|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования |
| **«МИРЭА – Российский технологический университет»** |
| **РТУ МИРЭА** |
|  |

|  |  |
| --- | --- |
| **Отчет по выполнению практического задания № 6, часть 1** | |
| **Тема:** | |
| **«Быстрый доступ к данным с помощью хеш-таблиц»** | |
| Дисциплина: «Структуры и алгоритмы обработки данных» | |
|  | Выполнил студент: Лисовский И.В |
|  | Группа: ИКБО-21-23 |

Москва – 2024

СОДЕРЖАНИЕ

[1 ЦЕЛЬ 3](#_Toc179439570)

[2 ЗАДАНИЕ 4](#_Toc179439571)

[2.1 Формулировка задачи 4](#_Toc179439572)

[2.2 Математическая модель решения 4](#_Toc179439573)

[2.3 Код программы с комментариями 9](#_Toc179439574)

[2.4 Тестирование программы 12](#_Toc179439575)

[6 ВЫВОД 16](#_Toc179439576)

[7 ЛИТЕРАТУРА 17](#_Toc179439577)

# **1 ЦЕЛЬ**

Изучение и реализация механизма быстрого доступа к данным с использованием хеш-таблиц, позволяющего эффективно решать задачи поиска, вставки и удаления элементов с минимальной временной сложностью.

# **2 ЗАДАНИЕ**

## **2.1 Формулировка задачи**

Разработайте приложение, которое использует хеш-таблицу (пары «ключ - хеш») для организации прямого доступа к элементам динамического множества полезных данных. Множество реализуйте на массиве, структура элементов (перечень полей) которого приведена в индивидуальном варианте.

Приложение должно содержать класс с базовыми операциями: вставки, удаления, поиска по ключу, вывода. Включите в класс массив полезных данных и хеш-таблицу. Хеш-функцию подберите самостоятельно, используя правила выбора функции.

Реализуйте расширение размера таблицы и рехеширование, когда это требуется, в соответствии с типом разрешения коллизий.

Предусмотрите автоматическое заполнение таблицы 5-7 записями.

Реализуйте текстовый командный интерфейс пользователя для возможности вызова методов в любой произвольной последовательности, сопроводите вывод достаточными для понимания происходящего сторонним пользователем подсказками.

Проведите полное тестирование программы (все базовые операции, изменение размера и рехеширование), тест-примеры определите самостоятельно. Результаты тестирования включите

в отчет по выполненной работе.

**Вариант №19:**

****

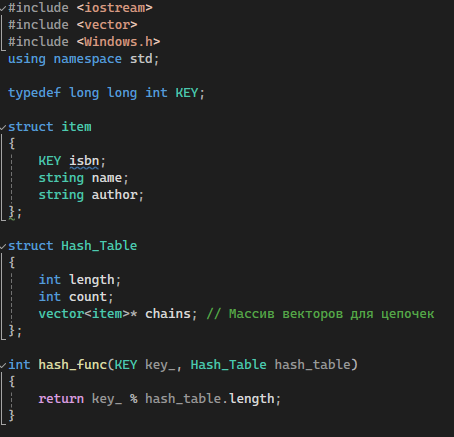
## **2.2 Математическая модель решения**

Хеш-таблица — это структура данных, которая используется для организации **быстрого доступа** к элементам на основе хеш-функции. Основной задачей в данном случае является эффективная вставка, удаление и поиск элементов множества данных. В данной математической модели будет рассмотрен процесс организации множества данных с использованием хеш-таблицы и динамической переалокации при переполнении.:

**Множество данных** — это динамический массив элементов. Каждый элемент множества имеет следующие атрибуты: isbn - ключ типа long long int, который является уникальным идентификатором элемента; name - строка, представляющая имя элемента; author - строка, представляющая автора элемента.

**Хеш-функция**: Хеш-функция — это отображение ключа в индекс массива. Была использована формула h(k) = k mod n, где k – ключ элемента, n – длина хеш-таблицы.

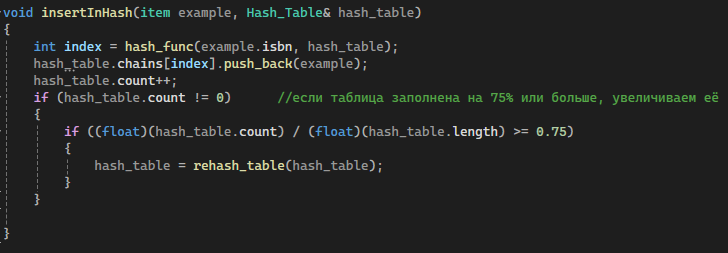
**Хеш-таблица** состоит из 3-х полей: вместимости, количества элементов, и массива векторов для цепного хеширования.



Листинг 1 — Структура записей, структура хеш-таблицы и хеш-функция

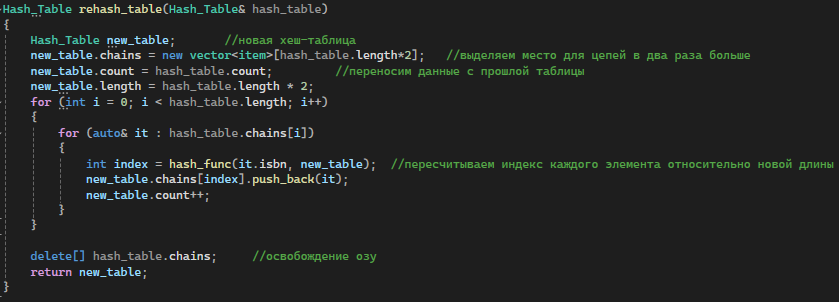
**Цепное хеширование**: В случае возникновения коллизий (одинаковый хеш для нескольких элементов), применяется **цепное хеширование**. Это означает, что в каждом индексе таблицы хранится вектор (динамический массив) элементов, которые имеют одинаковый хеш. Доступ к элементам в каждом векторе осуществляется через последовательный поиск.

**Операция вставки** в хеш-таблицу. При вставке элемента, вычисляется индекс – формируется с помощью хеш-функции. Далее, в массив элементов хеш-таблицы вставляется добавляемый элемент, при чем идет добавление в вектор с индексом рассчитанной хеш-функции.



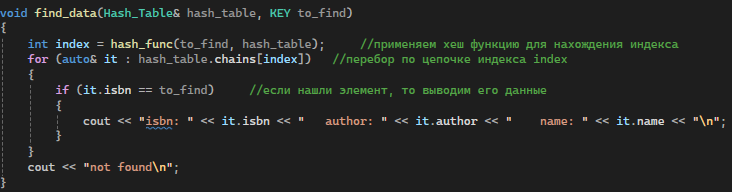
Листинг 2 —Код операции вставки

**Операция рехеширования**. Так как хеш-таблица будет пополнятся различными данными, то на нее будет большая нагрузка. В связи с этим, нужно помнить про условие, что если это коэффициент > 0.75, то следует увеличить вместимость хеш-таблицы в 2 раза.

****

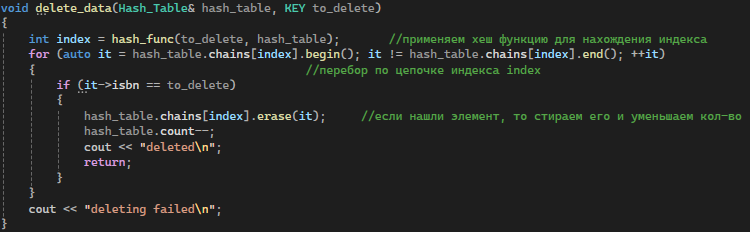
Листинг 3 —Код операции рехеширования

**Операция поиска** элемента. Для нахождения элемента, вычисляется хеш – номер цепочки, в которой лежит элемент. Внутри идет линейный поиск записи – сверяется исходный ключ с ключом записи.



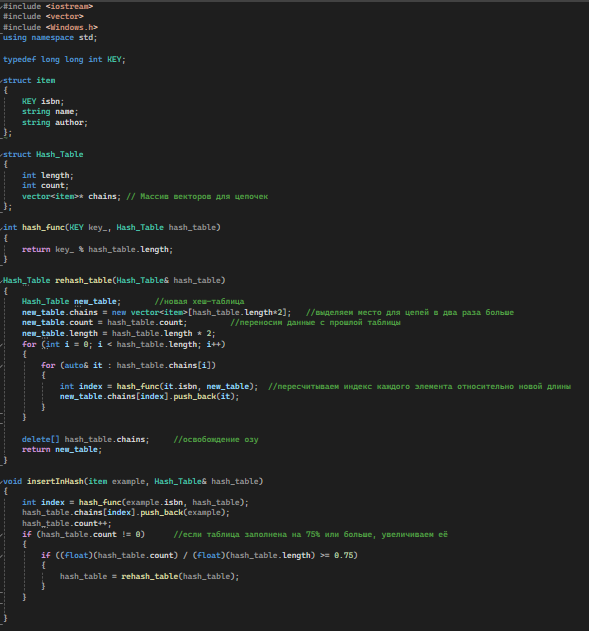
Листинг 4 —Код операции поиска по хеш-таблице

**Операция удаления** элемента. Вычисляется хеш. Находится цепочка в которой лежит запись, по ней линейно ищется необходимая запись, которая в последствии удаляется из вектора.

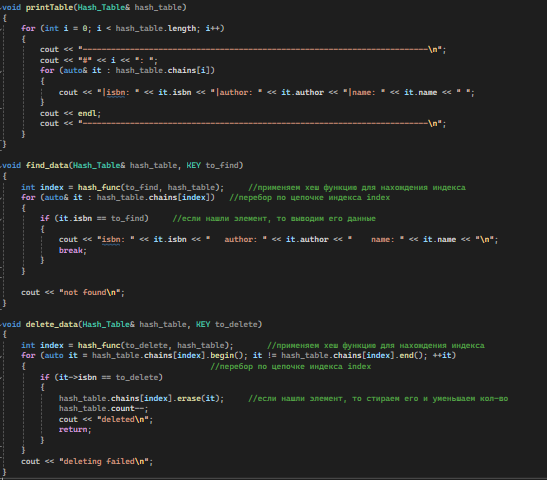


Листинг 5 —Код операции удаления элемента

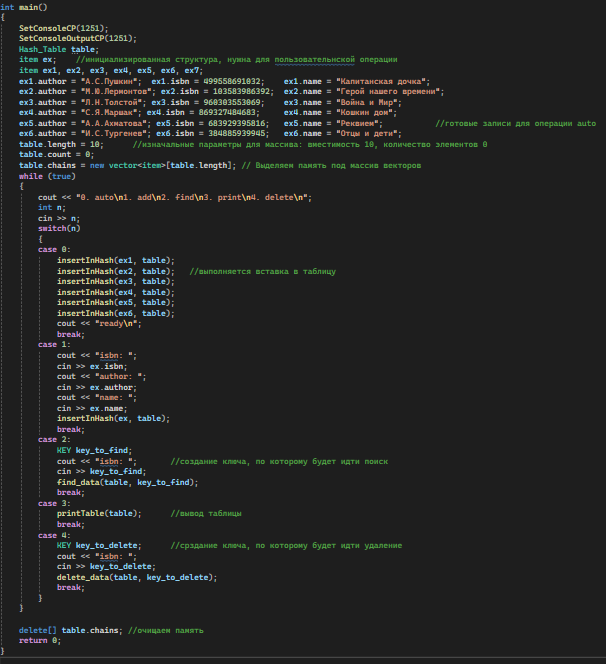
## **2.3 Код программы с комментариями**



Листинг 1 — Код программы



Листинг 2 — Код программы



Листинг 3 — Код программы

## **2.4 Тестирование программы**



Рисунок 1 — Тестирование кода – автоматический ввод данных и вывод



Рисунок 2 — Тестирование кода – добавление элемента, создание коллизии

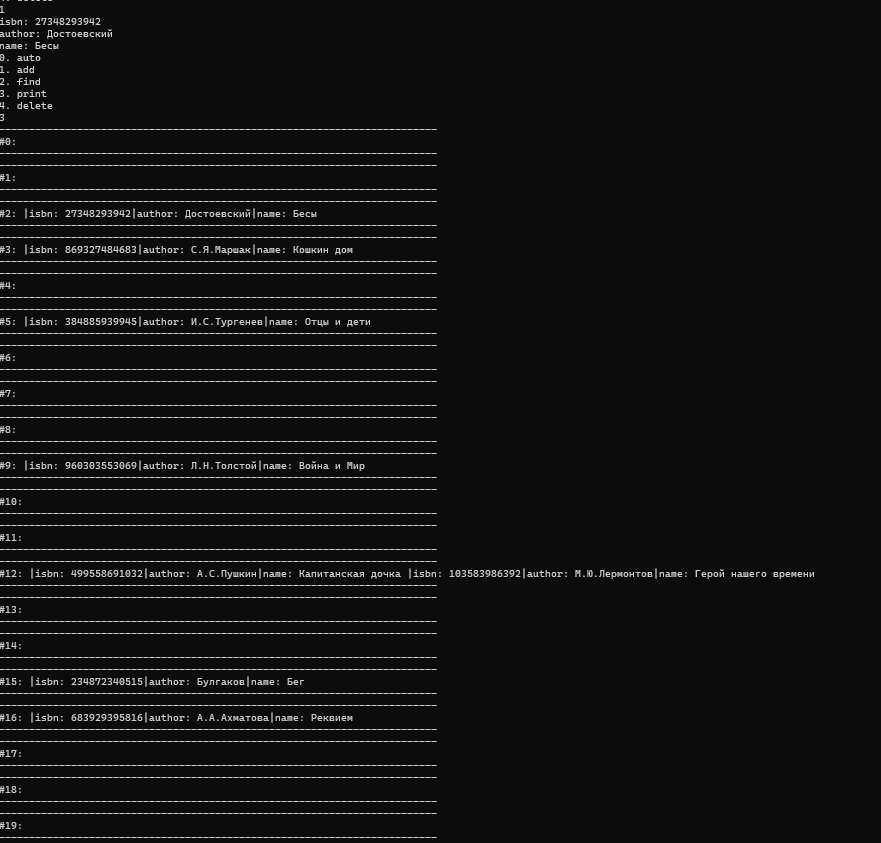


Рисунок 3 — Тестирование кода – создание нагрузки, в последствии - рехеширование

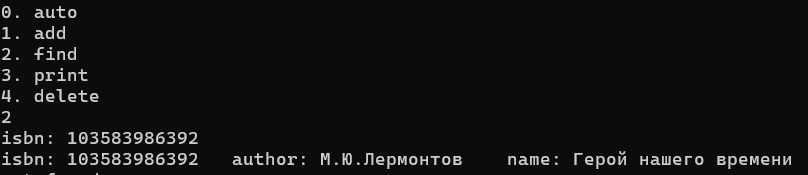


Рисунок 4 — Тестирование кода функции поиска по ключу

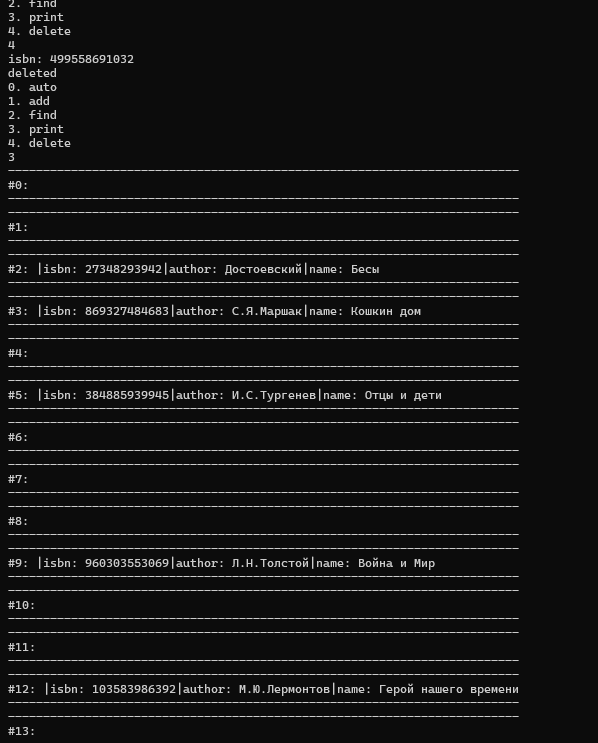


Рисунок 5 — Тестирование кода функции удаления по ключу

\

# **6 ВЫВОД**

В ходе работы была реализована хеш-таблица с использованием метода цепного хеширования на основе векторов для организации быстрого доступа к данным. В процессе разработки были выполнены следующие задачи:

1. Разработана и реализована эффективная хеш-функция, которая использует остаток от деления ключа на размер таблицы, что позволяет равномерно распределять данные по хеш-таблице.
2. Введена структура данных, обеспечивающая хранение ключей и связанных с ними элементов, а также базовые операции для работы с хеш-таблицей: вставка, поиск, удаление и вывод данных.
3. Для обработки коллизий использовано цепное хеширование, где каждая ячейка хеш-таблицы хранит вектор элементов, попавших в одну и ту же корзину. Это позволяет эффективно решать проблему коллизий и поддерживать высокую производительность операций.
4. Реализован механизм перехеширования, который автоматически увеличивает размер хеш-таблицы при достижении порога заполненности (75%). Это обеспечивает баланс между временем вставки элементов и использованием памяти, предотвращая ухудшение производительности при увеличении количества элементов.
5. Операции вставки, поиска и удаления данных имеют амортизированную временную сложность O(1), что делает хеш-таблицы высокоэффективным инструментом для организации быстрого доступа к данным в больших наборах.

Реализованная программа демонстрирует высокую производительность при работе с большими объемами данных и решает задачу организации быстрого доступа к элементам множества. Хеш-таблица является отличным выбором для систем, требующих постоянного доступа к данным за минимальное время.

# **7 ЛИТЕРАТУРА**

1. Вирт Н. Алгоритмы и структуры данных. Новая версия для Оберона, 2010.

2. Кнут Д. Искусство программирования. Тома 1-4, 1976-2013.

3. Бхаргава А. Грокаем алгоритмы. Иллюстрированное пособие для про-граммистов и любопытствующих, 2017.

4. Кормен Т.Х. и др. Алгоритмы. Построение и анализ, 2013.

5. Лафоре Р. Структуры данных и алгоритмы в Java. 2-е изд., 2013.

6. Макконнелл Дж. Основы современных алгоритмов. Активный обуча-ющий метод. 3-е доп. изд., 2018.

7. Скиена С. Алгоритмы. Руководство по разработке, 2011.

8. Хайнеман Д. и др. Алгоритмы. Справочник с примерами на C, C++, Java и Python, 2017.

9. Гасфилд Д. Строки, деревья и последовательности в алгоритмах. Ин-форматика и вычислительная биология, 2003.

По языку С++:

10. Страуструп Б. Программирование. Принципы и практика с использова-нием C++. 2-е изд., 2016.

11. Павловская Т.А. C/C++. Программирование на языке высокого уровня, 2003.

12. Прата С. Язык программирования С++. Лекции и упражнения. - 6-е изд., 2012.

13. Седжвик Р. Фундаментальные алгоритмы на C++, 2001-2002

14. Хортон А. Visual C++ 2010. Полный курс, 2011.

15. Шилдт Г. Полный справочник по C++. 4-е изд., 2006.