного автора.

Процесс записи книги на человека:

* библиотекарь вручную вписывает книгу в карточку читателя.

Процесс возврата книги:

* библиотекарь вручную вычёркивает книгу из карточки читателя.

После введения программы данные процессы станут автоматизированными. Библиотекарь сможет вводить данные через интерфейс программы. В свою очередь программа сама сохраняет и обрабатывает все данные, позволяет моментально выводить данные по запросу пользователя. Благодаря данной программе сократится количество человеко-часов на 50%.

# **Раздел 2. Функциональные требования к ПС**

Основные функции программы:

1. Добавление новых книг:

Пользователь может ввести информацию о новой книге, включая ее название, автора, издательство, год издания и количество экземпляров. Это позволяет легко пополнять библиотечный фонд новыми книгами.

(программы еще нет, нужно описывать в будущем времени)

Чтобы добавить новую книгу, необходимо будет выполнить следующие шаги:

1. Ввести 1 в основном меню для выбора опции «Добавить книгу»;
2. Подтвердить нажав клавишу «Enter»;
3. Ввести название книги;
4. Подтвердить нажав клавишу «Enter»;
5. Ввести автора;
6. Подтвердить нажав клавишу «Enter»;
7. Ввести издательство;
8. Подтвердить нажав клавишу «Enter»;
9. Ввести год издания;
10. Подтвердить нажав клавишу «Enter»;
11. Ввести количество экземпляров;
12. Подтвердить создание новых экземпляров книги нажав клавишу «Enter»;
13. На экране появится сообщение «Добавлено … экземпляров книги.».

Название книги, автор и издательство должно вводиться в соответствии с кодировкой UTF8. Год издания и количество экземпляров должны вводится арабскими цифрами.

Если пользователь введёт год издания или количество экземпляров буквами, то на экране появится сообщение «Год издания и количество экземпляров должны вводиться числами.».

Если пользователь введёт отрицательный год издания или количество экземпляров, то на экране появится сообщение «Год издания и количество экземпляров должны быть больше нуля.».

1. Удаление книг:

Если какая-то книга устарела или была заменена новой версией, пользователь может удалить её из базы данных. Это помогает поддерживать актуальность библиотечного фонда.

Чтобы удалить книгу, необходимо будет выполнить следующие шаги:

1. Ввести 2 в основном меню для выбора опции «Удалить книгу»;
2. Подтвердить нажав клавишу «Enter»;
3. Ввести ID книги;
4. Подтвердить удаление книги нажав клавишу «Enter»;
5. На экране появится сообщение «Книга '…' (ID: …) удалена.».

Если книги с таким ID нет в библиотеке, то на экране появится сообщение «Книга не найдена.».

Если пользователь введёт ID буквами, то на экране появится сообщение «ID должно вводиться числом.».

Если пользователь введёт отрицательный ID, то на экране появится сообщение «ID должно быть больше нуля.».

ID должно вводится арабскими цифрами.

1. Поиск книг:

Программа предоставляет возможность искать книги по автору и по названию книги (частичному совпадению). Это особенно полезно для пользователей библиотеки, которые могут быстро найти нужную литературу.

Чтобы осуществить поиск книг по автору, необходимо будет выполнить следующие шаги:

1. Ввести 3 в главном меню для выбора опции «Поиск по автору»;
2. Подтвердить нажав клавишу «Enter»;
3. Ввести автора для поиска;
4. Подтвердить поиск книги нажав клавишу «Enter»;
5. Программа покажет все подходящие книги.

Если книг данного автора нет в библиотеке, то на экране появится сообщение «Книги не найдены.».

Если пользователь введёт пустого автора, то программа покажет все книги.

Автор книги должен вводиться в соответствии с кодировкой UTF8.

Чтобы осуществить поиск книг по названию книги (частичному совпадению), необходимо будет выполнить следующие шаги:

1. Ввести 4 в основном меню для выбора опции «Поиск по названию или частичному совпадению»;
2. Подтвердить нажав клавишу «Enter»;
3. Ввести название или несколько первых букв названия книги;
4. Подтвердить поиск книги нажав клавишу «Enter»;
5. Программа покажет все подходящие книги.

Если книги с данным названием нет в библиотеке, то на экране появится сообщение «Книги не найдены.».

Если пользователь введёт пустое название, то программа покажет все книги.

Название книги должно вводиться в соответствии с кодировкой UTF8.

1. Запись книги на человека:

В программе можно указать, кто и когда взял определенную книгу. Это помогает библиотекарю контролировать выдачу и возврат книг, предотвращая их потерю.

Чтобы записать книгу за определенным человеком, необходимо будет выполнить следующие шаги:

1. Ввести 5 в основном меню для выбора опции «Записать книгу»;
2. Подтвердить нажав клавишу «Enter»;
3. Ввести ID книги;
4. Подтвердить ID книги нажав клавишу «Enter»;
5. Ввести ФИО человека, который взял книгу;
6. Подтвердить ФИО нажав клавишу «Enter»;
7. На экране появится сообщение «Книга '…' записана за '…'.».

Если книги с таким ID нет в библиотеке, то на экране появится сообщение «Книга не найдена.».

Если пользователь введёт ID буквами, то на экране появится сообщение «ID должно вводиться числом.».

Если пользователь захочет взять уже записанную книгу, то на экране появится сообщение «Книга уже взята.».

ID должно вводится арабскими цифрами.

ФИО должно вводиться в соответствии с кодировкой UTF8.

1. Возврат книги:

Чтобы вернуть книгу, необходимо будет выполнить следующие шаги:

1. Ввести 6 в основном меню для выбора опции «Вернуть книгу»;
2. Подтвердить нажав клавишу «Enter»;
3. Ввести ID книги;
4. Подтвердить ID книги нажав клавишу «Enter»;
5. На экране появится сообщение «Книга возвращена.».

Если книги с таким ID нет в библиотеке или она не взята, то на экране появится сообщение «Книга не найдена или не была взята.».

ID должно вводится арабскими цифрами.

Если пользователь введёт ID буквами, то на экране появится сообщение «ID должно вводиться числом.».

ID должно вводится арабскими цифрами.

1. Выход из программы:

Чтобы выйти из программы, необходимо будет выполнить следующие шаги:

1. Ввести 7 в основном меню для выбора опции «Выход»;
2. Подтвердить нажав клавишу «Enter».

Программа должна проверять, что вводимые данные соответствуют необходимым типам данным (число – int, строка - string).

# **Раздел 3. Модульная структура проектируемого ПС**

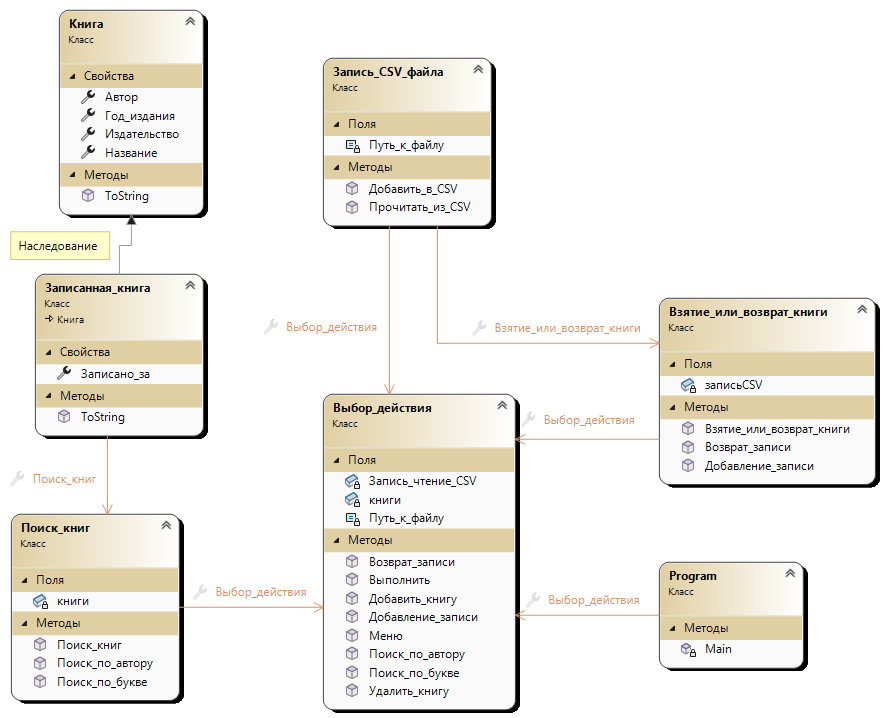


Рисунок 3 – Диаграмма классов

Какие поля или методы класса «Книга» наследует класс «Записанная\_книга»? Связь должна быть содержит или включает, наследования тут никакого нет.

Классы, поля и методы на диаграмме по стандарту именуется на английском языке, названия разделов также на английском

У полей не указаны типы данных

У методов не указаны типы возвращаемых значений

Связи подписаны неверно, должен быть указан глагол или отглагольное существительное, которое показывает, как связаны между собой классы (например, «содержит», «использует метод», «запрашивает данные», «наследует» и т.д.)

**Класс:** Выбор\_действия

Данные класса:

Запись\_CSV\_файла Запись\_чтение\_CSV — экземпляр класса Запись\_CSV\_файла для работы с CSV файлом.

List<Записанная\_книга> книги — список всех книг в библиотеке.

const string Путь\_к\_файлу — путь к файлу, в котором хранятся данные о книгах.

Методы класса:

void Добавить\_книгу() — добавляет новую книгу в список книг.

void Удалить\_книгу() — удаляет книгу из списка книг.

void Поиск\_по\_автору() — промежуточный метод между пользовательским вводом и логикой, реализованной в классе Поиск\_книг.

void Поиск\_по\_букве() — промежуточный метод между пользовательским вводом и логикой, реализованной в классе Поиск\_книг.

void Добавление\_записи() — промежуточный метод между пользовательским вводом и логикой, реализованной в классе Взятие\_или\_возврат\_книги.

void Возврат\_записи() — промежуточный метод между пользовательским вводом и логикой, реализованной в классе Взятие\_или\_возврат\_книги.

void Меню() — отображает меню выбора действий.

void Выполнить() — запускает цикл взаимодействия с пользователем.

Описание:

Этот класс представляет собой интерфейс пользователя. Он взаимодействует с пользователем, предлагая различные действия (например, добавить или удалить книгу), и вызывает соответствующие методы других классов для выполнения этих действий. Данный класс важен, так как он обеспечивает взаимодействие пользователя с системой и управляет основными процессами работы с библиотекой.

**Класс:** Книга

Данные класса:

string Название— название книги.

string Автор — имя автора книги.

string Издательство — издательство, выпустившее книгу.

int Год\_издания — год издания книги.

Методы класса:

string ToString() — переопределённый метод для представления объекта книги в виде строки.

**Описание:**

Класс Книга является основным классом в системе. Каждый объект этого класса содержит информацию о книге, такую как её название, автор, издательство, год издания. Этот класс необходим, так как он является родителем класса Записанная\_книга, используемого для всех операций с книгами в библиотеке.

**Класс:** Записанная\_книга

Данные класса:

string Записано\_за — ФИО взявшего книгу.

Методы класса:

string ToString() — переопределённый метод для представления объекта взятой книги в виде строки.

**Описание:**

Класс Записанная\_книга является наследником класса Книга. Каждый объект этого класса содержит информацию класса Книга и текущий статус (кто взял книгу). Этот класс необходим, так как он является основой для всех операций с книгами в библиотеке.

**Класс:** Запись\_CSV\_файла

Данные класса:

string Путь\_к\_файлу — путь к CSV файлу.

Методы класса:

void Добавить\_в\_CSV (Записанная\_книга книга, bool перезаписатьФайл) — метод для сохранения списка книг в CSV файл.

List<Записанная\_книга> Прочитать\_из\_CSV() — читает содержимое CSV-файла и преобразует его в список книг.

CSV\_файл представляет из себя таблицу Mc Excel и предназначен для долговременного хранения данных о книгах.

**Описание:**

Класс Запись\_CSV\_файла отвечает за взаимодействие с CSV-файлом, в котором хранятся данные о книгах. Он предоставляет методы для сохранения списка книг в файл, добавления новых записей и чтения данных из файла. Этот класс важен, так как он обеспечивает долгосрочное хранение данных о книгах и их восстановление при необходимости.

При чтении CSV\_файла информация, хранимая в нём может отображаться не в соответствии с кодировкой. Данная ошибка может возникать из-за индивидуальных настроек кодировки Mc Excel. Также на работу программы данная ошибка не влияет и не является критической.

Форма записи книги без указания взятия книги человеком:



Пример ошибки:



Форма записи книги с указанием взятия книги человеком:



**Класс:** Program

Методы класса:

void Main() — главный метод программы, который запускает приложение и вызывает метод Выполнить().

**Описание:**

Класс Program является точкой входа в программу. Он создаёт экземпляр класса Выбор\_действия и запускает его метод Выполнить(), который начинает взаимодействие с пользователем. Этот класс необходим, так как он инициализирует работу приложения и передаёт управление классу Выбор\_действия.

**Класс:** Взятие\_или\_возврат\_книги

Данные класса:

Запись\_CSV\_файла записьCSV — поле для работы с CSV-файлом через класс Запись\_CSV\_файла.

Взятие\_или\_возврат\_книги(Запись\_CSV\_файла записьCSV) — конструктор класса, который инициализирует объект класса, принимая экземпляр Запись\_CSV\_файла для работы с данными.

Методы класса:

void Добавление\_записи() — добавляет запись книги на читателя.

void Возврат\_записи() — возвращает взятую книгу.

**Класс:** Поиск\_книг

Данные класса:

List<Записанная\_книга> книги — поле, хранящее список книг для работы.

Поиск\_книг(List<Записанная\_книга> книги) — конструктор класса, который инициализирует объект класса, принимая список книг для последующего поиска.

Методы класса:

void Поиск\_по\_автору() — выполняет поиск книг по автору.

void Поиск\_по\_букве() — выполняет поиск книг по первой букве названия.

**Описание:**

Класс Взятие\_или\_возврат\_книги представляет собой функциональный модуль для поиска книг в библиотеке. Он взаимодействует с пользователем через консоль, запрашивая критерии поиска (автора или первую букву названия), выполняет фильтрацию данных и выводит результаты. Класс принимает список книг через конструктор, что позволяет ему работать с актуальными данными, полученными из CSV-файла или других источников. Он обеспечивает пользователю возможность быстро находить нужные книги, что делает работу с библиотекой удобной и эффективной.

**Критерии выделения классов:**

1. **Каждый класс имеет чётко определённую ответственность**:

* Класс Program отвечает за инициализацию приложения, запуск основного меню и управление жизненным циклом программы.
* Класс Книга отвечает за представление одной книги и хранение всей необходимой информации о ней (название, автор, год издания и т.д.).
* Класс Записанная\_книга отвечает за расширение функциональности базовой книги, включая отслеживание статуса (ФИО человека, взявшего книгу и время взятия/возврата).
* Класс Запись\_CSV\_файла отвечает за взаимодействие с CSV-файлом: чтение данных, запись новых или обновлённых записей, а также обеспечение целостности данных.
* Класс Выбор\_действия отвечает за взаимодействие с пользователем и выполнение основных операций с книгами, выступая связующим звеном между интерфейсом и логикой.
* Класс Взятие\_или\_возврат\_книги отвечает за управление статусом книг: отметку о выдаче (с записью ФИО и времени выдачи) и возврате (очистка статуса и отметка времени возврата), а также валидацию операций (например, проверку, не занята ли книга).
* Класс Поиск\_книг отвечает за реализацию алгоритмов поиска по заданным критериям (автор, первая буква названия) и вывод результатов пользователю.

1. **Избежание дублирования кода**

Разделение ответственности между классами позволяет избежать дублирования кода и улучшить структурированность программы.

1. **Легкость поддержки и расширения**

Чёткое разделение функционала на отдельные классы делает код более понятным и простым в поддержке и расширении.

**Связи между классами:**

**1. Наследование**

* Класс Записанная\_книга наследует базовые свойства (Название, Автор, Издательство, Год\_издания) от класса Книга и добавляет новое свойство Записано\_за для отслеживания статуса книги.

**2. Зависимости через конструкторы:**

* Класс Поиск\_книг принимает список объектов Записанная\_книга через конструктор, чтобы выполнять поиск по актуальным данным.
* Класс Взятие\_или\_возврат\_книги получает экземпляр Запись\_CSV\_файла через конструктор для чтения/записи данных.

**3. Агрегация и использование**

* Класс Выбор\_действия содержит экземпляры:

Запись\_CSV\_файла — для работы с CSV-файлом.

Поиск\_книг — для делегирования операций поиска.

Взятие\_или\_возврат\_книги — для управления статусом книг.

**4. Взаимодействие с данными**

* Класс Запись\_CSV\_файла читает данные из CSV-файла и преобразует их в объекты Записанная\_книга, а также сохраняет изменения этих объектов обратно в файл.

**5. Запуск приложения**

* Класс Program создаёт экземпляр Выбор\_действия и вызывает его метод Выполнить(), который управляет всем жизненным циклом приложения.

**Аргументация выбора типов полей в классах:**

1. Класс Книга

Название, Автор, Издательство - тип: string - эти свойства содержат текстовую информацию (название книги, имя автора, название издательства), для которой строковый тип (string) является естественным выбором.

Год\_издания - тип: int — целочисленное значение без дробной части. Использование int обеспечивает корректную арифметическую обработку.

2. Класс Записанная\_книга

Записано\_за - тип: string - свойство хранит ФИО человека, который взял книгу. Так как ФИО — это текст, используется тип string. Пустая строка ("") обозначает, что книга свободна.

3. Класс Запись\_CSV\_файла

Путь\_к\_файлу - тип: const string - константа содержит путь к файлу, который не изменяется в ходе работы программы. Использование const гарантирует, что значение останется фиксированным.

4. Класс Поиск\_книг

книги - тип: List<Записанная\_книга> - список (List<T>) позволяет эффективно выполнять операции поиска и фильтрации через LINQ (Where, Any).

5. Класс Взятие\_или\_возврат\_книги

записьCSV - тип: Запись\_CSV\_файла - поле хранит экземпляр класса для работы с CSV-файлом. Это обеспечивает инкапсуляцию логики чтения/записи данных.

6. Класс Выбор\_действия

книги - тип: List<Записанная\_книга> - список используется для временного хранения данных, полученных из CSV-файла, перед передачей в другие классы (например, Поиск\_книг).

Запись\_чтение\_CSV - тип: Запись\_CSV\_файла - экземпляр класса Запись\_CSV\_файла обеспечивает доступ к методам работы с файлом (чтение, добавление, обновление).

7. Класс Program

выбор\_действия - тип: Выбор\_действия - экземпляр класса Выбор\_действия используется для запуска основного функционала приложения через метод Выполнить().

**Порядок использования методов:**

1. **Запуск приложения** (Program.Main):

* Создаётся Выбор\_действия, вызывается Выполнить() для отображения меню.

1. **Взаимодействие с меню** (Выбор\_действия.Выполнить):

* Пользователь выбирает действие (1-7).

1. **Основные сценарии:**

* **Добавить книгу** (1 → Добавить\_книгу):

Запрашивает данные → Создает Записанная\_книга → Сохраняет в CSV.

* **Удалить книгу** (2 → Удалить\_книгу):

Ищет книгу в CSV → Удаляет строку.

* **Поиск по автору** (3 → Поиск\_по\_автору):

Читает CSV → Фильтрует книги → Выводит результат.

* **Поиск по букве** (4 → Поиск\_по\_букве):

Читает CSV → Фильтрует книги → Выводит результат.

* **Взять книгу** (5 → Добавление\_записи):

Проверяет статус книги → Записывает ФИО в CSV.

* **Вернуть книгу** (6 → Возврат\_записи):

Очищает поле Записано\_за → Обновляет CSV.

* **Выход** (7):

Завершает программу.

**Раздел 4. Описание алгоритмов программы**

Опишем метод добавления новой книги.

Шаги алгоритма:

1. **Начало**

* Начало алгоритма.

1. **Запрос данных у пользователя:**
   * Программа последовательно запрашивает название книги, автора, издательство и год издания.
   * Для года издания реализована проверка на корректность ввода (числовой формат) через цикл while с обработкой исключения FormatException.
2. **Создание объекта книги**
   * На основе введенных данных создается экземпляр класса Записанная\_книга с полями: Название, Автор, Год\_издания, Издательство.
3. **Сохранение данных:**
   * Книга добавляется в список книги.
   * Через метод Добавить\_в\_CSV экземпляра Запись\_чтение\_CSV данные сохраняются в CSV-файл (Библиотека.csv). Флаг true указывает на дополнение файла (а не перезапись).
4. **Конец**
   * Конец алгоритма.

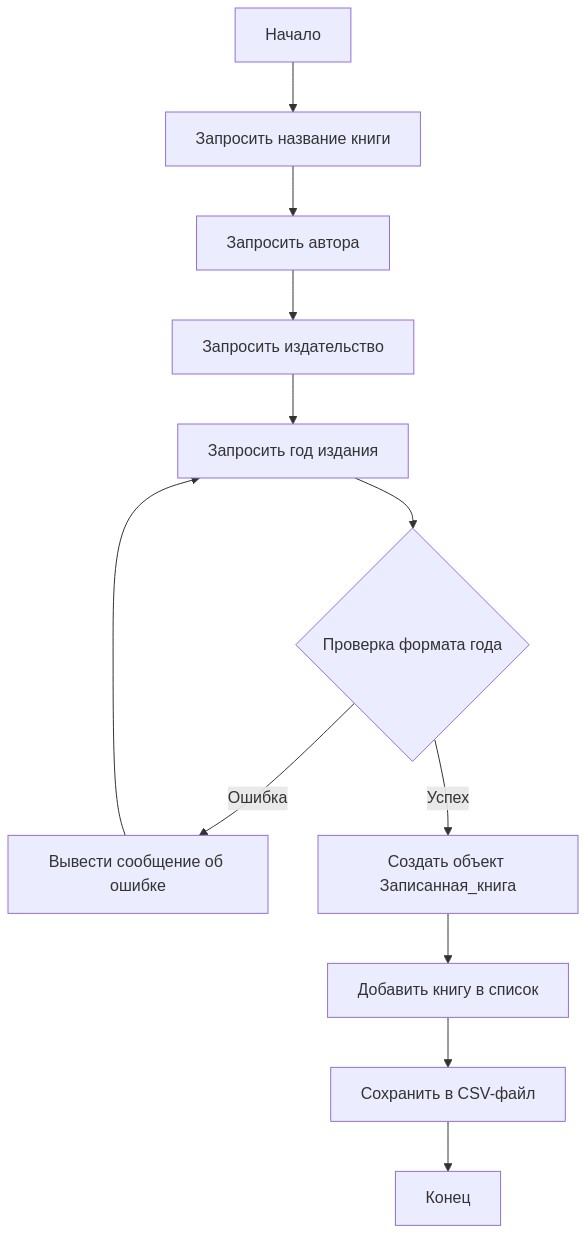


Рисунок 4 – Блок-схема для метода добавления новой книги

Давайте рассмотрим второй немаловажный метод: поиск книги по автору.

Шаги алгоритма:

1. **Начало**
   * Начало алгоритма.
2. **Запрос ввода от пользователя:**
   * Выводится сообщение: *"Введите автора для поиска: "*.
   * Пользователь вводит имя автора через консоль.
3. **Фильтрация списка книг:**
   * + Используется LINQ-запрос – Where.
     + **Условие фильтрации:**

Свойство Автор объекта Записанная\_книга должно совпадать с введённым значением.

Сравнение регистронезависимое (OrdinalIgnoreCase).

1. **Проверка результатов:**
   * Если найденные книги существуют (найденные\_книги.Any() == true):

Выводится заголовок: *"Найденные книги: "*.

Для каждой книги в списке вызывается Console.WriteLine(книга).

* + Если книги **не найдены**:

Выводится сообщение: *"Книги не найдены."*.

1. **Конец**
   * Конец алгоритма.

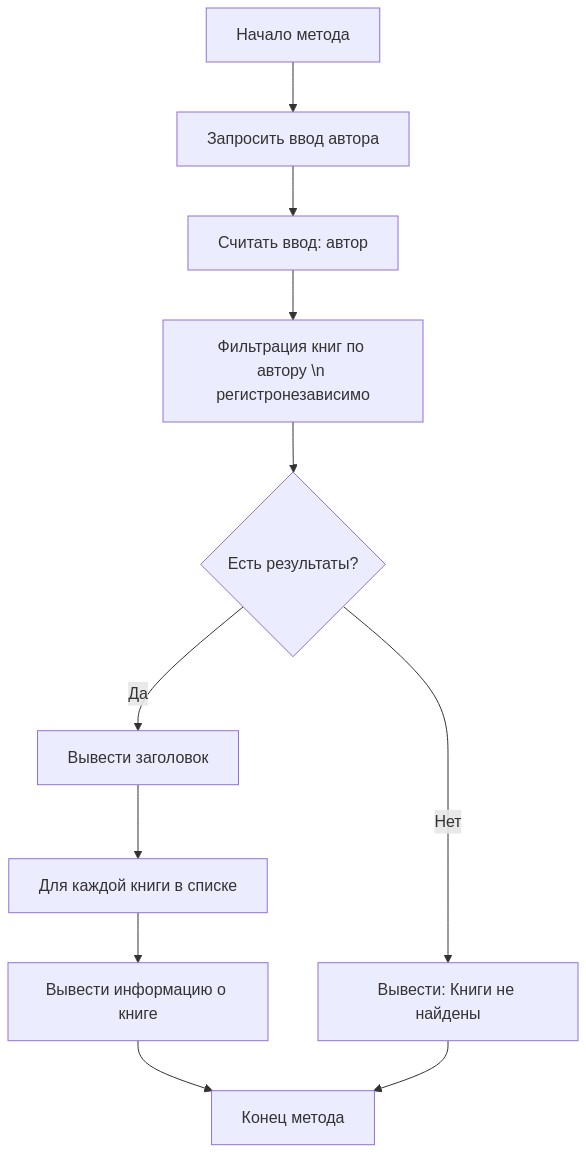


Рисунок 5 – Блок-схема для метода поиска книг по автору

# **Раздел 5.** Разработка проекта Visual Studio C#

Рисунок 6 - Точка входа в программу



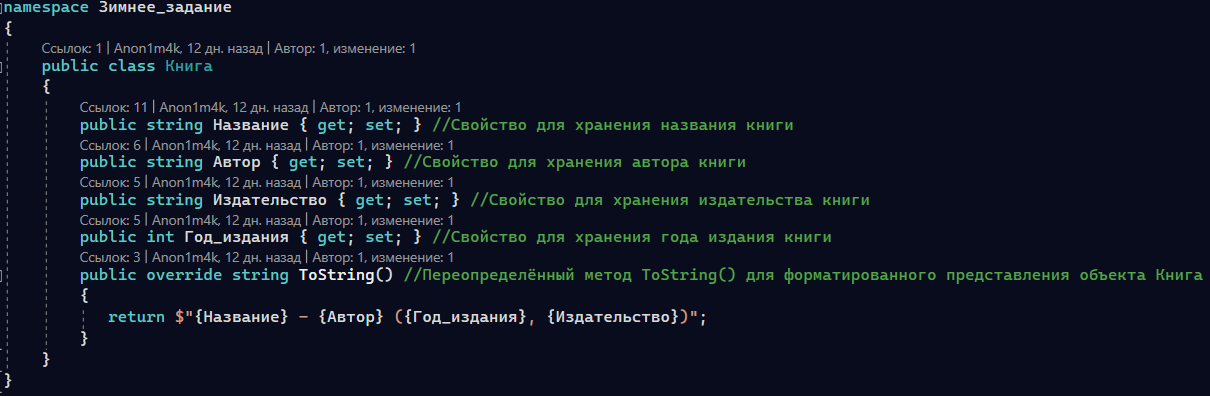
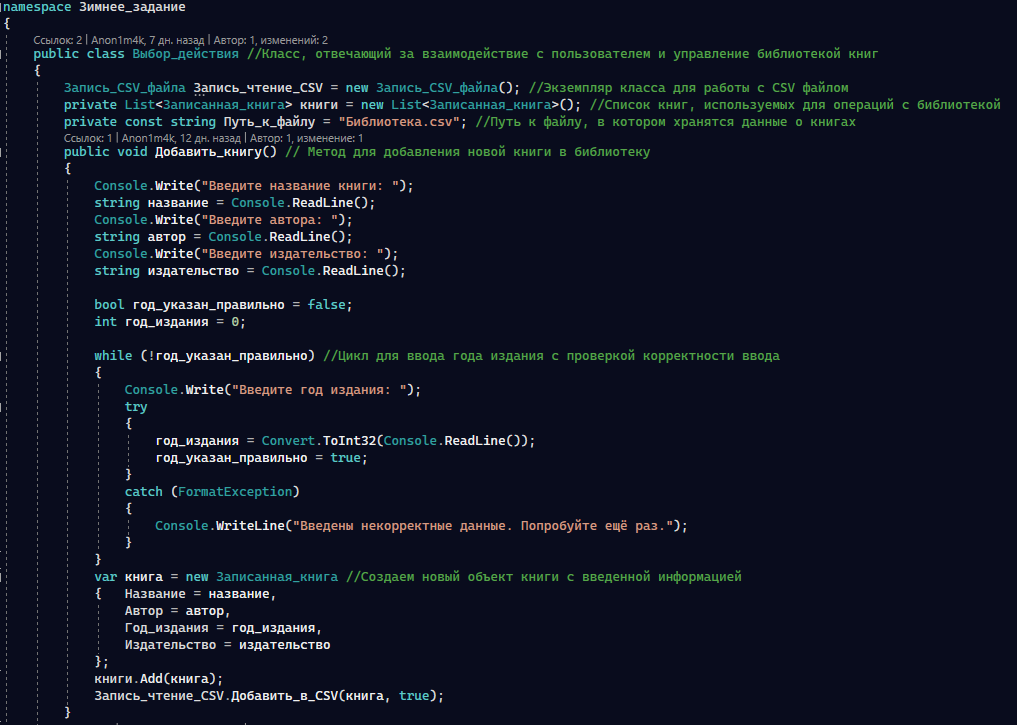
Рисунок 8 – Методы для записи и возврата книги

Рисунок 9 - Определение класса "Книга"

Рисунок 10 – Метод добавления новой книги

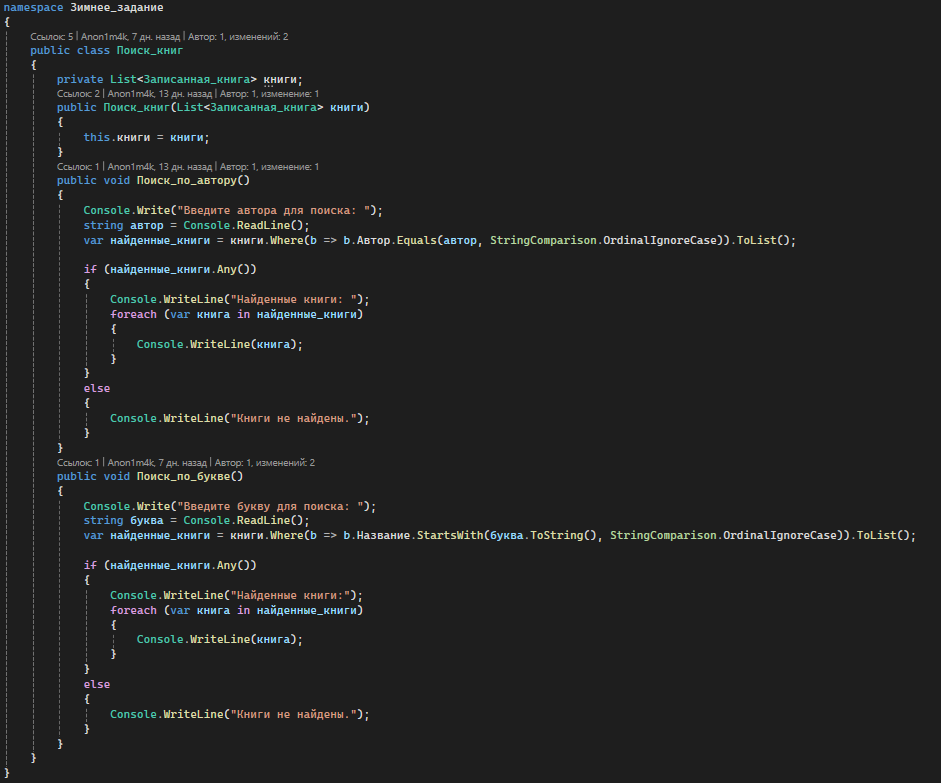


Рисунок 11 – Методы для поиска книг по букве и по автору

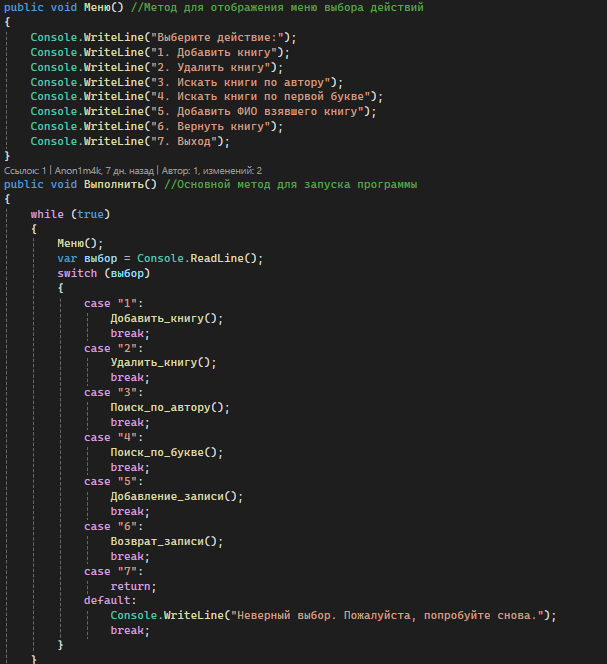
****

Рисунок 12 - Метод меню для выбора действия

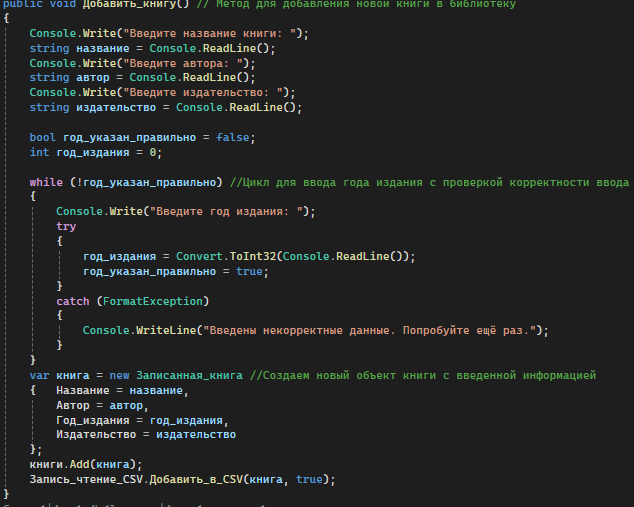


Рисунок 13 - Метод добавления новой книги

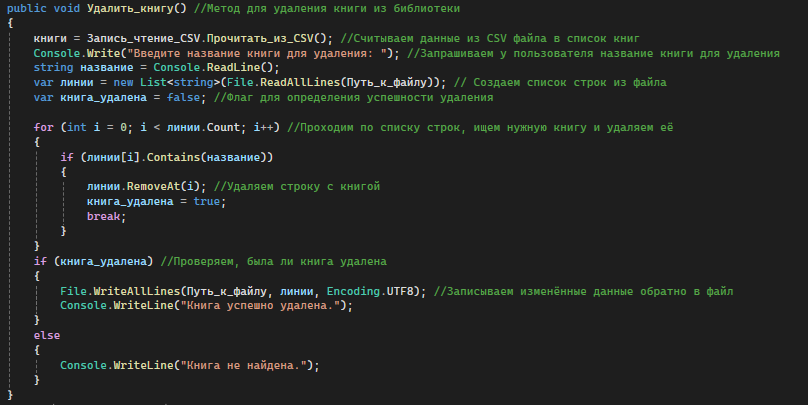


Рисунок 14 – Метод для удаления книги

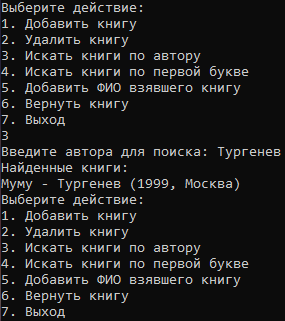


Рисунок 1 – Пример поиска книги по автору

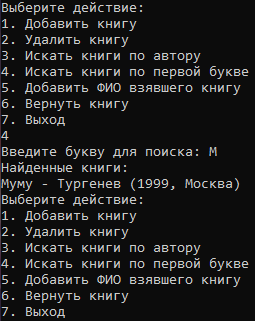


Рисунок 2 – Пример поиска книги по первой букве