Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

Дисциплина: Программированиена языках высокого уровня

К ЗАЩИТЕ ДОПУСТИТЬ

\_\_\_\_\_\_\_\_\_\_\_ А. М. Ковальчук

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту

на тему

«СВЕДЕНИЯ ОБ ИГРОКАХ ФУТБОЛЬНОЙ КОМАНДЫ»

БГУИР КП 1-40 02 01 110 ПЗ

Студент: Феденко Р.Ю.

Руководитель: Ковальчук А.М.

Минск 2022

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

УТВЕРЖДАЮ

Заведующий кафедрой

*.*

(подпись)

2022 г.

ЗАДАНИЕ

по курсовому проектированию

Студенту *Феденко Роману Юрьевичу*

1. Тема проекта «*Сведения об игроках футбольной команды*»

2. Срок сдачи студентом законченного проекта-*28 декабря 2020г.*

3. Исходные данные к проекту *теккстовые файлы, в которых содержится информация: Пользователь (ID, фамилия, имя, отчество), Игроки (ID, ФИО, дата рождения,число забитых мячей,количество голевых передач), Директор (ID Пользователя, ФИО Пользователя, ID игрока, ФИО игрока, дата рождения, число забитых мячей,количество голевых передач); среда разработки Visual Studio 2022 (кросс-платформенная).*

4. Содержание расчетно-пояснительной записки (перечень вопросов, которые подлежат разработке)

*Введение. 1. Обзор литературы. 2. Структурное проектирование.*

*3. Функциональное проектирование. 4. Проектирование программных модулей 5. Руководство пользователя. 6. Список литературы. Заключение*

5. Перечень графического материала (с точным обозначением обязательных чертежей и графиков)

*1. Схема структурная*

*2. Схема алгоритма авторизации пользователя*

*3. Схема алгоритма регистрации пользователя*

# *4. Диаграмма классов*

*5. Скриншоты работающей программы*

6. Консультант по проекту (с обозначением разделов проекта)   *Ковальчук А.М.*

7. Дата выдачи задания  *14 сентября 2020 г.––––––*

8. Календарный график работы над проектом на весь период проектирования (с обозначением сроков выполнения и трудоемкости отдельных этапов):

*разделы 1,2 к 05.10.  –  10 %; –*

*раздел 3   к 25.10.  –  15 %*

*раздел 4 к 07.11. – 15 %; –*

*раздел 5 к 19.11. – 25 %; –*

*раздел 6 к 25.11.  –  25 %; ––*

*оформление пояснительной записки и графического материала к 1.12.2020 – 10 %*

*Защита курсового проекта 28.12*

*––––––––––––––*

РУКОВОДИТЕЛЬ *Ковальчук А.М.*

(подпись)

Задание принял к исполнению *Р. Ю. Феденко*

(дата и подпись студента)

**Оглавление**

[1.ОБЗОР ЛИТЕРАТУРЫ 7](#_Toc59564794)

[2.СТРУКТУРНОЕ ПРОЕКТИРОВАНИЕ 9](#_Toc59564795)

[2.1 Блок приложения 9](#_Toc59564796)

[2.1.1 Блок регистрации и авторизации пользователя 10](#_Toc59564797)

[2.1.2 Блок добавления и удаления книг 10](#_Toc59564798)

[2.1.3 Блок вывода информации 10](#_Toc59564799)

[2.1.4 Локальная база данных 10](#_Toc59564802)

[3.ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВАНИЕ 11](#_Toc59564803)

[3.1 Текстовой файл 11](#_Toc59564804)

[3.1.1 Файл BooksList.txt 12](#_Toc59564805)

[3.1.2 Файл LibraryBook.txt 12](#_Toc59564806)

[3.1.3 Файл PasswordLibrian.txt 12](#_Toc59564807)

[3.1.4 Файл StudentsList.txt 13](#_Toc59564808)

[3.1.5 Класс Library 13](#_Toc59564809)

[4.ПРОЕКТИРОВАНИЕ ПРОГРАММНЫХ МОДУЛЕЙ 14](#_Toc59564810)

[5.РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ 16](#_Toc59564811)

[6.СПИСОК ЛИТЕРАТУРЫ 19](#_Toc59564812)

[7.ЗАКЛЮЧЕНИЕ 20](#_Toc59564813)

[Приложение А 21](#_Toc59564814)

[Приложение Б 22](#_Toc59564815)

[Приложение В 23](#_Toc59564816)

[Приложение Г 24](#_Toc59564817)

[Приложение Д 25](#_Toc59564818)

**ВВЕДЕНИЕ**

В данном курсовом проекте предполагается реализовать программное обеспечение по управлению библиотекой при помощи инструментальной среды программного обеспечения. Для реализации этой задачи я использовал язык программирования С++ который был разработан в начале 1980-х годах сотрудником фирмы Bell Labs Бьёрном Страуструпом.

C++ — это язык с C-подобным синтаксисом.

Страуструп решил дополнить язык C возможностями, имеющимися в языке Симула. Из этого языка были позаимствованы возможности объектно-оринтированного программирования (возможность работы с классами и объектами). К 1983 году в язык были добавлены новые возможности, такие как виртуальные функции, перегрузка функций и операторов, ссылки, константы, пользовательский контроль над управлением свободной памятью, улучшенная проверка типов и новый стиль комментариев (*//*). Получившийся язык уже перестал быть просто дополненной версией классического C (*“*C с классами*“*) и был переименован в «C++». Его первый коммерческий выпуск состоялся в октябре 1985 года.

C++ уже давно поддерживает много полезных функций:

* инкапсуляция,
* наследование,
* полиморфизм,
* перегрузка операторов,
* статическая типизация.

При этом он всё ещё активно развивается, и с каждой новой версией появляется всё больше интересного — например лямбды, динамическое связывание, асинхронные методы и т.д.

Когда говорят C++, нередко имеют в виду технологии платформы .NET. И наоборот, когда говорят .NET, нередко имеют в виду C++. Однако, хотя эти понятия связаны, отождествлять их неверно. Язык C++ был создан специально для работы с фреймворком .NET, однако само понятие .NET несколько шире.

Когда-то Билл Гейтс сказал, что .NET — это лучшее, что сделала компания Microsoft. У него есть весомые основания так считать. Фреймворк .NET представляет мощную платформу для создания приложений.

Вот его киллер-фичи:

* Поддержка нескольких языков. В основе  .NET — общеязыковая среда исполнения Common Language Runtime (CLR), благодаря чему платформа поддерживает несколько языков: наряду с C# это VB.NET, C++, F#, а также различные диалекты других языков, привязанные к .NET, например, Delphi.NET. Код на любом из этих языков компилируется в сборку на общем языке CIL (Common Intermediate Language) — своего рода ассемблер платформы .NET. Поэтому можно сделать отдельные модули одного приложения на разных языках.
* Мощная библиотека классов. .NET представляет единую для всех поддерживаемых языков библиотеку классов. Какое бы приложение мы ни собирались писать на C++ — текстовый редактор, чат или сложный веб-сайт — так или иначе мы задействуем библиотеку классов .NET.
* Разнообразие технологий. Общеязыковая среда исполнения CLR и базовая библиотека классов — это основа для целого стека технологий, которые разработчики могут задействовать при создании разных приложений. Например, для баз данных в этом стеке имеется технология ADO.NET и Entity Framework Core. .NET долгое время развивался под названием .NET Framework — преимущественно как платформа для Windows. Но с 2019 она больше не развивается — последней версией этой платформы стала .NET Framework 4.8.

В 2014 Microsoft начал выпускать альтернативную платформу - .NET Core, которая должна была вобрать в себя все возможности устаревшего .NET Framework и добавить новую функциональность. Поэтому следует различать .NET Framework, который предназначен преимущественно для Windows, и кроссплатформенный .NET Core.

Существует множество способов для написания графических приложений Windows Forms, Qt Creator и т.д. В данном курсовом проекте используется одна из самых популярных IDE – WPF.

# **1. ОБЗОР ЛИТЕРАТУРЫ**

Ниже представлены основные термины в программировании, так как без понимания их невозможно будет выучить ни один из языков программирования, а также проектирование ПО:

**Программирование** – [процесс](https://ru.wikibooks.org/w/index.php?title=%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81&action=edit&redlink=1) и [искусство](https://ru.wikibooks.org/w/index.php?title=%D0%98%D1%81%D0%BA%D1%83%D1%81%D1%81%D1%82%D0%B2%D0%BE&action=edit&redlink=1) создания компьютерных программ с помощью [языков программирования](https://ru.wikibooks.org/w/index.php?title=%D0%AF%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F&action=edit&redlink=1). В узком смысле слова, программирование рассматривается как кодирование – реализация одного или нескольких взаимосвязанных алгоритмов на некотором языке программирования. В более широком смысле, программирование – процесс создания программ, то есть разработка программного обеспечения. Большая часть работы программиста связана с написанием исходного кода на одном из языков программирования.

**Программа**– данные, предназначенные для управления конкретными компонентами системы обработки информации в целях реализации определенного алгоритма.

**Программное обеспечение**– совокупность программ системы обработки информации и программных документов, необходимых для эксплуатации этих программ.

**Разработка программного обеспечения** – это проектирование, написание, тестирование и поддержка компьютерных программ с целью решения задач для множества пользователей; это создание надежных защищенных решений, которые выдержат испытание временем и справятся с некоторыми не известными заранее задачами, лежащими в области, близкой к очевидным исходным задачам.

**Программный модуль** – программа или функционально завершенный фрагмент программы, предназначенный для хранения, трансляции, объединения с другими программными модулями для загрузки в оперативную память.

**Объектно-ориентированное программирование** – метод построения программ как совокупностей объектов и классов объектов, которые могут вызывать друг друга.

**Компиляция** – трансляция программы с языка высокого уровня в форму, близкую к программе на машинном языке.

**Абстракция** — мысленное отвлечение, обособление от тех или иных сторон, свойств или связей предметов и явлений для выявления существенных их признаков.

**Диаграмма классов** структурная [диаграмма](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%B0%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0_(UML)) языка моделирования [UML](https://ru.wikipedia.org/wiki/UML), демонстрирующая общую структуру иерархии [классов](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81%D1%81_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) системы, их коопераций, [атрибутов](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D0%B5_%D0%BA%D0%BB%D0%B0%D1%81%D1%81%D0%B0) (полей), [методов](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_(%D1%8F%D0%B7%D1%8B%D0%BA%D0%B8_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F)), интерфейсов и взаимосвязей между ними.

Целью создания диаграммы классов является графическое представление статической структуры декларативных элементов системы (классов, [типов](https://ru.wikipedia.org/wiki/%D0%A2%D0%B8%D0%BF_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) и т. п.) Она содержит в себе также некоторые элементы поведения (например — операции), однако их динамика должна быть отражена на диаграммах других видов ([диаграммах коммуникации](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%B0%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0_%D0%BA%D0%BE%D0%BC%D0%BC%D1%83%D0%BD%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D0%B8), диаграммах состояний). Для удобства восприятия диаграмму классов можно также дополнить представлением [пакетов](https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D0%BA%D0%B5%D1%82_(UML)), включая вложенные[[2]](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%B0%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0_%D0%BA%D0%BB%D0%B0%D1%81%D1%81%D0%BE%D0%B2#cite_note-_00071d72222e96c1-2).

При представлении сущностей реального мира разработчику требуется отразить их текущее состояние, их поведение и их взаимные отношения. На каждом этапе осуществляется абстрагирование от маловажных деталей и концепций, которые не относятся к реальности (производительность, [инкапсуляция](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D0%BA%D0%B0%D0%BF%D1%81%D1%83%D0%BB%D1%8F%D1%86%D0%B8%D1%8F_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), [видимость](https://ru.wikipedia.org/wiki/%D0%92%D0%B8%D0%B4%D0%B8%D0%BC%D0%BE%D1%81%D1%82%D1%8C_(UML)) и т. п.). Классы можно рассматривать с позиции различных уровней. Как правило, их выделяют три основных: аналитический уровень, уровень проектирования и уровень реализации:

* на уровне анализа класс содержит в себе только набросок общих контуров системы и работает как логическая концепция предметной области или программного продукта.
* на уровне проектирования класс отражает основные проектные решения касательно распределения информации и планируемой функциональности, объединяя в себе сведения о состоянии и операциях.
* на уровне реализации класс дорабатывается до такого вида, в каком он максимально удобен для воплощения в выбранной среде разработки; при этом не воспрещается опустить в нём те общие свойства, которые не применяются на выбранном языке программирования.
  1. **Постановка задачи**

Темой данного курсового проекта является «Сведения об игроках футбольной команды».

Данная сиcтема содержит программное обеспечение для персонального компьютера.

*Цели* курсового проектирования: овладеть практическими навыками проектирования и разработки законченного, отлаженного и протестированного программного продукта с использованием языка высокого уровня С++;

Минимальные требования к реализации программного продукта:

* хранение пользовательских данных
* запись, удаление и изменение информации
* простой и понятный интерфейс.

Для достижения поставленной цели необходимо решить следующие задачи:

*Задача* курсового проектирования – проектирование прикладного программного обеспечения.

# **2. СТРУКТУРНОЕ ПРОЕКТИРОВАНИЕ**

Основной функцией компьютера является обработка информации.Программная обработка данных на компьютере реализуется следующим образом:

1. После запуска на выполнение программы, хранящейся во внешней долговременной памяти, она загружается в оперативную память.

2. Процессор последовательно считывает команды программы и выполняет их.

3. Необходимые для выполнения команды данные загружаются из внешней памяти в оперативную и над ними производятся необходимые операции. Данные, полученные в процессе выполнения команды, записываются процессором обратно в оперативную или внешнюю память.

4. В процессе выполнения программы процессор может запрашивать данные с устройств ввода информации и пересылать данные на устройства вывода информации.

Программная конфигурация ПК многоуровневая. Это связано с тем, что требования к программам, предназначенным для работы с устройствами, существенно отличаются от требований к программам, предназначенным для работы с людьми. Общий принцип такой: чем ниже уровень программ, тем больше они работают с устройствами и меньше с человеком. Этот принцип соблюдается во всей компьютерной технике от отдельного ПК до всемирной компьютерной сети Интернет.

В понятие структуры программы включается состав и описание связей всех модулей, которые реализуют самостоятельные функции программы и описание носителей вводимых и выводимых данных, а также данных, участвующих в обмене между отдельными подпрограммами.

Для разработки больших и сложных программ программисту необходимо овладеть специальными приемами получения рациональной структуры программы, которая обеспечивает почти двукратное сокращение объема программирования. Подчиненность модулей программы отражается в схеме, представленной в приложении А. Однако последняя не отражает порядок их вызова или функционирование программы.

# **2.1 Блок приложения**

В приложении можно выделить следующие структурные блоки:

* блок регистрации и авторизации пользователя;
* блок добавления и удаления данных;
* блок вывода информации;
* локальная база данных.

# **2.1.1 Блок регистрации и авторизации пользователя**

Данный блок предоставляет пользователю возможность зарегистрироваться и авторизоваться в программе. Блок имеет удобный пользовательский интерфейс, в котором находится два метода. В первоначальном окне пользователь может выбрать кем он является: пользователем или директором. Пользователь может авторизоваться под своим ID или зарегистрироваться введя свои данные, при этом ему выдадут свой ID.

# **2.1.2 Блок добавления и удаления сведений**

Данный блок предоставляет пользователь, а именно директора, возможность добавления сведений в локальную базу данных в виде текстового документа, а также удалить эти данные.

# **2.1.3 Блок вывода информации**

Данный блок предоставляет пользователю возможность вывести информацию из локальной базы данных. В зависимости от своего статуса он может выводить разную информацию. Пользователь может вывести информацию о сведениях, а директор может вывести список зарегистрированных пользователей и также сведения.

# **2.1.4 Локальная база данных**

Локальная база данных используется для сохранения ранее скачанных данных для предоставления пользователю возможности просматривать ранее загруженную информацию оффлайн без подключения к сети Интернет. В качестве локального хранилища данных будет использован метод хранения информации в текстовом редакторе.

# **3. ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВАНИЕ**

# **3.1 Текстовой файл**

**Текстовой файл** — это файл, компонентами которого являются символьные строки переменной длины, заканчивающиеся специальным маркером конца строки.

Преимущества:

* Минимальный объём файла (при малом количестве текстовых данных).
* Универсальность — текстовый файл может быть прочитан (так или иначе) на любой системе или ОС, особенно если речь идёт об однобайтных кодировках вроде ASCII — они не подвержены многим проблемам, характерным для других форматов файлов — для них не важна разница в порядке байтов или длине машинного слова на разных платформах.
* Устойчивость — каждое слово и символ в таком файле самодостаточны, и если случится повреждение такого файла, то в этом случае обычно легче восстановить данные и продолжить обработку остального содержимого — повреждение любого из байтов такого файла никак не сказывается на сохранность остальных, в отличие от или сжатых или двоичных (напр. \*.doc). Многие распространённые системы управления версиями (например в MS Word) рассчитаны на текстовые файлы, и с двоичными файлами могут работать только как с единым целым — при повреждении даже нескольких байтов такой файл восстановлению не подлежит.

Для работы с файлами я использовал классы ofstream и ifstream.

В таблице 1.1 представлены команды используемые в приложении для работы с файлами.

Таблица 1.1 – Используемые методы DbConect и DbSet

| Тип метода | Метод | Описание метода |
| --- | --- | --- |
| Методы доступа к файлу | open() | Этим методом можно открыть заданный файл, сопоставив его с одним из объектов потока. В зависимости от передаваемых аргументов, файл может быть открыт для чтения, для записи (либо для полной, либо для добавления данных), как бинарный, или как текстовый файл. |
|  | close() | Функция закрывает файл, то есть прекращает доступ к нему, таким образом освобождая его для других функций или программ. |

# 

# **3.1.1 Файл BooksList.txt**

В файле BooksList.txt находится основная информация о книгах, использующих приложение. Более подробно рассмотрим поля, который хранятся в данном файле:

* поле id. Уникальный номер киниг, является первичным ключом файла;
* поле title. Содержит название книги;
* поле writer. Содержит данные о авторе книги;
* поле year. Содержит год выпуска книги;

# **3.1.2 Файл LibraryBook.txt**

В файле LibraryBook.txt находится основная информация о студентах которые взяли какие либо книги. Более подробно рассмотрим поля, который хранятся в данном файле:

* поле id\_student. Уникальный номер студента;
* поле surname. Содержит фамилию студента;
* поле name. Содержит имя студента;
* поле patrinymic. Содержит отчество студента;
* поле grupa. Содержит номер группы студента;
* поле id\_book. Уникальный номер киниг, является первичным ключом файла;
* поле title. Содержит название книги;
* поле writer. Содержит данные о авторе книги;
* поле year. Содержит год выпуска книги;

# **3.1.3 Файл PasswordLibrian.txt**

В файле PasswordLibrian.txt находится информация о пароле библиотекаря.

# **3.1.4 Файл StudentsList.txt**

Файл StudentsList.txt находится основная информация о студентах которые взяли какие либо книги. Более подробно рассмотрим поля, который хранятся в данном файле:

* поле id. Уникальный номер студента;
* поле surname. Содержит фамилию студента;
* поле name. Содержит имя студента;
* поле patrinymic. Содержит отчество студента;
* поле grupa. Содержит номер группы студента;

# **3.1.5 Класс Library**

Класс Library используется работы приложения, это основной класс, в котором находятся все данные и методы для работы с приложением.

Класс имеет методы:

* void Start() , производит старт программы.
* void StudentMain(int id) , производит старт студенческого окна.
* void LibrarianMain(), производит старт окна для библиотекаря .
* void Student() , производит запуск авторизации или регистрации студента.
* void Librarian(), производит авторизацию библиотекаря.
* void AddBook(), отвечает за добавления данных о новых книгах.
* void BookList(int n, int id\_student), отвечает вывод данных о имеющихся книгах.
* void Delete(),производит удаления книги из файла.
* int AmountOfData(string filename), производит выдачу информации о количестве данных в фале.
* void LibrarianStudentList(),производит вывод информации о зарегистрированных студентах.
* void LibrarianStudentSearch(),производит поиск студента по его уникальному номеру.
* void SearchBook(int n, int id\_student), производит поиск книги по его уникальному номеру.

# **4.ПРОЕКТИРОВАНИЕ ПРОГРАММНЫХ МОДУЛЕЙ**

Программа начинает свою работу с созданием объекта класса и вызова метода запуска главного меню программы.

Library obj;

obj.Start();

Во всем коде программы встречаются методы, с помощью которых производится работа с локальной базой данных.

Library lubf;

ifstream reading("StudentsList.txt");

string buf; int buf\_year, buf\_grupa;

int size;

if (reading)

{

reading >> size;

for (int i = 0; i < size; i++)

{

reading >> lubf.student.id;

reading >> buf\_grupa;

reading >> buf;

reading >> buf;

reading >> buf;

if (id == lubf.student.id)

{

system("cls");

reading.close();

StudentMain(id);

}

}

}

Выше показан код при котором производится поиск студента по его уникальному номеру.

Так же в программе реализован метод удаление данных:

void Library::Delete()

{

Copy("BooksList.txt",2);

ifstream reading("Buffer.txt");

ofstream record("BooksList.txt", ios::out);

int size, new\_size, n;

if (reading) {

if (record) {

reading >> size;

new\_size = size - 1;

cout << "\n\t\tВыберите номер: ";

cin >> n;

n--;

record << new\_size << endl;

for (int i = 0; i < n; i++) {

if (i != n) {

reading >> book.id;

record << book.id << endl;

reading >> book.title;

record << book.title << endl;

reading >> book.writer;

record << book.writer << endl;

reading >> book.year;

if (i < n - 1) record << book.year << endl;

else record << book.year;

}

else {

reading >> book.id;

reading >> book.title;

reading >> book.writer;

reading >> book.year;

}

}

}

system("cls");

record.close();

reading.close();

remove("Buffer.txt");

\_getch();

system("cls");

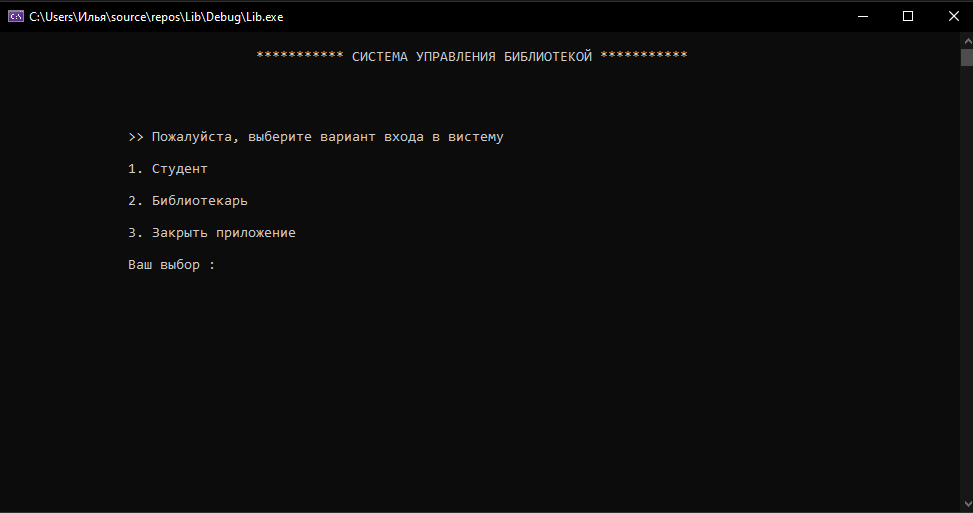
LibrarianMain();

}

}

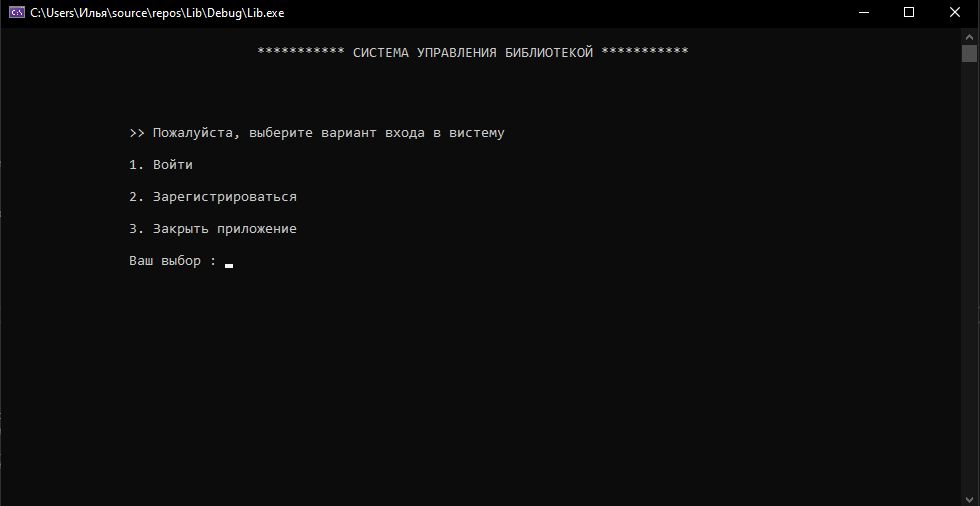
# **5. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ**

При запуске программы открывается окно



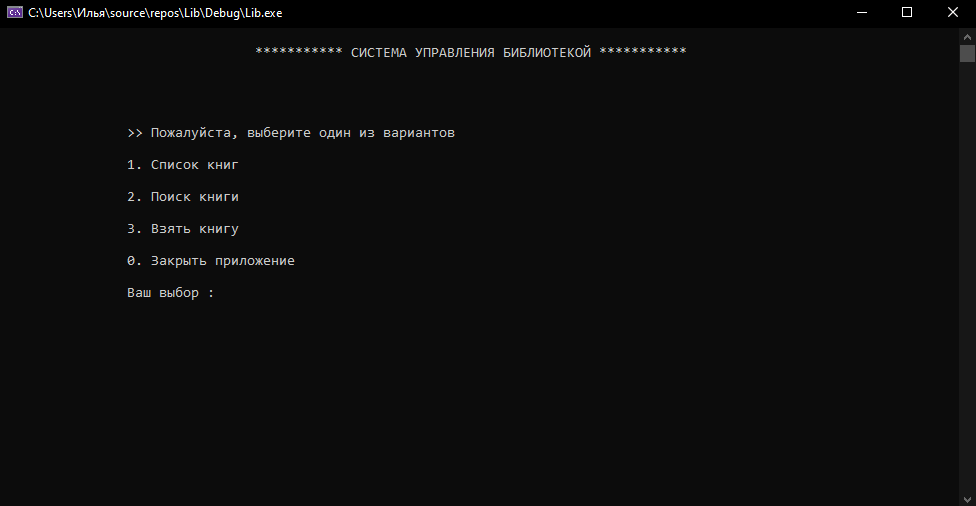
Где мы наблюдаем 3 выбора, пользователь может зайти как студент, так и как библиотекарь. Также пользователь может закрыть программу

При выборе номера “1” открывается окно показанный ниже на рисунке.

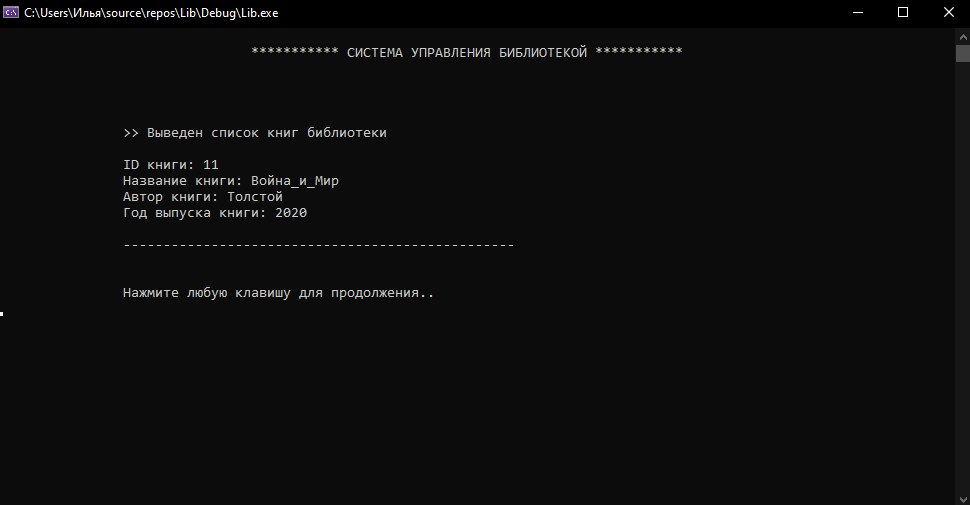


Где студент может как авторизоваться, так и зарегистрироваться.

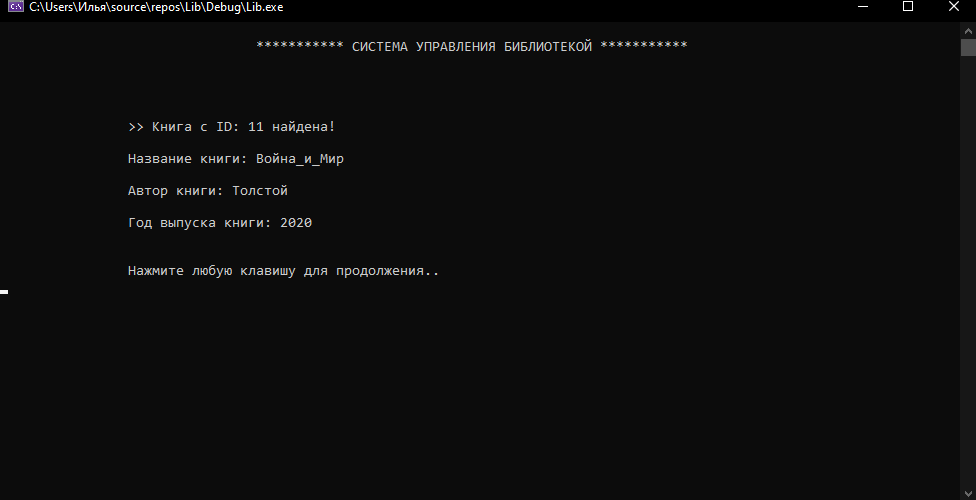
При успешной регистрации или авторизации мы можем наблюдать студенческое окно, показанный ниже на рисунке.



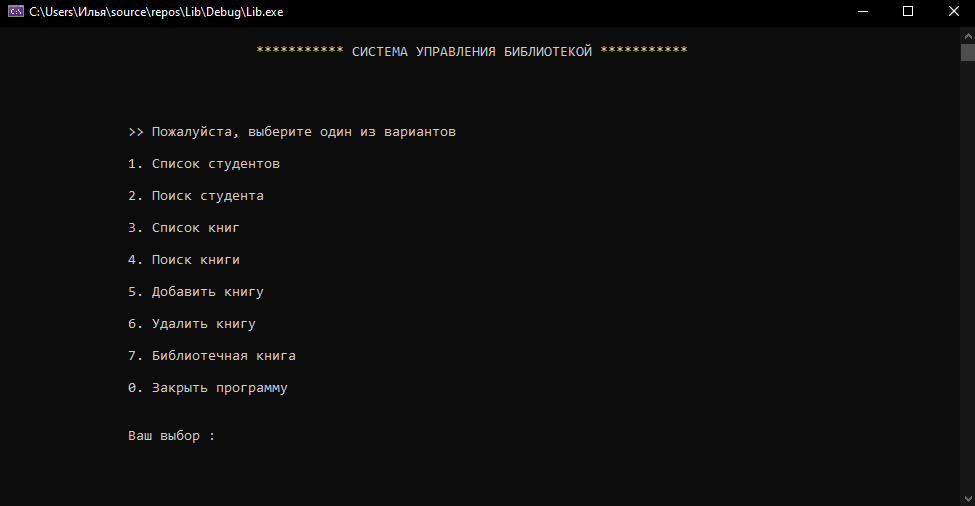
В данном окне мы можем выбрать один из 4 методов. При выборе методе “1” мы можем увидеть список книг (рисунок ниже).



При выборе методе “2” мы можем найти книгу по его уникальному номеру (рисунок ниже).



Теперь рассмотрим главное окно при успешной авторизации библиотекаря. Рисунок главного окна библиотекаря предоставлен ниже.



В данном окне мы можем вывести список книг и студентов, так же мы можем вывести список студентов взявший книгу. Мы можем искать студента или книгу по их уникальным номерам, а также у нас появляется возможность добавить или удалить книгу из локальной базы данных.

# **6. СПИСОК ЛИТЕРАТУРЫ**

[1].Язык программирования С++ [Электронный ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki>/C++. – Дата доступа 19.9.2019.

[2].Языки программирования [Электронный ресурс]. – Режим доступа: <http://sd-company.su/article/help_computers/programming_language> – Дата доступа 5.10.2019.

[3].Технологии программирования [Электронный ресурс]. – Режим доступа: [http://info-master.su/programming/profi/programming-technologies.php – Дата доступа 7.10.2019](http://info-master.su/programming/profi/programming-technologies.php%20–%20Дата%20доступа%207.10.2019).

[4].Структура программного обеспечения ПК [Электронный ресурс]. – Режим доступа: [https://studopedia.ru/4\_4312\_tema--struktura-programmnogo-obespecheniya-pk.html – Дата доступа 20.10.2019](https://studopedia.ru/4_4312_tema--struktura-programmnogo-obespecheniya-pk.html%20–%20Дата%20доступа%2020.10.2019).

[5].Руководство по разработке на компонентах [Электронный ресурс]. – Режим доступа: https://habr.com/ru/company/ruvds/blog/461661/ – Дата доступа 11.10.2019.

[6].Теоретические основы алгоритмизации и программирования [Электронный ресурс]. – Режим доступа: <https://studfile.net/preview/4693947/> – Дата доступа 19.10.2019.

[7].Луцик, Ю. А. Объектно-ориентированное программирование на языке С++: учеб. пособие по курсу «Объектно-ориентированное программирование» для студ. спец. «Вычислительные машины, системы и сети» всех форм обуч. / Ю. А. Луцик, А. М. Ковальчук, И. В. Лукьянова. – Минск: БГУИР, 2003. – 203 с.:ил.

# **ЗАКЛЮЧЕНИЕ**

В результате выполнения курсового проекта была разработано программное средство по управлению библиотекой. Во время реализации данного проекта были использован язык программирования C++. При разработке программы стало понятно, что классы и объекты в С++ являются основными концепциями объектно-ориентированного программирования — ООП. А  объектно-ориентированное программирование — расширение структурного программирования, в котором основными концепциями являются понятия классов и объектов. Стало ясно что основное отличие языка программирования С++ от С состоит в том, что в С нет классов, а следовательно язык С не поддерживает ООП, в отличие от С++.

В программе удалось реализовать удобный пользовательский интерфейс, хороший дизайн. Программа связана с локальной базой данных в виде файлов. Программа проста в использовании и не имеет большого функционала, но в ней реализованы самые нужные функции при работе с библиотекой.

**Приложение А**

*(обязательное)*

Схема структурная

**Приложение Б**

*(обязательное)*

Схема алгоритма

**Приложение В**

*(обязательное)*

Схема алгоритма

**Приложение Г**

*(обязательное)*

Код программы

**Приложение Д**

*(обязательное)*

Диаграмма классов