

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение   
высшего образования

**«Дальневосточный федеральный университет»**

(ДВФУ)

**ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ**

**Департамент программной инженерии и искусственного интеллекта**

ФЕНЬ ЕВГЕНИЯ ВАДИМОВНА

СОРТИРОВЩИК КНИГ

**КУРСОВОЙ ПРОЕКТ**

по дисциплине «Фундаментальные структуры данных и алгоритмы»

по образовательной программе подготовки бакалавров по направлению

09.03.04 - Программная инженерия

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | |  |  | Студентка гр. Б9120-09.03.04прогин | | |
|  |  |  | | Е.В.Фень |
|  | | | | | | |  |  | (подпись) | |  |
|  | | | | | | | | | | | |
| Защищен с оценкой | | | | | | |  |  | Руководитель | | |
|  | | | | | | |  |  | ст. преподаватель | | |
|  | | | |  |  | |  |  | ученая степень, должность |  | О.А. Крестникова |
| (подпись) | | | |  | (И.О. Фамилия) | |  |  | (подпись) |  | (И.О. Фамилия) |
| « |  | » |  | | | 2022 г. |  |  |  | | |
|  | | | | | | | | | | | |

г. Владивосток

2022

# Оглавление

[Оглавление 2](#_Toc115958907)

[Введение 3](#_Toc115958908)

[1 Анализ предметной области 4](#_Toc115958909)

[1.1 Объекты предметной области 4](#_Toc115958910)

[1.2 Законы ПО 5](#_Toc115958911)

[1.3 Постановка задачи «Формирование отчета «Сортировщик книг»» 6](#_Toc115958912)

[2 Теоретическая часть 7](#_Toc115958913)

[2.1 Хеш-таблица 7](#_Toc115958914)

[2.1.1 Хеш-функция 8](#_Toc115958915)

[2.1.2 Разрешение коллизий методом открытой адресации 9](#_Toc115958916)

[2.2 АВЛ 10](#_Toc115958917)

[2.2.1 Элемент дерева – односвязный список 11](#_Toc115958918)

[3 Требования к информационной системе 12](#_Toc115958919)

[3.1 Требования к данным 12](#_Toc115958920)

[3.1.1 Требования к входным данным 12](#_Toc115958921)

[3.1.2 Требования к выходным данным 13](#_Toc115958922)

[3.2 Функциональные требования 14](#_Toc115958923)

[4 Реализация 16](#_Toc115958924)

[4.1Спецификация структур данных 16](#_Toc115958925)

[4.2Описание среды разработки 27](#_Toc115958926)

[4.3Руководство пользователя 27](#_Toc115958927)

[4.4 Тестирование 33](#_Toc115958928)

[Заключение 38](#_Toc115958929)

[Список литературы 39](#_Toc115958930)

# Введение

В ходе выполнения курсового проекта, должен быть реализован пакет подпрограмм для работы со справочниками.

Целью курсового проекта является: разработка информационной системы «Сортировщик книг». Подсистема работы с двумя справочниками «Авторы» и «Книги».

Для достижения поставленной цели необходимо выполнить следующие задачи:

1. Провести анализ предметной области «Сортировщик книг».

2. Изучить теоретические основы методов построения справочников.

3. Определить требования к информационной системе.

4. Реализовать и провести тестирование.

# 1 Анализ предметной области

Требуется разработать информационную систему для предметной области (ПО) «Сортировщик книг».

Система должна решать следующие задачи:

1. хранить, позволять просматривать, добавлять и удалять информацию из справочников «Авторы» и «Книги»;
2. позволять искать информацию об авторах по ФИО автора и о книгах по ФИО автора и жанру;
3. выводить информацию об авторе по заданному ФИО автора и жанру.
   1. Объекты предметной области

Исходя из задач, которые должна решать информационная система, выделены объекты ПО, представленные в Таблице 1.

Таблица 1 –Объекты ПО

|  |  |
| --- | --- |
| Тип и название объекта | Описание объекта |
| Справочник «Авторы» | Хранит информацию по каждому автору книги |
| Справочник «Книги» | Хранит информацию окнигах |
| Отчет «Сортировщик книг» | Хранит информацию об авторе, пишущем в определенном жанре. |

Каждый автор характеризуется следующими параметрами: сайт, ФИО, страна.

**ФИО** – строка состоящая из букв русского алфавита и пробелов.

**Сайт** – строка состоящая из символов.

**Страна** – строка состоящая из букв русского алфавита и пробелов.

Пример справочника «Авторы» представлен в таблице 2.

Таблица2–Справочник «Авторы»

|  |  |  |
| --- | --- | --- |
| **Сайт** | **ФИО** | **Страна** |
| <https://ru.wikipedia.org/wiki/Достоевский,_Федор_Михайлович> | Федор Михайлович Достоевский | Россия |
| <https://ru.wikipedia.org/wiki/Бердяев,_Николай_Александрович> | Бердяев Николай Александрович | Россия |
| <https://ru.wikipedia.org/wiki/Фолкнер,_Уильям> | Уильям Фолкнер | США |
| <https://vk.com/id218232615> | Семченко Иванна Олеговна | Россия |
| <https://ru.wikipedia.org/wiki/Замятин,_Евгений_Иванович> | Замятин Евгений Иванович | Россия |
| <https://ru.wikipedia.org/wiki/Слоун,_Альфред> | Альфред Причарт Слоун | США |
| <https://ru.wikipedia.org/wiki/Зусак,_Маркус> | Макус Зусак | Австралия |
| <https://ru.wikipedia.org/wiki/Никулин,_Юрий_Владимирович> | Никулин Юрий Владимирович | Россия |
| <https://ru.wikipedia.org/wiki/Сэлинджер,_Джером_Дэвид> | Джером Дэвид Сэлинджер | США |
| <https://ru.wikipedia.org/wiki/Ремарк,_Эрих_Мария> | Эрих Мария Ремарк | Германия |

Каждая книга характеризуется следующими параметрами:название, ФИО автора, год выпуска, жанр.

**Название** – слово, состоящее из букв русского алфавита и символов.

**ФИО** – строка состоящая из букв русского алфавита и пробелов.

**Год** – целое число от 1000 до 9999.

**Жанр** – строка состоящая из букв русского алфавита и пробелов.

Пример справочника «Книги» представлен в таблице 3.

Таблица3–Справочник «Книги»

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **ФИО** | **Год** | **Жанр** |
| Там где синеют маки | Семченко Иванна Олеговна | 2021 | Проза |
| Мы | Замятин Евгений Иванович | 1920 | Роман |
| Три товарища | Эрих Мария Ремарк | 1932 | Роман |
| Записки из Мертвого дома | Федор Михайлович Достоевский | 1862 | Повесть |
| Когда я умирала | Уильям Фолкнер | 1930 | Роман |
| Книжный вор | Макус Зузок | 2005 | Военный |
| Почти серьезно | Никулин Юрий Владимирович | 2020 | Биография |
| Мои годы в дженирол моторс | Альфред Причарт Слоун | 1963 | Биография |
| Тени в раю | Эрих Мария Ремарк | 1998 | Роман |
| На западном фронте без перемен | Эрих Мария Ремарк | 1928 | Военная проза |

1.2 Законы ПО

1. У каждого автора может быть несколько книг;
2. У одной книги может быть только один автор;
3. Не может быть книги без автора;
4. Не может быть книга у несуществующего автора;
5. У автора может не быть книг.

1.3 Постановка задачи «Формирование отчета «Сортировщик книг»»

Входные данные: Справочник «Авторы», Справочник «Книги»,ФИО, Жанр.

Выходные данные: Отчет «Сортировщик книг».ФИО, Отчет «Сортировщик книг».Страна, Отчет «Сортировщик книг».Сайт.

Связь:

Если выполняются условия поиска:

1. Справочник «Авторы».ФИО == ФИО
2. Справочник «Книги».Жанр == Жанр, то

Отчет «Сортировщик книг».ФИО = Справочник «Авторы».ФИО

Отчет «Сортировщик книг».Страна = Справочник «Авторы».Страна

Отчет «Сортировщик книг».Сайт = Справочник «Авторы».Сайт

# 2 Теоретическая часть

Для работы со справочником «Авторы», необходима проверка на уникальность автора, при добавлении новых данных. Поскольку ключ ФИО уникален, то для этого поиска требуется хеш-таблица. Из-за того, что ключ является строкой, была выбрана хеш-функция свертки. Хеш-таблица является статической по размеру, поскольку количество записей в справочнике ограничено. Разрешение коллизий происходит методом открытой адресации.

Поскольку у каждого автора может быть несколько книг (1 закон ПО), то в справочнике «Книги» ФИО и жанр становятся неуникальными. Для добавления, поиска и удаления неуникальных ключей ФИО и жанра решено использовать АВЛ дерево.

2.1 Хеш-таблица

Хеш-таблица – это массив Т[0..m-1], в котором номер ячейки для ключа key вычисляется с помощью хеш-функции h. Функция h отображает совокупность ключей Uна ячейки хеш-таблицы Т[0..m-1]: , где размерm хеш-таблицы обычно гораздо меньше значения |U|.[2]

Существуют два основных варианта хеш-таблиц: с цепочками и открытой адресацией.[2]

Выполнение операции добавления, удаления или поиска начинается с вычисления хеш-функции для ключа. Получающееся хеш-значение, i=hash(key),играет роль индекса ключа в массиве H. После этого выполняется операция добавления, удаления или поиска.[2]

Ситуация, когда для различных ключей получается одно и то же хеш-значение, называется коллизией.К счастью, имеются эффективные технологии разрешения конфликтов, вызываемых коллизиями.[2]

В некоторых ситуациях можно избежать коллизии. К примеру, когда заранее известны все элементы (или очень редко меняются), то для них можно найти некоторую совершенную хеш-функцию, которая распределит их по ячейкам хеш-таблицы без коллизий. Хеш-таблицы, использующие подобные хеш-функции, не нуждаются в механизме разрешения коллизий, и называются хеш-таблицами с прямой адресацией[2].

2.1.1 Хеш-функция

Хеш-функция назначает каждому из ключей хеш-адрес, который представляет собой целое число от 0 до m-1 (где m– это размер массива)[2].

Хорошая функция преобразования ключей должна обеспечивать как можно более равномерное распределение ключей по всему диапазону значений индекса. Других ограничений на распределение нет, но на самом деле желательно, чтобы оно казалось случайным. Это свойство дало методу несколько ненаучное название хеширование. Очевидно, эта функция должна допускать эффективное вычисление, то есть состоять из очень небольшого числа основных арифметических операций [2].

Рассмотри вычислениезначения хеш-функции на примере метода свертки.Вычисление значения хеш-функции этим методом состоит в отображении ключа k на одну из 𝑚 ячеек путем получения остатка от деления 𝑘 на 𝑚, где 𝑘 - сумма всех цифр в строке.Тогда хеш-функция имеет вид:h(𝑘)=()𝑚𝑜𝑑𝑚, где 𝑘𝑖 символ ключа, i∈{1..k.length}[2]

Предположим, что размер хеш-таблицы равен 20, а на вход поступают записи авторов из ПО. Приведем 10 примеров:

Аналогично: Как видно из примера, два ключа могут быть хешированы в одну и ту же ячейку, например, для ключей () и (. Такая ситуация называется коллизией. Есть несколько методов решения коллизий – метод цепочек и метод открытой адресации: линейный, квадратичный и двойное хеширование.

2.1.2 Разрешение коллизий методом открытой адресации

При использовании метода открытой адресации все элементы хранятся непосредственно в хеш-таблице, т.е. каждая запись таблицы содержит либо элемент динамического множества, либо значение NIL. При поиске элемента мы систематически проверяем ячейки таблицы до тех пор, пока не найдем искомый элемент или пока не убедимся в его отсутствии в таблице. Таким образом, в методе открытой адресации хеш-таблица может оказаться заполненной и сделать вставку новых элементов будет невозможно [2].

Главным преимуществом этого метода является отсутствие указателей. Вместо того, чтобы следовать по указателям (метод цепочек), мы вычисляем последовательность проверяемых ячеек. Дополнительная память, освобождается в результате отказа от указателей, позволяет использовать хеш-таблицы большего размера при том же общем количестве памяти, потенциально приводя к меньшему количеству коллизий и более быстрой выборке.[2]

В рамках программы будет использоваться метод открытой адресации: линейный.

Пусть задана обычная хеш-функция , которую в дальнейшем будем именовать вспомогательной хеш-функцией. Линейный метод для вычисления последовательности исследований использует хеш-функцию для . Гдеi – номер попытки нахождения пустой ячейки или ключа,k – шаг (в программе он задан 1),key – ключ хеш-таблицы,m – размер хеш-таблицы.[2]

Таким образом, линейный метод сводиться к последовательному перебору ячеек с шагом 1.

На рисунке 1 представлен пример хеш-таблицы на записях предметной области, приведенных в главе 1.1 – Таблица2–Справочник «Авторы»

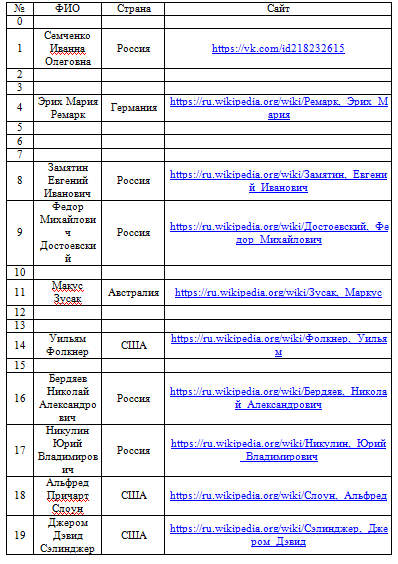


Рисунок 1 – «Метод открытой адресации»

2.2 АВЛ

АВЛ–дерево представляет собой бинарное дерево поиска, в котором показатель сбалансированности каждого узла, определяемый как разность высот левого и правого поддеревьев узла, равен 0, +1 или -1 (высота пустого дерева считается равной -1). Пример АВЛ–дерева представлен на рисунке 2.[1]

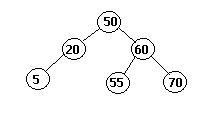


Рисунок 2 – «Пример АВЛ–дерева»

Перед добавлением нового ключа в непустое дерево, мы ищем его место, то куда его нужно поставить. Сначала мы сравниваем новый ключ с корнем дерева и если ключ больше корня, то идем вправо, иначе влево. Если мы не нашли его место, то снова сравниваем ключ с корнем и выбираем направление, иначе вставляем ключ. Мы продолжаем сравнивать ключ и корень до тех пор, пока не найдем свободное место. [1]

Если вставка нового узла делает АВЛ–дерево несбалансированным, то оно преобразуется при помощи поворота. Поворот в АВЛ–дереве представляет собой локальное преобразование поддерева, корень которого имеет показатель сбалансированности, равный +2 или -2; если таких узлов несколько, мы поворачиваем дерево с несбалансированными корнем, который наиболее близко к вновь вставленному листу. Всего имеется только четыре типа поворотов, причем два из них представляют собой зеркальное отображение двух других. В простейшей форме четыре возможных поворота показаны на рисунке 3[1]

Первый тип поворота – одиночный правый поворот илиR–поворот (представим поворот ребра, связывающего корень и его левый дочерний узел в бинарном дереве на рисунке 3.а, вправо). Обратим внимание, что такой поворот выполняется после вставки нового ключа в левое поддерево левого дочернего узла корня, который перед вставкой имел показатель сбалансированности +1. [1]

Симметричный ему одиночный левый поворотили L–поворот, представляет собой отражение одиночного R–поворота. Он выполняется после вставки нового ключа в правое поддерево правого дочернего узла корня, который перед вставкой имел показатель сбалансированности -1 (рисунок 3.б). [1]

Второй тип поворота – двойной право-левый поворот или LR–поворот. Он представляет собой объединение двух поворотов: выполняется L–поворот левого поддерева корня 1, за которым следует R–поворот нового поддерева, корнем которого является 1 (рисунок 3.в.). Он выполняется после вставки нового ключа в правое поддерево левого дочернего узла дерева, корень которого перед вставкой имеет показатель сбалансированности +1. [1]

Двойной право–левый поворот или RL–поворот, представляет собой зеркальное отображение двойного LR–поворота. Он выполняется после вставки нового ключа в левое поддерево правого дочернего узла дерева, корень которого перед вставкой имеет показатель сбалансированности -1. [1]

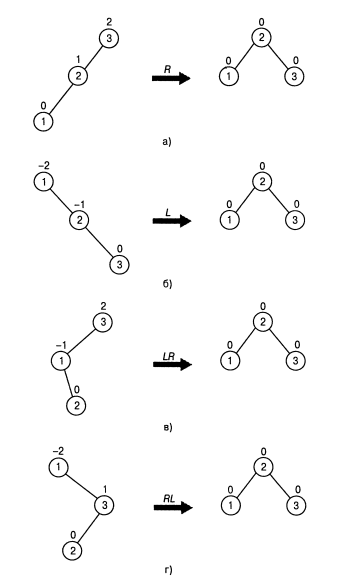


Рисунок 3 – «Повороты АВЛ–дерева»

Для удаления ключа из дерева, необходимо пройти по дереву, сравнивая корень и ключ. Если нужный корень найден, то он удаляется, а на его место ставиться минимальный элемент, справа от удаленного корня. При нарушении балансировки производиться один или несколько поворотов (зависит от необходимости) [1]

Приведем пример АВЛ–дерева на записях из ПО. (рисунок 4)

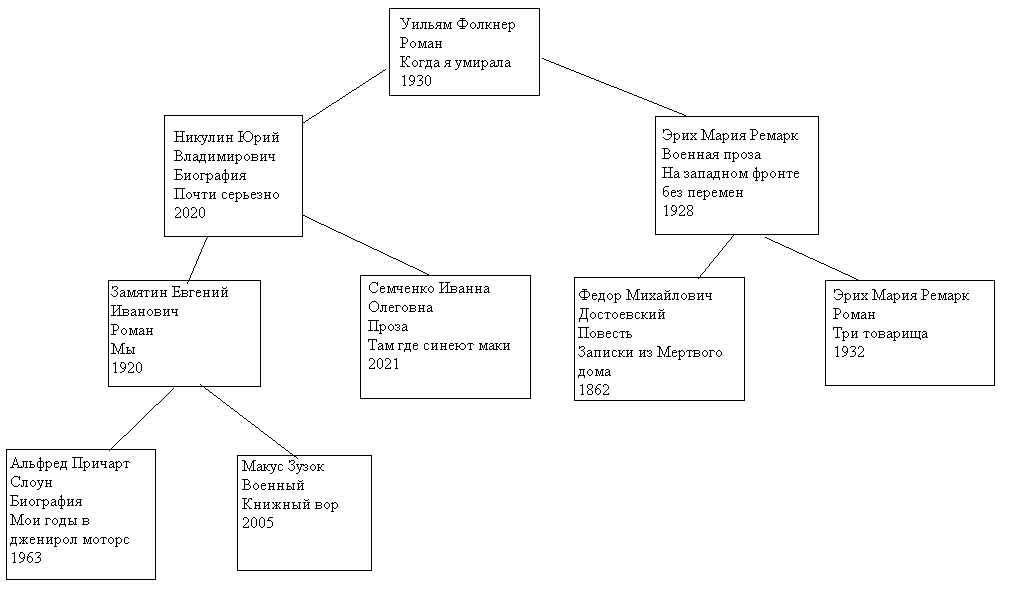


Рисунок 4 – «Пример АВЛ–дерева на записях из ПО»

2.2.1 Элемент дерева – односвязный список

Простейший способ связать набор элементов – выстроить их в простой список или очередь. Ибо в этом случае каждому элементу нужен единственный указатель на элемент, следующий за ним.

Пусть тип Node(узел) определен, тремя компонентами key – ключ, next – следующий элемент списка и возможно другая информация.

На рисунке 5 показан список узлов, причем указатель на его первый элемент хранится в переменной p типа Node. [3]

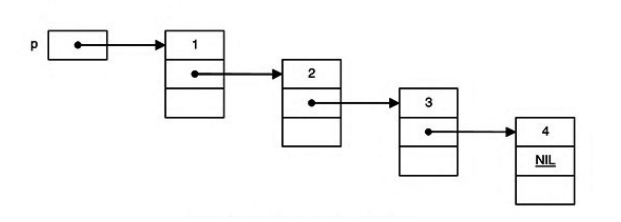


Рисунок 5 – «Пример связного списка»

Вставка элемента происходит в конец списка. У последнего элемента списка указатель будет ссылаться на новый элемент, а указатель нового элемента на NIL.[3]

Удаление элементов из списка может быть хоть в начале, хоть в середине, хоть в конце. Рассмотрим ситуацию удаления элемента в начале списка. В списке уже есть переменная p, указывающая на первый элемент списка. Добавим еще одну переменнуюt, она тоже будет указывать на первый элемент. Сдвинем ее на один шаг вперед. Получим, что p указывает на первый элемент, а tна вторую. И теперь у переменной p обнуляем указатель next, после говорим, что pравна t и обнуляем переменную t. Таким образом, мы избавились от первого элемента списка.[3]

Для удаления в конце списка необходимо так же добавитьеще две переменные t и g, изначально они указывают на первый элемент. Переменную t циклом доводим до последнего элемента, а переменную gдо предпоследнего. После указателюnextпеременнойgставим NIL. Таким образом, был удален последний элемент. [3]

И последняя ситуация, когда необходимо удалить элемент в середине списка. Все так же необходимо создать две переменные t и g, которые будут указывать на первый элемент. Циклом проходим переменной t, до нужного элемента в списке. После этого, так же циклом, двигаем переменную g, чтобы она стояла перед переменной t. После перекидываем указатель next у переменной g на элемент стоящий после t, а t обнуляем.[3]

# 3 Требования к информационной системе

3.1 Требования к данным

3.1.1 Требования к входным данным

Основываясь на анализе ПО, входными данными является:

* текстовый файл Author.txt, каждая строка файла содержит информацию об объекте «Авторы», а именно ФИО, Страна, сайт, разделенные символом \*.Входной файл, обязан начинаться со слова «Author» и иметь txt формат. Входной файл, удовлетворяющий вышесказанным условиям, является заведомо верным.

Пример текстового файла:

https://vk.com/id218232615\*Семченко Иванна Олеговна\*Россия

https://ru.wikipedia.org/wiki/Ремарк,\_Эрих\_Мария\*Эрих Мария Ремарк\*Германия

https://ru.wikipedia.org/wiki/Замятин,\_Евгений\_Иванович\*Замятин Евгений Иванович\*Россия

https://ru.wikipedia.org/wiki/Достоевский,\_Федор\_Михайлович\*Федор Михайлович Достоевский\*Россия

https://ru.wikipedia.org/wiki/Зусак,\_Маркус\*Макус Зузок\*Австралия

https://ru.wikipedia.org/wiki/Фолкнер,\_Уильям\*Уильям Фолкнер\*США

https://ru.wikipedia.org/wiki/Бердяев,\_Николай\_Александрович\*Бердяев Николай Александрович\*Россия

https://ru.wikipedia.org/wiki/Никулин,\_Юрий\_Владимирович\*Никулин Юрий Владимирович\*Россия

https://ru.wikipedia.org/wiki/Слоун,\_Альфред\*Альфред Причарт Слоун\*США

https://ru.wikipedia.org/wiki/Сэлинджер,\_Джером\_Дэвид\*Джером Дэвид Сэлинджер\*США

* текстовый файл Book.txt, каждая строка файла содержит информацию об объекте «Книги», а именно ФИО, Название, Жанр, Год, разделенные символом \*. Входной файл, обязан начинаться со слова «Book» и иметь txt формат. Входной файл, удовлетворяющий вышесказанным условиям, является заведомо верным.

Пример текстового файла:

Три товарища\*Эрих Мария Ремарк\*1932\*Роман

На западном фронте без перемен\*Эрих Мария Ремарк\*1928\*Военная проза

Записки из Мертвого дома\*Федор Михайлович Достоевский\*1862\*Повесть

Когда я умирала\*Уильям Фолкнер\*1930\*Роман

Там где синеют маки\*Семченко Иванна Олеговна\*2021\*Проза

Почти серьезно\*Никулин Юрий Владимирович\*2020\*Биография

Книжный вор\*Макус Зузок\*2005\*Военный

Мы\*Замятин Евгений Иванович\*1920\*Роман

Мои годы в дженирол моторс\*Альфред Причарт Слоун\*1963\*Биография

* ФИО, Страна, Сайт, Название, Жанр, Год (см. п. 1.1)

3.1.2 Требования к выходным данным

Выходными данными являются:

* текстовый файл Author.txt, каждая строка файла содержит информацию об объекте «Авторы», а именно ФИО, Страна, Сайт, разделенный символом \*.
* текстовый файл Book.txt, каждая строка файла содержит информацию об объекте «Книги», а именно ФИО, Название, Год, Жанр, разделенный символом \*.
* текстовый файл History.txt хранит одну строку, которая содержит информацию об объекте «Авторы», а именно ФИО, Страна, сайт, разделенные пробелом.

Сообщения об ошибках:

* "Такой автор уже есть"
* "Поля оказались пусты"
* "Такого автора еще нет. Добавьте для начала автора"
* "Такой книги нет, чтобы ее можно было удалить"
* "Нет ни одной книги"
* "Автор не пишет в таком жанре"
* "Список книг пуст"
* "Чтобы удалить этого автора, надо удалить все его книги. Вы этого хотите?"

3.2 Функциональные требования

Информационная система, должна позволять:

* считывать данные справочников из текстовых файлов;
* добавлять информацию в справочник «Авторы» по значениям: ФИО, Страна, Сайт;
* проверять добавляемые значения в справочник «Авторы» на корректность, а именно: ФИО –строка состоящая из букв русского алфавита и пробелов, Страна – строка состоящая из букв русского алфавита и пробелов, Сайт –строка состоящая из символов;
* добавлять информацию в справочник «Книги» по значениям: Название, ФИО, Год, Жанр;
* проверять добавляемые значения в справочник «Книги» на корректность, а именно: Название – слово, состоящее из букв русского алфавита и символов, Год – целое число от 1000 до 9999, Жанр – строка состоящая из букв русского алфавита и пробелов, ФИО – строка состоящая из букв русского алфавита и пробелов;
* соблюдать целостность между справочниками «Авторы» и «Книги» при добавлении книги в справочник «Книги», т.е. если ФИО автора есть в справочнике «Авторы», то добавляем в справочник книгу, иначе выводить ошибку;
* проверять на уникальность ФИО автора, введенное пользователем, при добавлении в справочник «Авторы» и выводить соответствующую ошибку при повторном добавлении;
* удалять информацию из справочника «Авторы» по значению ФИО;
* удалять информацию из справочника «Книги» по значениям: ФИО, Название, Жанр;
* соблюдать целостность между справочниками «Авторы» и «Книги» при удалении из справочника «Авторы», т.е. если у автора нет связных записей в справочнике «Книги», то удалить из справочника, иначе выводить соответствующую ошибку;
* сохранять все внесенные изменения в справочники «Авторы» и «Книги» в файл;
* создавать отчет «Справочник книг» по заданному ФИО и Жанру;
* сохранять отчет «Справочник книг» в заданный файл;
* находить информацию в справочниках «Авторы» и «Книги» по значениям: ФИО, Название, Жанр, Год, Страна, Сайт и выводитьсоответствующую информацию о существовании (несуществовании).

# 4 Реализация

4.1Спецификация структур данных

Тут описание СД для справочников и отчета.

Класс Key\_1 - класс, описывающий ключ в дереве, а так же односвязный список.

Поля:

* Key\_1 next – указатель на следующий элемент
* fio – ФИО автора
* name – название книги
* genre – жанр книги
* year – год издания книги

Методы:

* Key\_1(stringname\_v, string fio\_v, string genre\_v, string year\_v) – конструкторсписка;

Входные данные: название книги, ФИО автора, жанр книги, год издания книги;

Формальные параметры:переменная stringname\_v – название книги, переменная stringfio\_v – фамилия автора книги, переменная stringgenre\_v – жанр книги, переменнаяstringyear\_v – год издания книги;

Выходные данные: заполненные поля класса.

Класс AVLTree - класс, описывающий АВЛ-дерево.

Поля:

* Класс Node – класс, описывающий корень дерева
  + Поля:
    - int balans – переменна, хранит баланс узла дерева
    - Key\_1 key – переменная, хранит ключ
    - Node left – указатель на левый потомок узла дерева
    - Node right – указатель на правый потомок узла дерева

Методы:

* Node(string name\_v, string fio\_v, string genre\_v, string year\_v) –конструкторкорнядерева;

Входные данные: название книги, ФИО автора, жанр книги, год издания книги;

Формальные параметры:переменная stringname\_v – название книги, переменная stringfio\_v – фамилия автора книги, переменная stringgenre\_v – жанр книги, переменная stringyear\_v – год издания книги;

Выходные данные: заполненные переменные класса и ключ дерева.

* Поля:
  + Node root – корень дерева

Методы:

* void Add(ref Node tree, string name\_v, string fio\_v, string genre\_v, string year\_v, ref bool flag) – вспомогательныйметоддляметода Check(string name\_v, string fio\_v, string genre\_v, string year\_v, bool flag);

Входные данные: дерево, название книги, ФИО автора, жанр книги, год издания книги, флаг;

Формальныепараметры:переменная Nodetree – дерево, переменная stringname\_v – название книги, переменная stringfio\_v – фамилия автора книги, переменная stringgenre\_v – жанр книги, переменная stringyear\_v – год издания книги, переменная boolflag – показывает, необходимость балансировки дерева;

Выходные данные: дерево с добавленным ключом.

* VoidBalanceL(refNodetree, refboolflag) – вспомогательныйметоддляметодаDeleteOne(refNodetree, refboolflag, stringfio\_v, stringgenre\_v, stringname\_v)иDeleteOne\_fio(refNodetree, refboolflag, stringfio\_v);

Входные данные: дерево, флаг;

Формальные параметры:переменная Nodetree – дерево,переменная boolflag – показывает, необходимость балансировки дерева;

Выходные данные: сбалансированное дерево.

* void BalanceR(ref Node tree, ref bool flag)– вспомогательныйметоддляметодаDeleteOne(refNodetree, refbooflag, stringfio\_v, stringgenre\_v, stringname\_v)иDeleteOne\_fio(refNodetree, refboolflag, stringfio\_v);

Входные данные:дерево, флаг;

Формальные параметры: переменная Nodetree – дерево,переменная boolflag – показывает, необходимость балансировки дерева;

Выходные данные:сбалансированное дерево.

* VoidDeleteTwo(refNoder, refboolflag, refNodeq) – вспомогательныйметоддляметодаDeleteOne(refNodetree, refboolflag, stringfio\_v, stringgenre\_v, stringname\_v)иметодаDeleteOne\_fio(refNodetree, refboolflag, stringfio\_v);

Входные данные: указатель на правуювершину дерева, относительно удаляемого корня, флаг, указатель на удаляемую вершину дерева;

Формальные параметры: переменнаяNoder – указатель на минимальную справа вершину дерева, переменнаяboolflag – показывает, необходимость балансировки дерева,переменная Nodeq – указатель на удаляемую вершину дерева;

Выходные данные: дерево с удаленным ключом.

* voidDeleteOne(refNodetree, refboolflag, stringfio\_v, stringgenre\_v, stringname\_v) – вспомогательныйметоддляметодаDelete(stringfio\_v, stringgenre\_v, stringname\_v, boolflag);

Входные данные: дерево, название книги, ФИО автора, жанр книги, флаг;

Формальные параметры: переменная Nodetree – дерево, переменная stringname\_v – название книги, переменная stringfio\_v – фамилия автора книги, переменная stringgenre\_v – жанр книги, переменная boolflag – показывает, необходимость балансировки дерева;

Выходные данные: дерево с удаленным ключом.

* VoidDeleteOne\_fio(refNodetree, refboolflag, stringfio\_v) – вспомогательныйметоддляметодаDelete\_fio(stringfio\_v, boolflag);

Входные данные: дерево, ФИО автора, флаг;

Формальные параметры: переменная Nodetree – дерево, переменная boolflag – показывает, необходимость балансировки дерева, переменная stringfio\_v – фамилия автора книги;

Выходные данные:дерево с удаленным ключом.

* boolCheck\_Number(stringvalue)­– вспомогательныйметоддляметода Check(string name\_v, string fio\_v, string genre\_v, string year\_v, bool flag);

Входные данные: переменная, которую необходимо проверить;

Формальные параметры: переменная stringvalue – строка содержащая только буквы русского или английского алфавита, а так же пробелы;

Выходные данные:trueили false.

* boolCheck\_Letter(stringvalue)­– вспомогательныйметоддляметода Check(string name\_v, string fio\_v, string genre\_v, string year\_v, bool flag);

Входные данные:переменная, которую необходимо проверить;

Формальные параметры:переменная stringvalue – строка содержащая только 4 цифры;

Выходные данные:trueили false.

* void Cleaning(ref Node tree)

Входные данные: дерево;

Формальные параметры: переменная Nodetree – АВЛ–дерево;

Выходные данные: пустое дерево.

* void Drawing(ref Node tree, int h, ref List<string> list) – вспомогательныйметоддляметодаDraw();

Входные данные: дерево, число пробелов в одной итерации, список;

Формальныепараметры:переменнаяNodetree – дерево, переменнаяinth – число на которое увеличиваются пробелы, переменнаяList<string>list – список, куда записываются вершины дерева, при обходе самого дерева;

Выходные данные: список всех ключей дерева.

* bool Seach\_Book(stringfio\_v, string genre\_v, string name\_v) – поискнужнойкнигивдереве;

Входные данные: дерево, ФИО автора, жанр книги, название книги;

Формальныепараметры:переменная stringfio\_v – фамилия автора книги, переменная stringgenre\_v – жанр книги, переменная stringname\_v – название книги;

Выходные данные:trueили false.

* voidgetSeach(stringfio\_v, stringgenre\_v, refList<string>list) – поисккнигавтораопределенногожанравсправочнике «Книги»;

Входные данные: дерево, ФИО автора, жанр книги, список книг;

Формальныепараметры: переменная stringfio\_v – фамилия автора книги, переменная stringgenre\_v – жанр книги, переменнаяList<string>list – список найденных книг;

Выходные данные:trueили false.

* boolSearch\_Author(stringfio\_v, stringgenre\_v, refintscore) – поискавторавдереве,пишущеговзаданномжанре;

Входные данные: дерево, ФИО автора, жанр книги, счетчик шагов по дереву;

Формальные параметры:переменная stringfio\_v – фамилия автора книги, переменная stringgenre\_v – жанр книги, переменная intscore – хранит количество шагов, пройденный по дереву во время поиска;

Выходные данные:trueили false.

* bool Search\_fio(stringfio\_v) – поискавторавдереве;

Входные данные: дерево, ФИО автора;

Формальные параметры: переменная stringfio\_v – фамилия автора книги;

Выходные данные:trueили false.

* voidRbl(refNodetree, refList<Key\_1>list) – вспомогательныйметоддляметода CopiAVL();

Входные данные: дерево, список;

Формальные параметры: переменнаяNodetree – дерево, переменнаяList<string>list – список, куда записываются вершины дерева;

Выходные данные: список ключей дерева.

* bool Check(stringname\_v, string fio\_v, string genre\_v, string year\_v, bool flag) – добавлениевершинывдерево;

Входные данные: дерево, название книги, ФИО автора, жанр книги, год издания книги, флаг;

Формальные параметры:переменная stringname\_v – название книги, переменная stringfio\_v – фамилия автора книги, переменная stringgenre\_v – жанр книги, переменная stringyear\_v – год издания книги, переменная boolflag – показывает, необходимость балансировки дерева;

Выходные данные:дерево с добавленным ключом.

* ~AVLTree() – деструктор дерева;

Входные данные:дерево;

Формальные параметры: отсутствуют;

Выходные данные: пустое дерево.

* voidDelete(stringfio\_v, stringgenre\_v, stringname\_v, boolflag) – удаляетключсзаданнымизначениями;

Входные данные:дерево, ФИО автора, название книги, жанр книги, флаг;

Формальные параметры: переменная stringfio\_v – фамилия автора книги, переменная stringgenre\_v – жанр книги, переменная stringname\_v – название книги, переменная boolflag – показывает, необходимость балансировки дерева;

Выходные данные: дерево с удаленным ключем.

* voidDelete\_fio(stringfio\_v, boolflag) – удаляетвсеключис автором;

Входные данные: дерево, ФИО автора, флаг;

Формальные параметры: переменная stringfio\_v – фамилия автора книги, переменная boolflag – показывает, необходимость балансировки дерева;

Выходные данные: дерево с удаленными ключами заданного автора.

* boolCheckRot() – проверяет дерево на пустату;

Входные данные: дерево;

Формальные параметры: отсутствуют;

Выходные данные:true илиfalse.

* voidReadfile(boolflag) – читаетданныеиз файла;

Входные данные: флаг;

Формальные параметры: переменная boolflag – показывает, необходимость балансировки дерева;

Выходные данные: дерево, заполненное данными из файла.

* voidWrite\_file() –записываетданные в файл;

Входные данные: дерево;

Формальные параметры: отсутствуют;

Выходные данные: файл, заполненный данными из дерева.

* AVLTree() – конструктор дерева;

Входные данные: отсутствуют;

Формальные параметры:отсутствуют;

Выходные данные:отсутствуют.

* voidClear() – удаление всех вершин дерева;

Входные данные: дерево;

Формальные параметры: отсутствуют;

Выходные данные: пустое дерево.

* List<string>Draw() – выписываетвсе ключи построчно в список;

Входные данные: дерево;

Формальные параметры: отсутствуют;

Выходные данные: список строк.

* List<Key\_1>CopiAVL() – выписываетвсе ключи дерева в список;

Входные данные: дерево;

Формальные параметры: отсутствуют;

Выходные данные: список всех ключей дерева.

Структура Notebook – структура, описывающая поля хеш-таблицы.

* Поля:
  + string website – ссылка на сайт писателя;
  + string fio – ФИО писателя;
  + stringcountry – страна рождения писателя;
  + intstate – статус строки (содержит элемент, элумент удален, строка пуска);
  + intindex1 – индекс элемента, вычисленный первичной хеш-функцией;
  + intindex2 – индекс элемента, вычисленный вторичной хеш-функцией;

КлассHash – класс, описывающий хеш-таблицу.

* Поля:
  + Notebook[] table – хеш-таблица;
  + int buffer\_size – размерхеш-таблицы;

Методы:

* intHash\_One\_Function(stringkey) – метод, высчитывающийпервичнуюхэш-функцию;

Входные данные:ключ для хеш-функции;

Формальные параметры:переменная stringkey – ключ, для которого считается хеш-функция;

Выходные данные:целочисленное число – хэш-функция заданного ключа.

Пример:

Предположим, что размер ХТ равен 20, а на вход поступает запись, содержащая информацию об авторе из ПО (см 1.1)

Hash1(key) = ()%buffer\_size;

Значение ключа =

Аналогично:

* intHash\_Two\_Function(intindex, intj) – метод, вычисляющийвторичнуюхеш-функцию;

Входные данные: индекс элемента, номер попытки;

Формальные параметры: переменная intindex – индекс элемента в хеш-таблице, переменная intj – номер попытки разрешения коллизии;

Выходные данные:число – вторичная хэш-функция заданного числа.

Пример:

Предположим, что размер ХТ равен 20,номер попыткиj равен 1 и на вход поступает запись, содержащая информацию об авторе из ПО(см 1.1)

Hash2(key) = (Hash1(key) + j\*1)%buffer\_size;

Значение ключа =

Аналогично:

* intCollision(intindex, stringbuff) – методнахождения индексапустой строки, приколлизии в добавлении элемента;

Входные данные: хеш-таблица, индекс первичной хеш-функции, строка ключ;

Формальные параметры:переменная intindex – индекс первичной хеш-функции, переменная stringbuff – строка, которую необходимо вставить;

Выходные данные: новый индекс для строки.

* intDecision(intindex, stringvalue) – методнахожденияиндекса строки, приколлизии в удалении;

Входные данные: хеш-таблица, индекс первичной хеш-функции, строка ключ;

Формальные параметры:переменная intindex – индекс первичной хеш-функции, переменная stringbuff – строка, которую удаляем;

Выходные данные:новый индекс для строки.

* bool Check\_Letter(stringvalue) – вспомогательныйметоддляметода Check(string fio\_v, string website\_v, string country\_v);

Входные данные:строка;

Формальные параметры: переменная stringvalue – строка, которая проверяется на корректность;

Выходные данные:trueили false;

* intSize\_table() –возвращаетразмер таблицы;

Входные данные:переменная buffer\_size;

Формальные параметры: отсутствуют;

Выходные данные: размер таблицы.

* NotebookCopielement(intindex) – возвращаетэлементтаблицы;

Входные данные:хеш-таблица, индекс элемента;

Формальные параметры: переменная intindex – число под которым стоит элемент в хеш-таблице;

Выходные данные: строка хеш-таблицы.

* Hash() – конструктор, создает хеш-таблицу размерности buffer\_size;

Входные данные:отсутствуют;

Формальные параметры: отсутствуют;

Выходные данные:хеш-таблица размера buffer\_size;

* bool Add(string fio\_v, stringwebsite\_v, string country\_v) – вспомогательныйметоддляметода Check(string fio\_v, string website\_v, string country\_v);

Входные данные: хеш-таблица, ФИО автора, ссылка на сайт автора, страна рождения автора;

Формальные параметры: переменная stringfio\_v – ФИО автора, переменная stringwebsite\_v – ссылка на сайт автора, переменная stringcountry\_v – страна рождения автора;

Выходные данные: хеш-таблица с добавленной записью.

* void Check(string fio\_v, stringwebsite\_v, string country\_v) – добавлениеновыхавтороввхеш-таблицу;

Входные данные:хеш-таблица, ФИО автора, ссылка на сайт автора, страна рождения автора;

Формальные параметры: переменная stringfio\_v – ФИО автора, переменная stringwebsite\_v – ссылка на сайт автора, переменная stringcountry\_v – страна рождения автора;

Выходные данные:хеш-таблица с добавленным автором.

* boolDel(stringfio\_v) – удаление авторов из хеш-таблицы;

Входные данные: хеш-таблица, ФИО автора;

Формальные параметры: переменная stringfio\_v – ФИО автора;

Выходные данные: хеш-таблица с удаленным автором.

* intSearch(stringfio\_v, ref int score) – поискавторавхеш-таблице;

Входные данные: хеш-таблица, ФИО автора, счетчик;

Формальные параметры: переменная stringfio\_v – ФИО автора, переменная intscore – счетчик шагов, пройденных по таблице во время поиска;

Выходные данные: индекс, под которым стоит автор в хеш-таблице.

* voidRead\_File()­– чтениеданных из файла и заполнение хеш-таблицы;

Входные данные:хеш-таблица;

Формальные параметры: отсутствуют;

Выходные данные: заполненная хеш-таблица.

* voidWrite\_file() –записьавторов в файл, из хеш-таблицы;

Входные данные:хеш-таблица;

Формальные параметры: отсутствуют;

Выходные данные:файл с авторами из хеш-таблицы.

* voidWrite\_History(intindex) – записьвфайлавтора, которогоискаливметоде int Search(stringfio\_v);

Входные данные: хеш-таблица, индекс элемента в таблице;

Формальные параметры: переменная intindex – индекс элемента в хеш-таблице;

Выходные данные: файл с записью.

* ~Hash() – деструктор хеш-таблицы;

Входные данные: отсутствуют;

Формальные параметры: отсутствуют;

Выходные данные: отсутствуют.

4.2Описание среды разработки

Для реализации была выбрана интегрированная среда разработки Visual Studio(VS), в которой можно свободно разрабатывать на С#.В ней так же присутствует графический интерфейс, который можно создать с помощью платформы .NET. Интерфейс разрабатывался с помощью технологии WindowsForm, поскольку в нем нет нужды писать все элементы кодом, а можно выбрать из списка предложенных. При этом каждому элементу можно удобно менять свойства и названия. В качестве разработки был выбран язык C#.

4.3Руководство пользователя

**Справочник «Книги»**

Для того чтобы добавить новые записи в справочник «Книги», необходимо перейти на вкладку «Основное окно» (рисунок 6), ввести данные в поля «ФИО» «Название» «Жанр» «Год» и нажать кнопку «Добавить», справа от полей (рисунок 7). Так же в «Основном окне» можно посмотреть записи справочника «Книги».

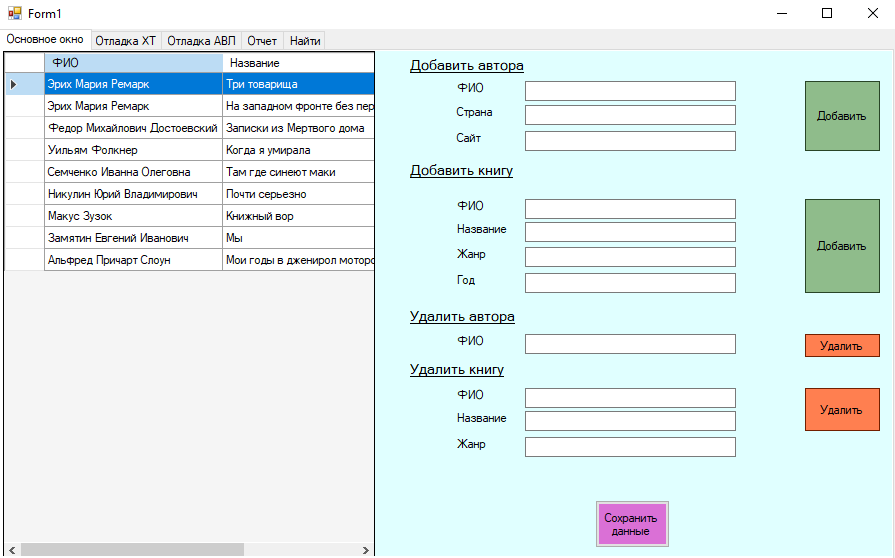


Рисунок 6 –«Вкладка «Основное окно»»

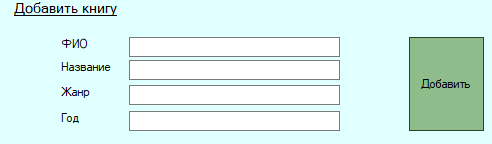


Рисунок 7 – «Добавление в справочник «Книги»»

Чтобы удалить книгу, необходимо, в той же кладке, ввести данные в поля «ФИО» «Название» «Жанр» и нажать кнопку «Удалить» , справа от полей (рисунок 8).



Рисунок 8 – «Удаление из справочника «Книги»»

Чтобы найти книги автора в определенном жанре, надо перейти на вкладку «Найти» (рисунок 9), ввести данные в поля «ФИО» «Жанр» и нажать кнопку «Найти», справа от полей (рисунок 10)

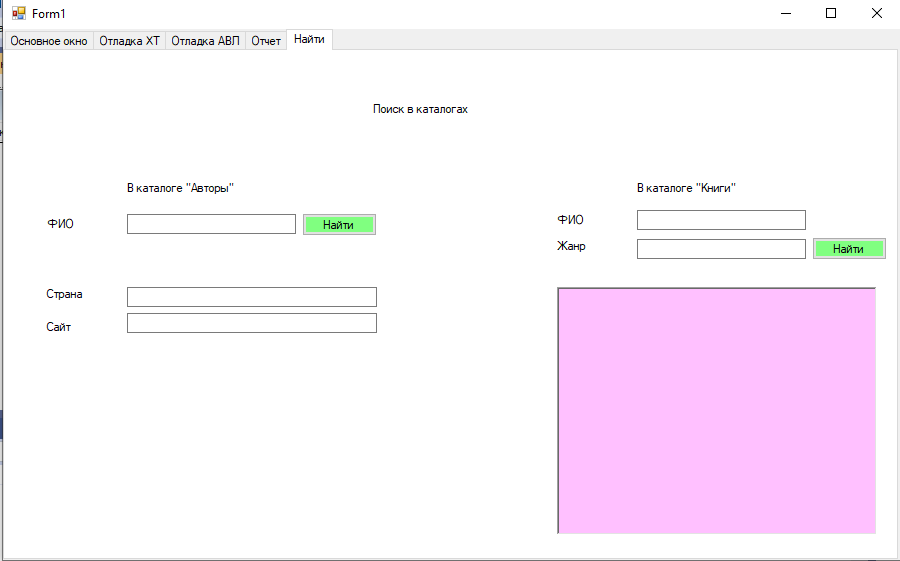


Рисунок 9 – «Окно поиска в двух справочниках»

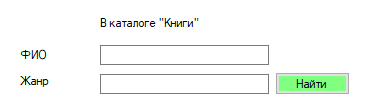


Рисунок 10 – «Поиск книги автора»

Данные автора будут выведены ниже в той же кладке (рисунок 11).

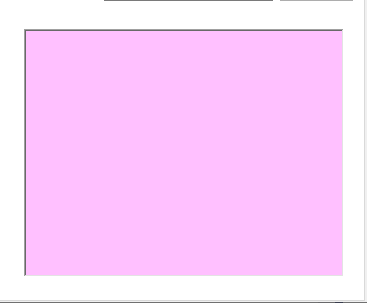


Рисунок 11 – «Выходные данные при поиске книги»

**Справочник «Авторы»**

Для того чтобы добавить новые записи в справочник «Авторы», необходимо перейти на вкладку «Основное окно» (рисунок 6), ввести данные в поля «ФИО» «Страна» «Сайт» и нажать кнопку «Добавить», справа от полей (рисунок 12).



Рисунок 12 – «Добавление в справочник «Авторы»»

Чтобы удалить автора, необходимо ввести ФИО автора в поле «ФИО», в той же кладке, и нажать кнопку «Удалить» , справа от полей (рисунок 13).

удалить автора.png

Рисунок 13 – «Удаление из справочника «Авторы»»

Чтобы найти автора, надо перейти на вкладку «Найти» (рисунок 9), ввести данные в поля «ФИО» и нажать кнопку «Найти», справа от полей (рисунок 14)

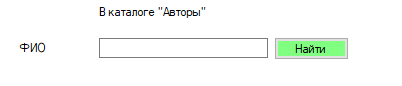


Рисунок 14 – «Поиск данных автора»

Данные автора будут выведены ниже, в той же вкладке (рисунок 15).

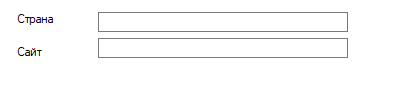


Рисунок 15 – «Выходные данные при поиске автора»

**Отчет «Сортировщик книг»**

Перейдя на вкладку «Отчет», можно видеть функционал для формирования отчета (рисунок 16)

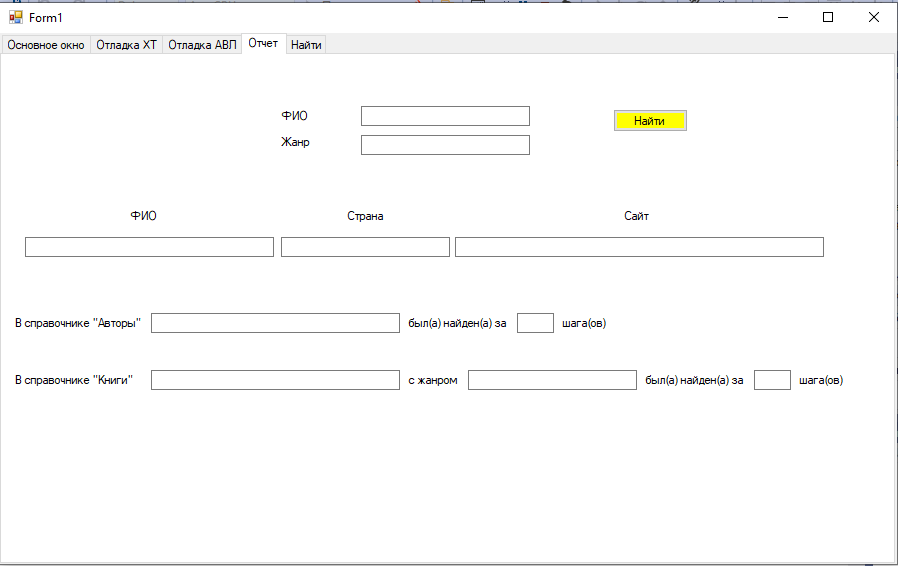


Рисунок 16 – «Вкладка «Отчет»»

Чтобы сформировать отчет необходимо ввести данные в поля «ФИО» «Жанр» и нажать кнопку «Найти» (рисунок 17). Результат поиска будет сразу сохранен в файл «History.txt»

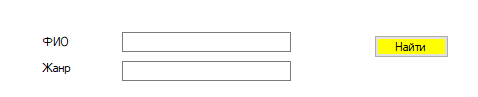


Рисунок 17 – «Поиск в двух справочниках»

В полях ниже будет выведен результат поиска в двух справочниках (рисунок 18)



Рисунок 18 – «Результат поиска»

Так же в этом будут выведены данные о том, сколько шагов пришлось проделать программе, в хеш-таблице и АВЛ дереве, для нахождения данных в каждом справочнике (рисунок 19)

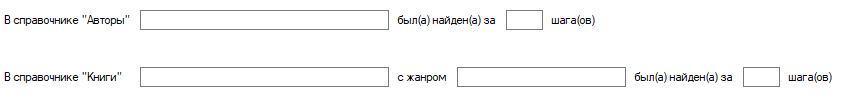


Рисунок 19 – «Количество шагов в АВЛ дереве и хеш-таблице»

**Дополнительно**

Чтобы сохранить данные справочников «Книги» и «Авторы», необходимо перейти на вкладку «Основное окно» и нажать на кнопку «Сохранить данные» (рисунок 20)

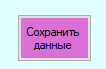


Рисунок 20 – «Кнопка «Сохранить данные»»

Перейдя на вкладку «Отладка ХТ», можно наблюдать функционал работы со справочником «Авторы» (рисунок 21)

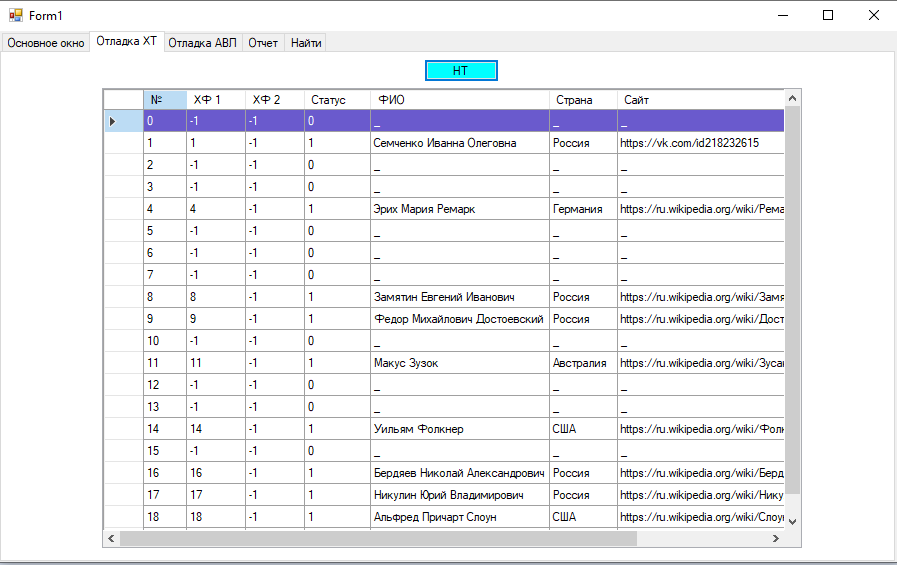


Рисунок 21 – «Вкладка «Отладка ХТ»»

Для выполнения отладки хеш-таблицы, необходимо нажать на кнопку «HT» (рисунок 22)

кнопка хт.png

Рисунок 22 – «Кнопка «HT»»

Перейдя на вкладку «Отладка АВЛ», можно наблюдать функционал работы со справочником «Книги» (рисунок 23)

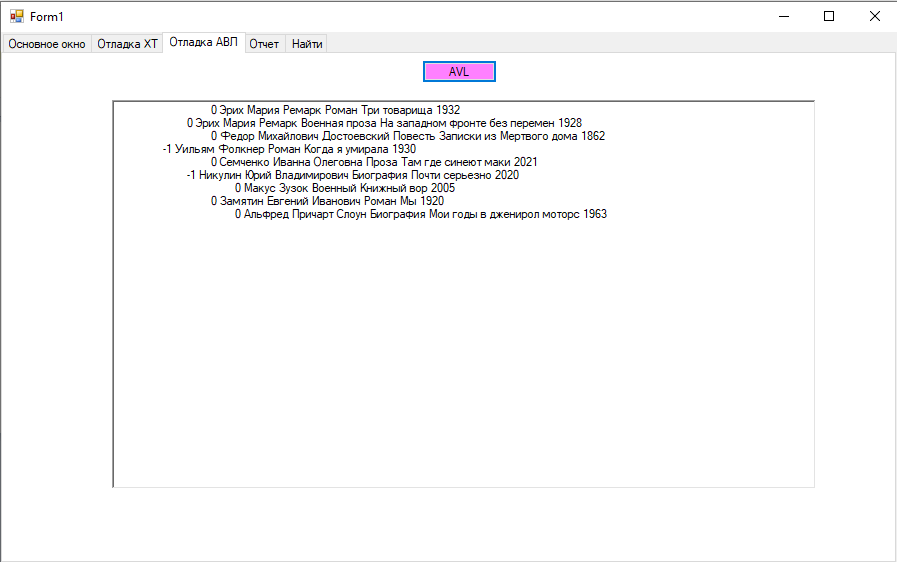


Рисунок 23 – «Вкладка «Отладка АВЛ»»

Для выполнения отладки авл дерева, необходимо нажать на кнопку «AVL» (рисунок 24)

кнопка авл.png

Рисунок 24 – «Кнопка «AVL»»

4.4 Тестирование

Тестирование проводилось по методу черного ящика. В таблице 4,5 описаны тестовые ситуации.

Таблица 4 – Тестирование работы со справочником «Авторы»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Описание тестовой ситуации | | Входные данные | | Выходные данные | |
| Хеш-таблица  (метод открытой адресации) | Атрибуты  (ФИО, ссылка на сайт, страна) | Хеш-таблица  (метод открытой адресации) | Атрибуты  (ФИО, ссылка на сайт, страна) |
| Добавление | | | | | |
| 1 | Добавление некорректных данных | – | Сем4енко Иванна Олеговна, <https://vk.com/id218232615>, Россия | – | Некорректно введены данные |
| – | Семченко Иванна Олеговна, <https://vk.com/id218232615>, Россия4 | – | Некорректно введены данные |
| 2 | Добавление корректных данных | – | Семченко Иванна Олеговна, <https://vk.com/id218232615>, Россия | Семченко Иванна Олеговна, <https://vk.com/id218232615>, Россия | Автор добавлена |
| 3 | Проверка уникальности ключа | Семченко Иванна Олеговна, <https://vk.com/id218232615>, Россия | Семченко Иванна Олеговна, <https://vk.com/id218232615>, Россия | Семченко Иванна Олеговна, <https://vk.com/id218232615>, Россия | Такой писатель уже существует |
| Семченко Иванна Олеговна, <https://vk.com/id218232615>, Россия | Макус Зузок, <https://ru.wikipedia.org/wiki/Зусак,_Маркус>, Австралия | Семченко Иванна Олеговна, <https://vk.com/id218232615>, Россия  Макус Зузок, <https://ru.wikipedia.org/wiki/Зусак,_Маркус>, Австралия | Автор добавлен |
| 4 | Добавление при коллизии | Никулин Юрий Владимирович, <https://ru.wikipedia.org/wiki/Никулин,_Юрий_Владимирович>, Россия | Джером Дэвид Сэлинджер, <https://ru.wikipedia.org/wiki/Сэлинджер,_Джером_Дэвид>, США | Никулин Юрий Владимирович, <https://ru.wikipedia.org/wiki/Никулин,_Юрий_Владимирович>, Россия  Джером Дэвид Сэлинджер, <https://ru.wikipedia.org/wiki/Сэлинджер,_Джером_Дэвид>, США  (Первичная хеш-функция совпадает, вторичная хеш-функция различна) | Автор добавлен |
| Поиск | | | | | |
| 5 | Запись не существует | – | Никулин Юрий Владимирович | – | – |
| 6 | Запись существует | Семченко Иванна Олеговна, <https://vk.com/id218232615>, Россия | Семченко Иванна Олеговна | Семченко Иванна Олеговна, <https://vk.com/id218232615>, Россия | Семченко Иванна Олеговна, <https://vk.com/id218232615>, Россия |
| 7 | Поиск при коллизии  (по ключу) | Никулин Юрий Владимирович, <https://ru.wikipedia.org/wiki/Никулин,_Юрий_Владимирович>, Россия  Джером Дэвид Сэлинджер, <https://ru.wikipedia.org/wiki/Сэлинджер,_Джером_Дэвид>, США | Джером Дэвид Сэлинджер | Никулин Юрий Владимирович, <https://ru.wikipedia.org/wiki/Никулин,_Юрий_Владимирович>, Россия  Джером Дэвид Сэлинджер, <https://ru.wikipedia.org/wiki/Сэлинджер,_Джером_Дэвид>, США | Джером Дэвид Сэлинджер, <https://ru.wikipedia.org/wiki/Сэлинджер,_Джером_Дэвид>, США |
| Удаление | | | | | |
| 8 | Запись не существует | – | Семченко Иванна Олеговна | – | Автор ненайден |
| 9 | Запись существует | Семченко Иванна Олеговна, <https://vk.com/id218232615>, Россия | Семченко Иванна Олеговна | – | Автор удален |
| 10 | Удаление при коллизии | Никулин Юрий Владимирович, <https://ru.wikipedia.org/wiki/Никулин,_Юрий_Владимирович>, Россия  Джером Дэвид Сэлинджер, <https://ru.wikipedia.org/wiki/Сэлинджер,_Джером_Дэвид>, США | Джером Дэвид Сэлинджер | Никулин Юрий Владимирович, <https://ru.wikipedia.org/wiki/Никулин,_Юрий_Владимирович>, Россия | Автор удален |
| 11 | Удаление при наличии книг в справочнике «Книги» | Семченко Иванна Олеговна, <https://vk.com/id218232615>, Россия | Семченко Иванна Олеговна | Семченко Иванна Олеговна, <https://vk.com/id218232615>, Россия | Нельзя удалить автора. У автора есть книги в справочнике «Книги» |
| 12 | Удаление при отсутствии книг в справочнике «Книги» | Семченко Иванна Олеговна, <https://vk.com/id218232615>, Россия | Семченко Иванна Олеговна | – | Автор удален |

Таблица 5 – Тестирование работы со справочником «Книги»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Описание тестовой ситуации | | Входные данные | | Выходные данные | |
| АВЛ–дерево | Атрибуты (Название, ФИО, год издания, жанр) | АВЛ–дерево | Атрибуты (Название, ФИО, год издания, жанр) |
| Добавление | | | | | |
| 1 | Добавление некорректных данных | – | Три товарища5, Эрих Мария Ремарк, 1932, Роман | – | Некорректно введены данные |
| – | Три товарища, Эри4х Мария Ремарк, 1932, Роман | – | Некорректно введены данные |
| – | Три товарища, Эрих Мария Ремарк, 193289, Роман | – | Некорректно введены данные |
| – | Три товарища, Эрих Мария Ремарк, 1932, Роман75 | – | Некорректно введены данные |
| 2 | Добавление корректных данных | – | Три товарища5, Эрих Мария Ремарк, 1932, Роман | Три товарища5, Эрих Мария Ремарк, 1932, Роман | Книга добавлена |
| 3 | Добавление корректных данных и существует автор (правый поворот) | авл правый повторот.png | Книжный вор, МакусЗузок, 2005, Военный | авл итог парвого поворота.png | Книга добавлена |
| 4 | Добавление корректных данных и существует автор (левый поворот) | авл левый поворот.png | Эрих Мария Ремарк, Роман, Три товарища, 1932 | авл итог левого поворота.png | Книга добавлена |
|  | Добавление корректных данных и существует автор (левый–правый поворот) | авл правый-левый поворот.png | Никулин Юрий Владимирович, Биография, Почти серьезно, 2020 | авл правый-л итог.png | Книга добавлена |
|  | Добавление корректных данных и существует автор (правый–левый поворот) | авл левый прав поворт.png | Никулин Юрий Владимирович, Биография, Почти серьезно, 2020 | авл правый-л итог.png | Книга добавлена |
|  | Добавление корректных данных и не существует автор | – | Книжный вор, Маркус Зузок, 2005, Военный | – | Нельзя добавить книгу у несуществующего автора |
| Поиск | | | | | |
| 5 | Запись не существует | – | Эрих Мария Ремарк, Роман | – | Книга не найдена |
| 6 | Запись существует | Эрих Мария Ремарк, Роман, Три товарища, 1932 | Эрих Мария Ремарк, Роман | Эрих Мария Ремарк, Роман, Три товарища, 1932 | Книга найдена |
| Удаление | | | | | |
| 8 | Запись не существует | – | Эрих Мария Ремарк, Роман, Три товарища | – | Книга не найдена |
| 9 | Запись существует (удаление корня дерева) | Эрих Мария Ремарк, Роман, Три товарища, 1932 | Эрих Мария Ремарк, Роман, Три товарища | – | Книга удалена |
| 10 | Запись существует (левый поворот) | авл удаление левый поврот.png | МакусЗузок, Военный, Книжный вор | авл удаление левый итог.png | Книга удалена |
| 11 | Запись существует (правый поворот) | авл удаление право.png | Эрих Мария Ремарк, Роман, Три товарища | авл удаление парво итог7.png | Книга удалена |
| 12 | Запись существует (левый–правый поворот) | авл удаление левый - правый.png | Эрих Мария Ремарк, Роман, Три товарища | авл удаление левый-правый итог.png | Книга удалена |
| 13 | Запись существует (правый–левый поворот) | авл удаление правый-левый.png | МакусЗузок, Военный, Книжный вор | авл удаление правый-левый итог.png | Книга удалена |

Таблица6–Тестирование работы с отчётом «Справочник книг»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Описание тестовой ситуации | | Входные данные | | Выходные данные | |
| Записи в справочнике  «Авторы» | Записи в справочнике  «Книги» | Данные, введённые пользователем | Выходные данные |
| 1 | Экспорт в некорректный файл | Семченко Иванна Олеговна  Россия  https://vk.com/id218232615 | Семченко Иванна Олеговна  Проза  Там где синеют маки  2021 | Семченко Иванна Олеговна  Проза | Неверный файл |
| 2 | У автора нет книги | Семченко Иванна Олеговна  Россия  https://vk.com/id218232615 | - | Семченко Иванна Олеговна  Проза | Таких авторов нет |
| 3 | Некорректно введен автор | Семченко Иванна Олеговна  Россия  https://vk.com/id218232615222333 33 | Семченко Иванна Олеговна  Проза  Там где синеют маки  2021 | Семченко55 Иванна Олеговна  Проза | Неправильно введённые данные |
| 4 | Некорректно введен жанр | Семченко Иванна Олеговна  Россия  https://vk.com/id218232615222333 33 | Семченко Иванна Олеговна  Проза  Там где синеют маки  2021 | Семченко Иванна Олеговна  Проза5 | Неправильно введённые данные |
| 5 | Автора нет | Семченко Иванна Олеговна  Россия  https://vk.com/id218232615222333 33 | Семченко Иванна Олеговна  Проза  Там где синеют маки  2021 | Никулин Юрий Владимирович  Биография | Такого автора нет |

# Заключение

Целью курсового проекта было: разработка информационной системы «Сортировщик книг».

Цель достигнута. Для достижения поставленной цели были выполнены следующие задачи:

1. Проведен анализ предметной области «Сортировщик книг» и построена ее модель;
2. Изучены теоретические основы методов построения справочников;
3. Определены требования к информационным системам;
4. Информационная система была реализована и спроектирована;
   1. Был изучен язык C#
   2. Изучена платформа .NET Framework 4.8
   3. Во время разработки, WindowsFormsбыл использован в качестве среды выполнения

# Список литературы

1. Левитин А.В. Алгоритмы: введение в разработку и анализ. / А. В. Левитин – Москва :Издательский дом «Вильямс», 2006. – 267с.
2. Кормен Т. Х. Алгоритмы: построение и анализ. / Т. Х. Кормен, Ч. И. Лейзерсон, Р. Л. Ривест, Клиффорд Штайн – 2-е изд. – Москва : Издательский дом «Вильямс», 2005. – 285 с.
3. Вирт Н. Алгоритмы и структуры данных. Новая версия для Оберона + CD / Н. Вирт – Москва :ДМК Пресс, 2010. – 175 с.