

МИНОБРНАУКИ РОСИИ

*Федеральное государственное бюджетное образовательное учреждение высшего образования*

***«МИРЭА – Российский технологический университет»***

**РТУ МИРЭА**



Отчет по выполнению практического задания №6.1

**Тема:**

Быстрый доступ к данным с помощью хеш-таблицДисциплина: «Структуры и алгоритмы обработки данных»

|  |  |  |
| --- | --- | --- |
| Выполнил студент: Васильев Б.А.  Группа: ИКБО-20-23 |  |  |

Москва 2024

**СОДЕРЖАНИЕ**

[ЦЕЛЬ РАБОТЫ 3](#_Toc178190517)

[ХОД РАБОТЫ 3](#_Toc178190518)

[Формулировка задачи 3](#_Toc178190519)

[Описание подхода к решению 4](#_Toc178190520)

[Коды программы 5](#_Toc178190521)

[Результаты тестирования 13](#_Toc178190522)

[ВЫВОД 16](#_Toc178190523)

[СПИСОК ИНФОРМАЦИОННЫХ ИСТОЧНИКОВ 16](#_Toc178190524)

# ЦЕЛЬ РАБОТЫ

Освоить приёмы хеширования и эффективного поиска элементов множества.

# ХОД РАБОТЫ

### 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 (п.3).

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.

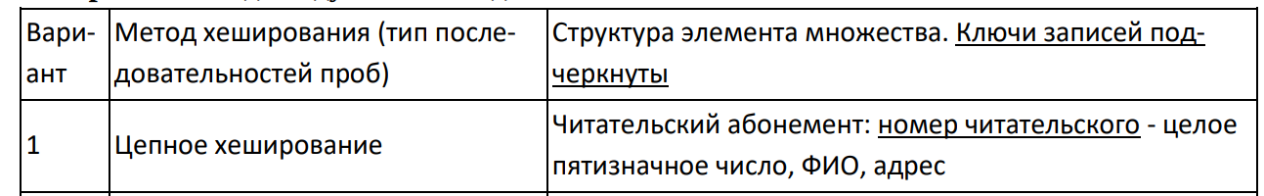


Рисунок 1 – Индивидуальный вариант задачи

### Описание подхода к решению

Подход к решению задачи заключается в использовании хеш-таблицы для организации быстрого доступа к элементам данных. Хеш-таблица строится на основе хеш-функции h вида K mod Q, где K — ключ (номер читательского билета), а M — размер хеш-таблицы. Такая функция распределяет элементы по индексам массива, что минимизирует количество коллизий. Коллизии разрешаются с помощью метода цепочек, при котором несколько элементов могут храниться в одном индексе в виде связанных списков.

По мере заполнения таблицы происходит автоматическое рехеширование, когда она становится заполненной на 75% (коэффициент загрузки 0.75). Это означает, что таблица увеличивается в размерах, и все элементы перераспределяются с использованием обновлённой хеш-функции для более равномерного распределения.

Хеш-таблица хранит указатели на записи, каждая из которых представляет собой элемент данных. При возникновении коллизий, несколько записей могут быть размещены в одном индексе в виде связанного списка, где каждая запись содержит ссылку на следующую. Таким образом, в случае совпадения хеш-значений, записи образуют цепочку, что позволяет хранить несколько элементов в одной ячейке таблицы и эффективно разрешать конфликты при вставке и поиске данных.

### Коды программы

Реализуем код приложения на языке программирования C++ (рис. 2-9)

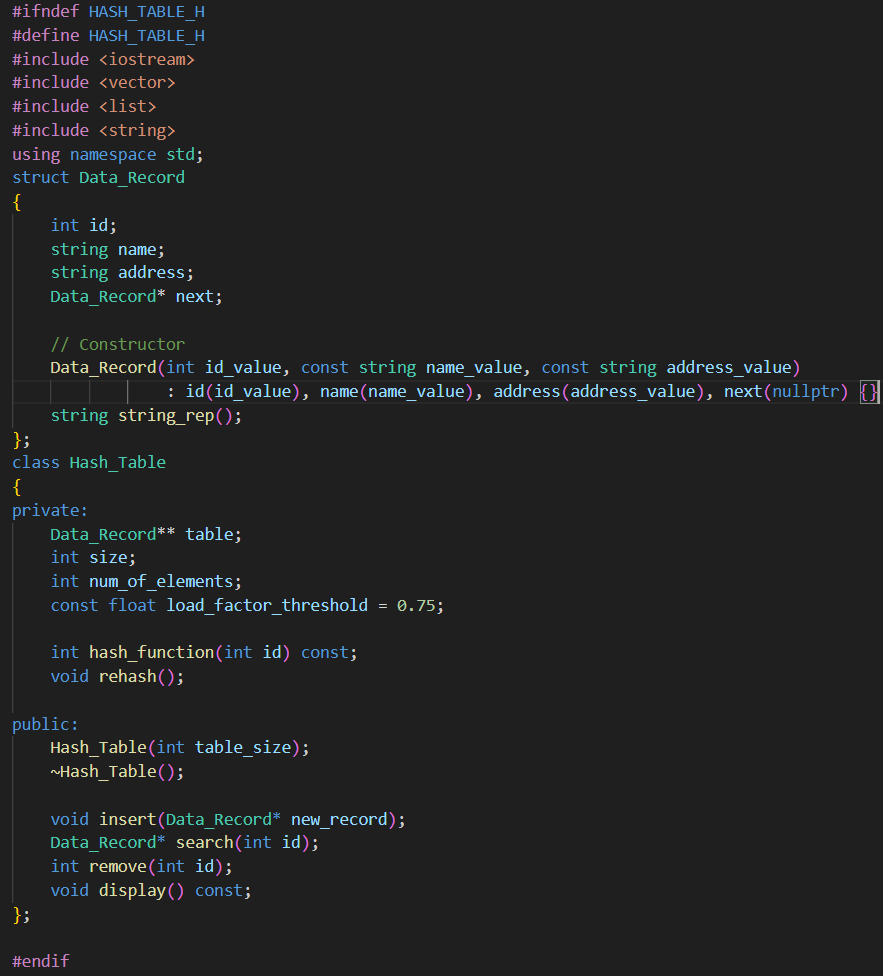


Рисунок 2 – файл Hash\_Table.h

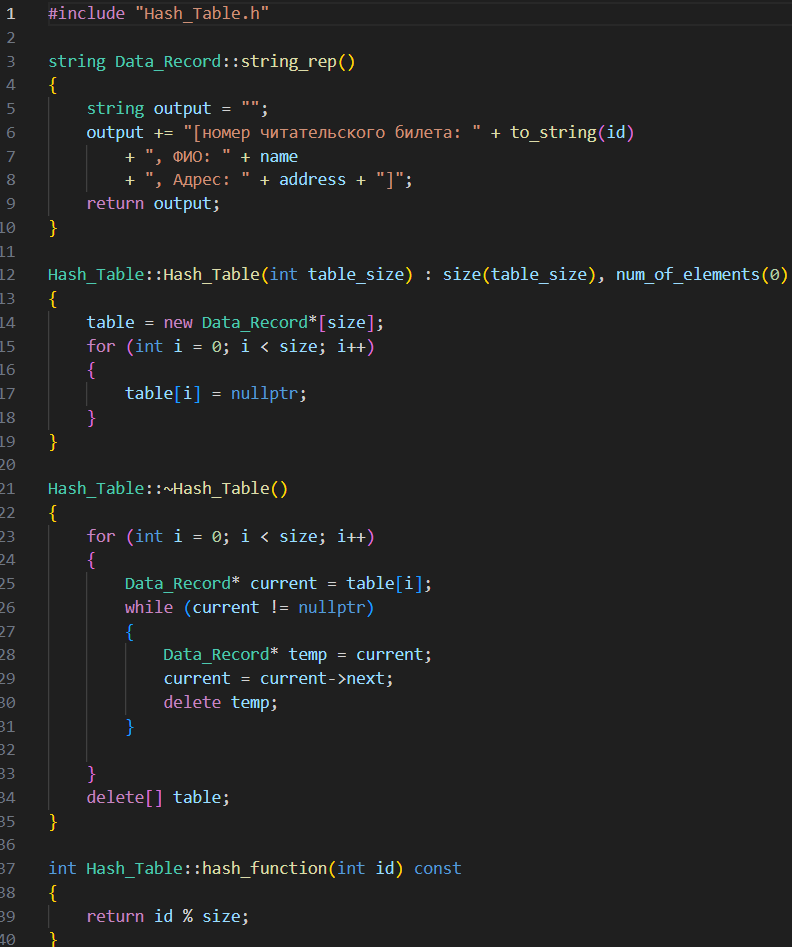


Рисунок 3 – Файл Hash\_Table.cpp (часть 1)

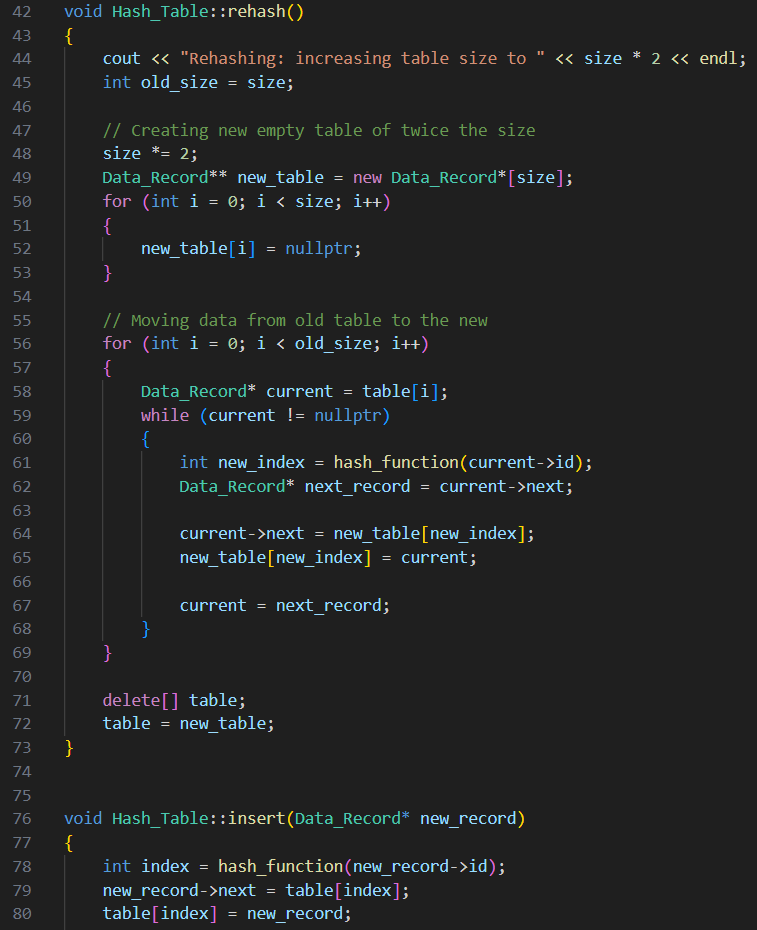


Рисунок 4 – Файл Hash\_Table.cpp (часть 2)

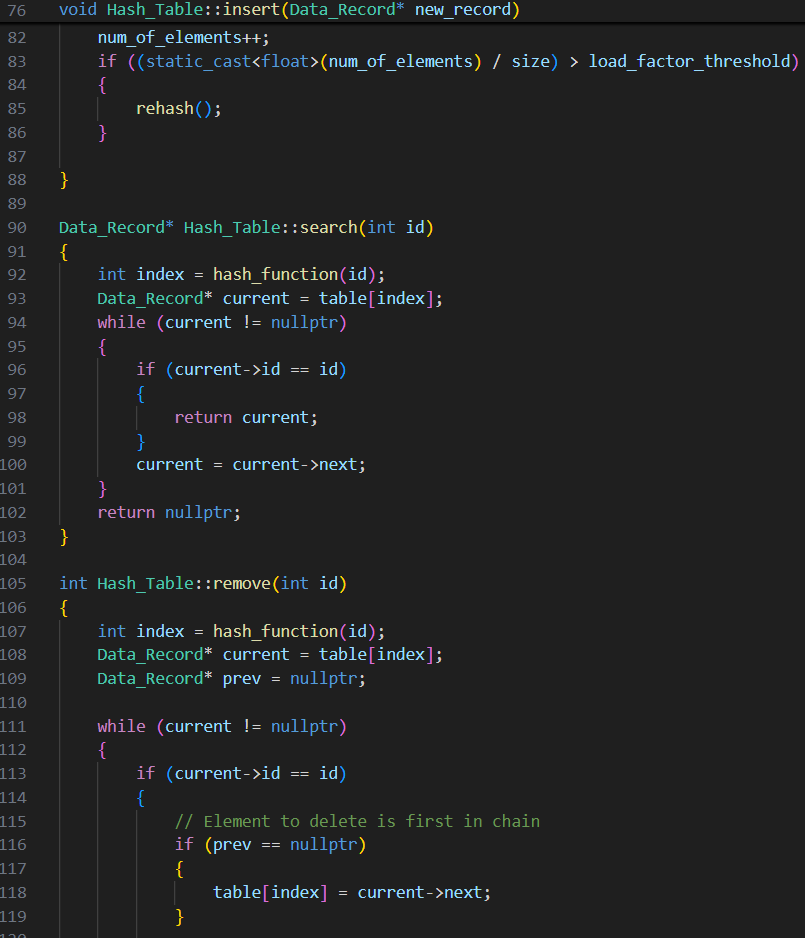


Рисунок 5 – Файл Hash\_Table.cpp (часть 3)

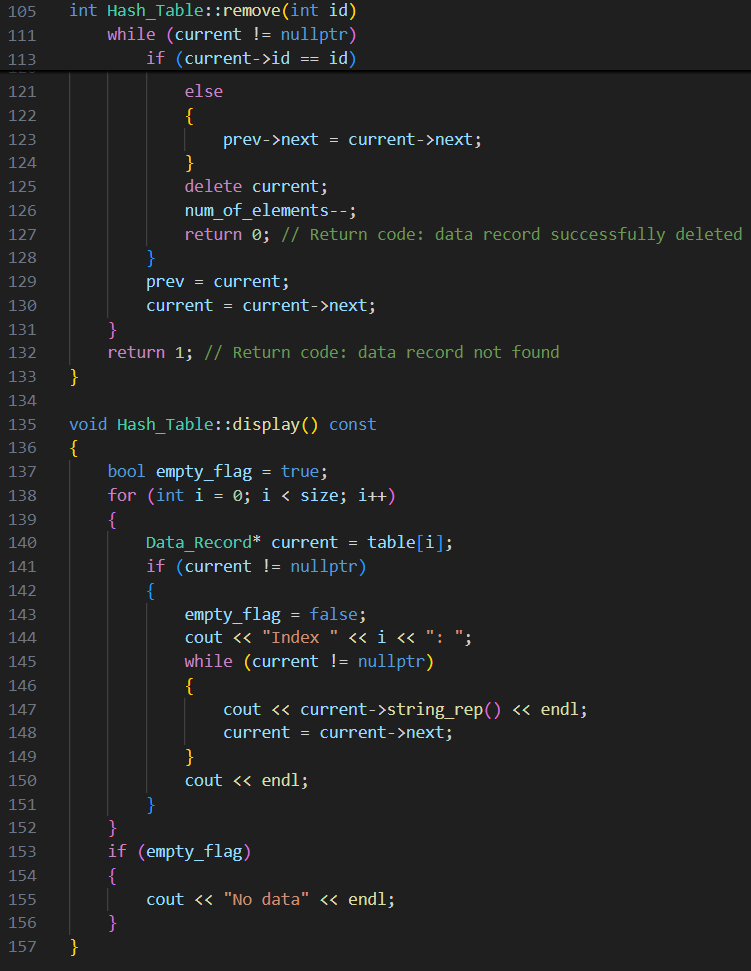


Рисунок 6 – Файл Hash\_Table.cpp (часть 4)

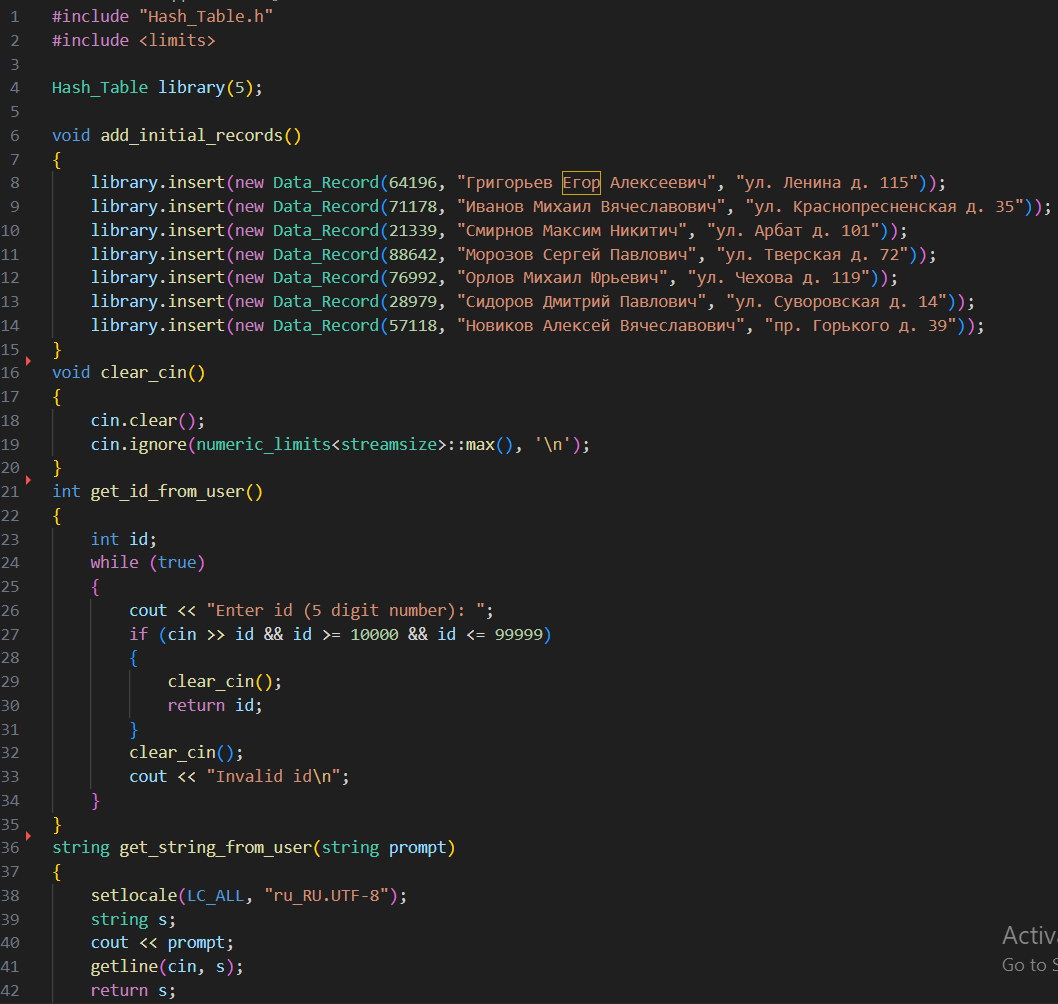


Рисунок 7 – Файл main.cpp (часть 1)

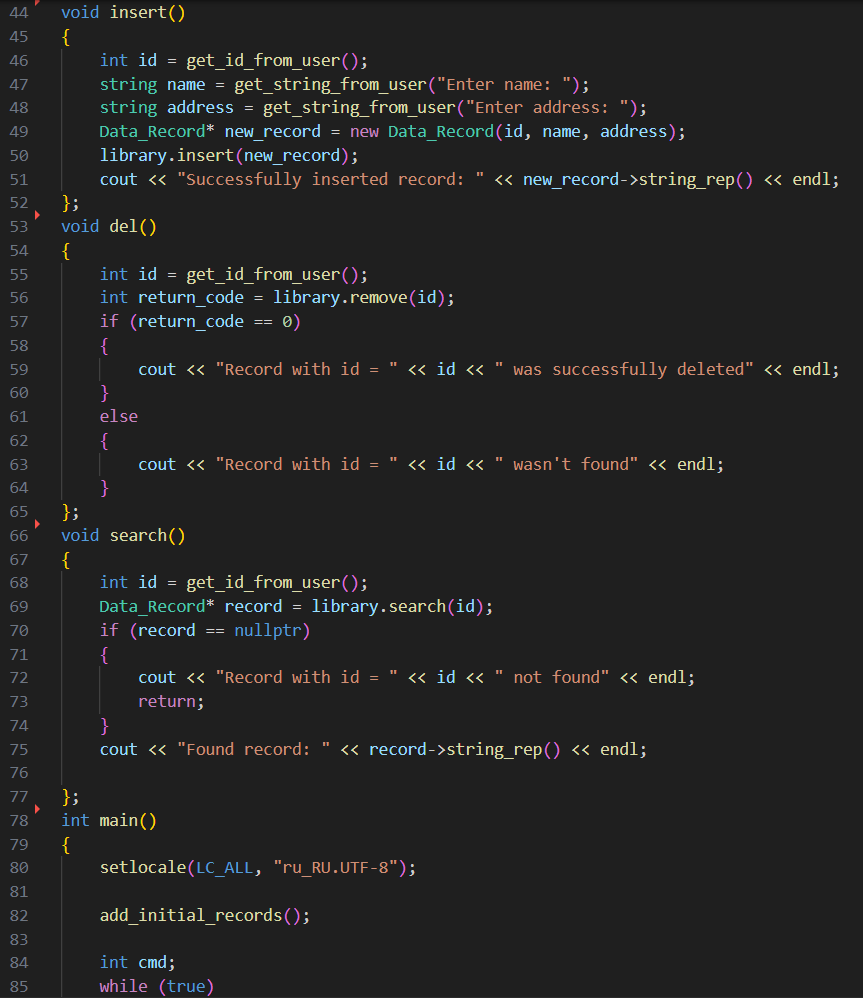


Рисунок 8 – Файл main.cpp (часть 2)

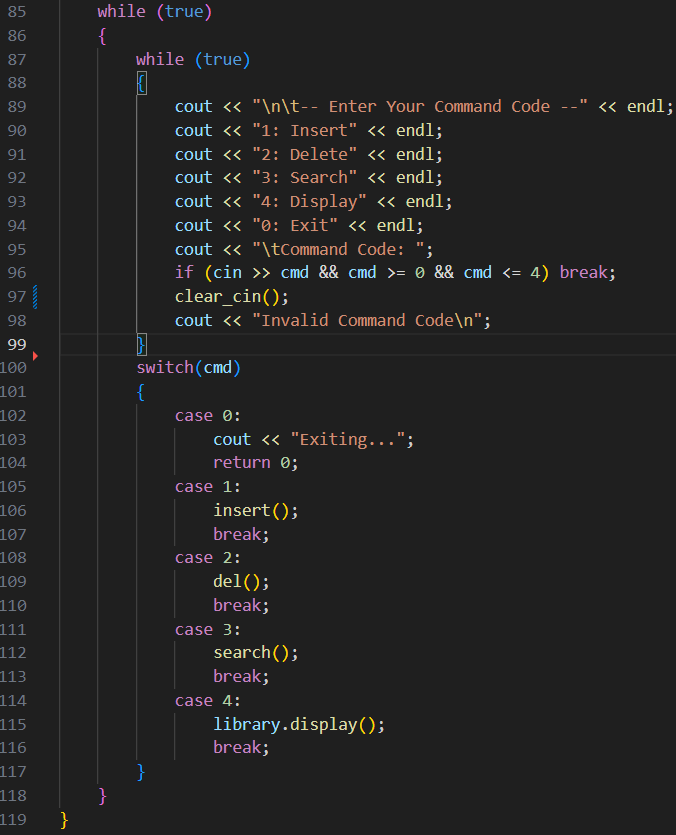


Рисунок 9 – Файл main.cpp (часть 3)

### Результаты тестирования

Выполним тестирование программы (рис. 10-15).

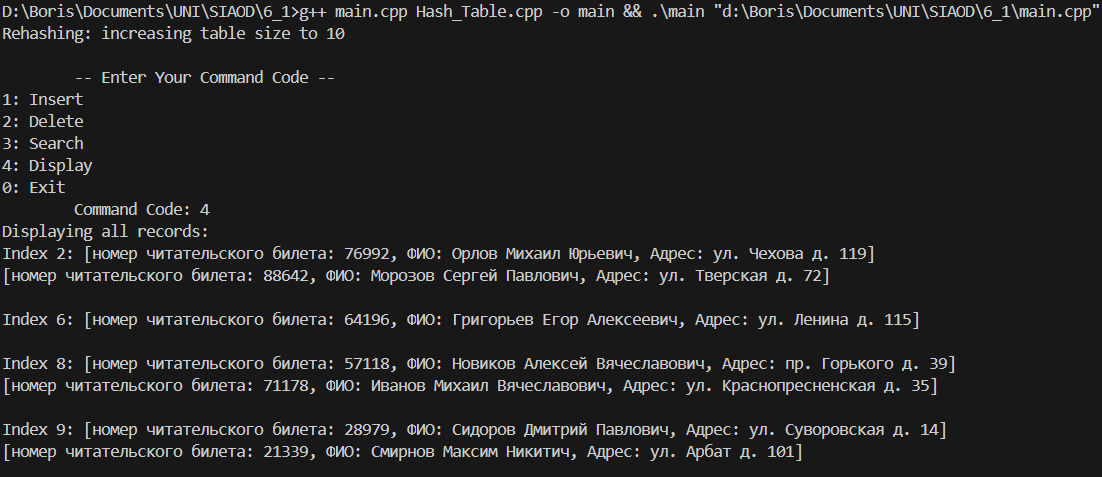
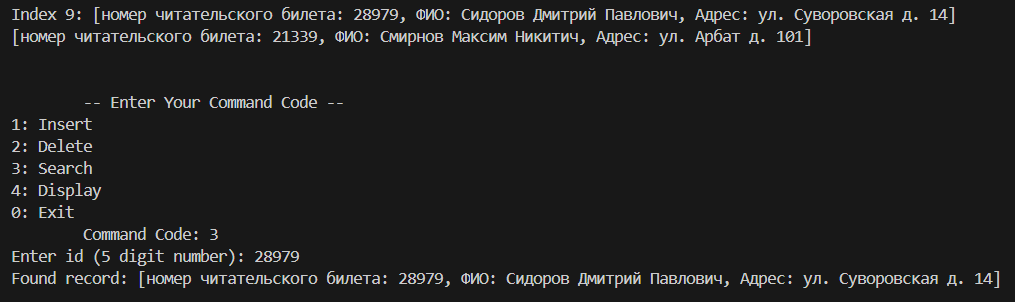
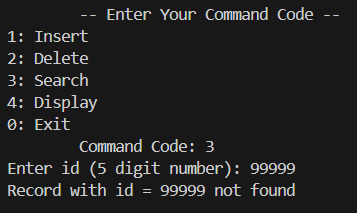
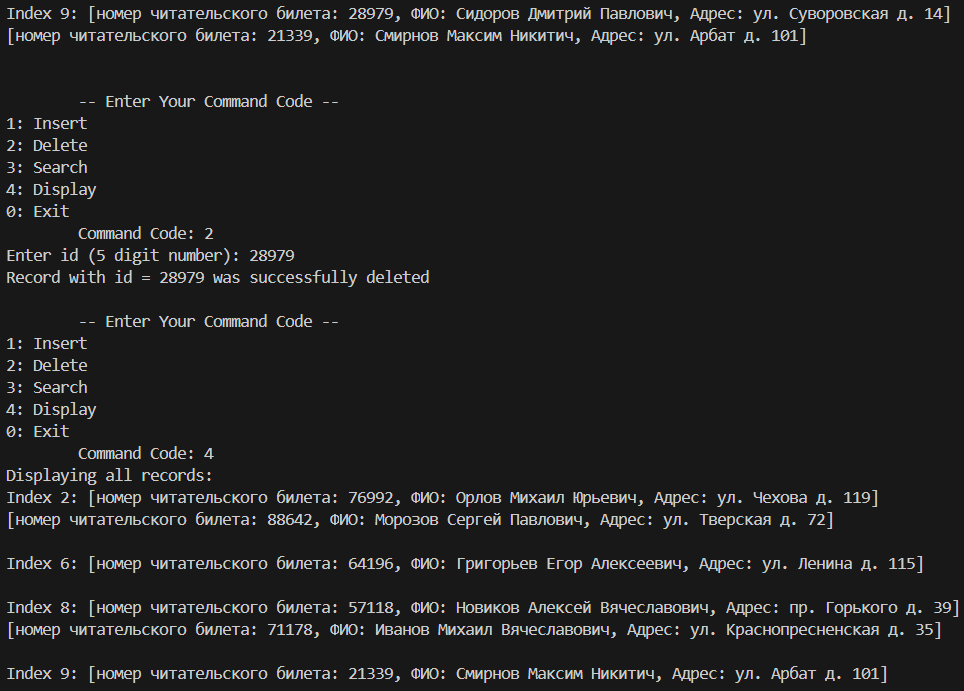
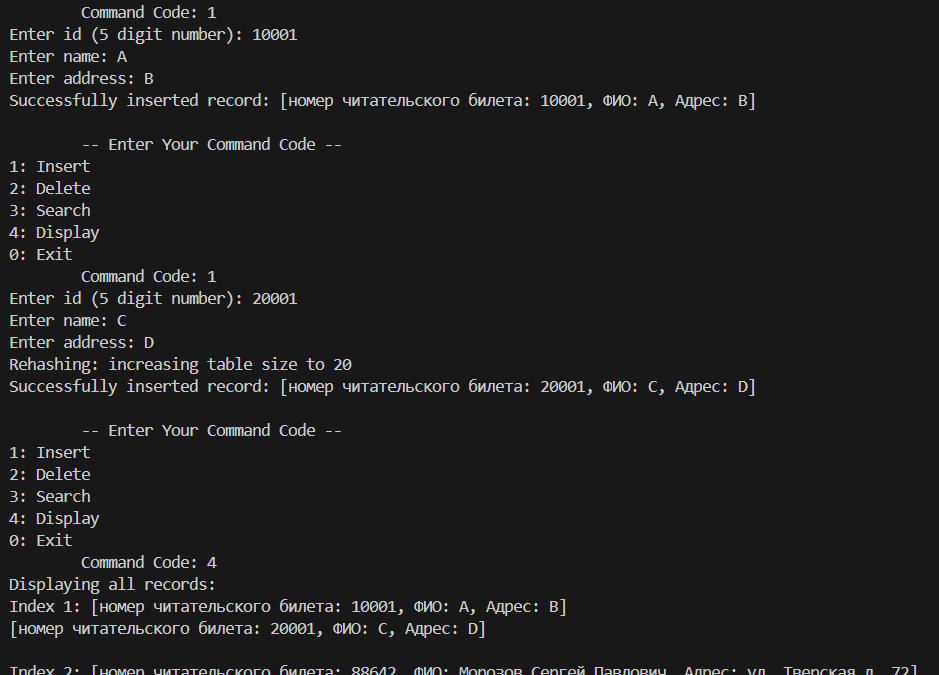


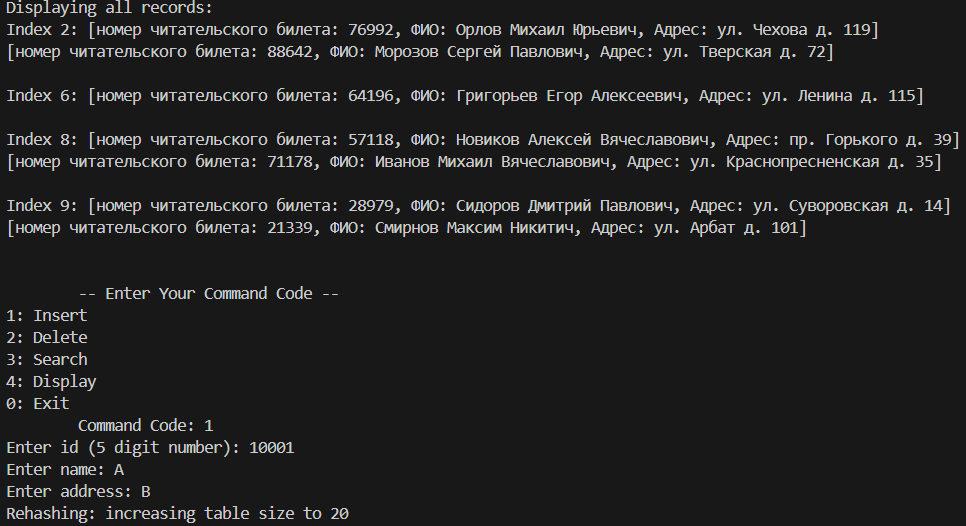
Рисунок 10 – Результат автоматического заполнения семью записями и тестирование вывода всех записей

  
Рисунок 11 – Тестирование поиска по ключу (часть 1)

  
Рисунок 12 – Тестирование поиска по ключу (часть 2)

  
Рисунок 13 – Тестирование удаления

  
Рисунок 14 – Тестирование добавления записей с одинаковым значением хэш-функции

  
Рисунок 15 – Тестирование рехеширования.

Тестирование показало, что программа работает корректно, в том числе в случае возникновения коллизий.

# ВЫВОД

В результате выполнения работы были освоены приёмы хеширования и эффективного поиска элементов множества. Был получен опыт реализации хэш-таблицы с цепным хешированием.

# СПИСОК ИНФОРМАЦИОННЫХ ИСТОЧНИКОВ

1. Рысин, М. Л. Введение в структуры и алгоритмы обработки данных : учебное пособие / М. Л. Рысин, М. В. Сартаков, М. Б. Туманова. — Москва : РТУ МИРЭА, 2022 — Часть 2 : Поиск в тексте. Нелинейные структуры данных. Кодирование информации. Алгоритмические стратегии — 2022. — 111 с. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/310826 (дата обращения: 25.09.2024).
2. Документация по языку С++ [Электронный ресурс]. URL: https://docs.microsoft.com/ruru/cpp/cpp/ (дата обращения 25.09.2024).