

**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ
ФЕДЕРАЦИИ**

**Федеральное государственное автономное образовательное учреждение
высшего образования**

«Национальный исследовательский

**Нижегородский государственный университет им. Н.И. Лобачевского»
(ННГУ)**

Институт информационных технологий, математики и механики

**Кафедра математического обеспечения и суперкомпьютерных
технологий**

ОТЧЕТ ПО УЧЕБНОЙ ПРАКТИКЕ

«Реализация класса Хеш-таблица »

Выполнил: студент группы
381706-2

Крюков Дмитрий Алексеевич

_____ Подпись

Научный руководитель:

ассистент каф. МОСТ ИИТММ

Лебедев Илья Геннадьевич

_____ Подпись

Нижний Новгород

2019

Содержание

Содержание.....	2
1. Введение.....	3
2. Постановка задачи.....	4
3. Руководство пользователя.....	5
4. Руководство программиста.....	6
4.1 Описание структуры программы.....	6
4.2 Описание структур данных.....	6
4.3 Описание алгоритмов.....	7
5. Заключение.....	8
6. Литература.....	9

1. Введение

Таблица – динамическая структура данных. Базисное множество – семейство линейных структур из записей, базисное отношение включения определяется операциями вставки и удаления записей

При поиске данных в таблицах во многих случаях можно предварительно оценить место расположения искомых записей

Функция преобразования значения ключа к номеру (адресу) строки памяти для хранения записи называется функцией (хеширования, перемешивания, рассеивания) расстановки (hash - мешанина, путаница)

Таблицы, представление которых организуется при использовании функции расстановки, называются таблицы с вычислимыми адресами (хеш-таблицы, перемешиваемые таблицы)

Операции под таблицей

Поиск записи по ключу

Вставка новой записи

Удаление записи

2. Постановка задачи

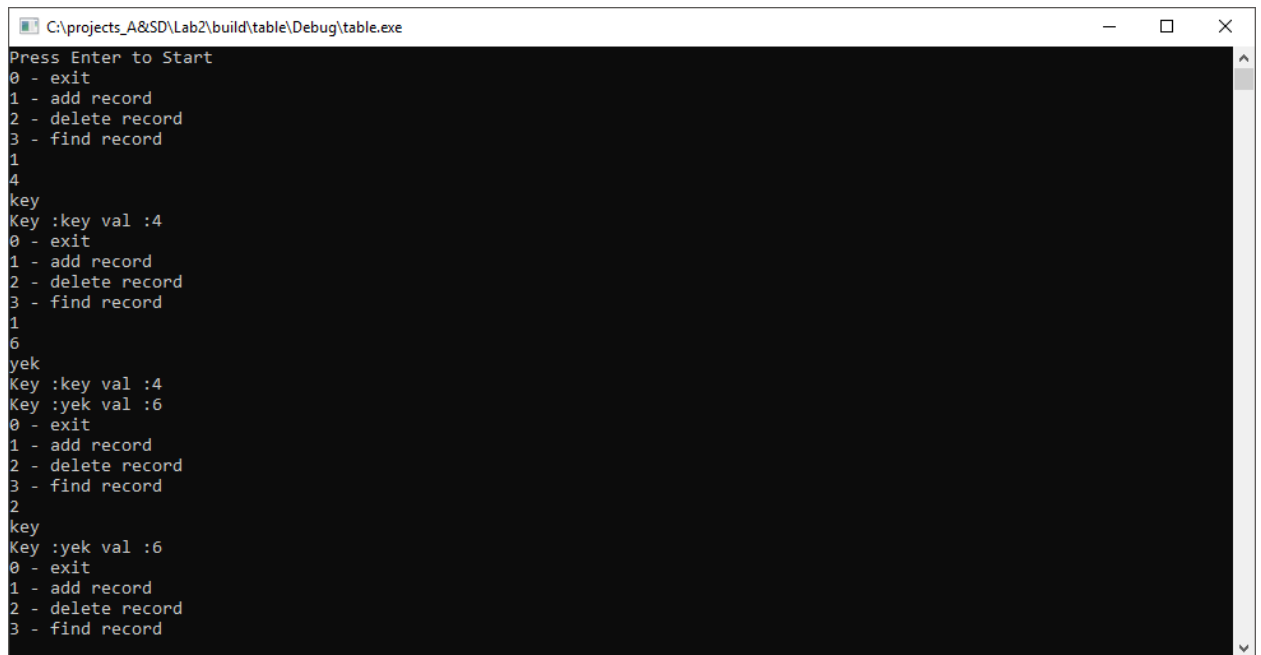
- Организация доступа по имени для управления информацией в привычной для человека форме
- Представление данных во многих задачах из разных областей приложений (таблицы идентификаторов, номенклатура изделий, словари и т.п.)
- Абстрагирование от проблем распределения памяти при размещении данных
- Отображение на ЭВМ такого важного математического понятия как *множества*
- Организация прямого доступа при помощи функции расстановки

Требования к функции расстановки (редкое возникновение коллизий)

- Быстрое вычисление
- Равномерное распределение имен
- Равномерное распределение часто используемых имен
- Равномерное распределение близких имен

3. Руководство пользователя

Данная программа предназначена для тестирования динамической структуры хэш-таблица



```
C:\projects_A&SD\Lab2\build\table\Debug\table.exe
Press Enter to Start
0 - exit
1 - add record
2 - delete record
3 - find record
1
key
Key :key val :4
0 - exit
1 - add record
2 - delete record
3 - find record
1
6
yey
Key :yey val :6
0 - exit
1 - add record
2 - delete record
3 - find record
2
key
Key :yey val :6
0 - exit
1 - add record
2 - delete record
3 - find record
```

пользователю предлагается набор команд :

- вставка записи в таблицу
- удаление записи из таблицы
- поиск записи в таблице

4. Руководство программиста

4.1 Описание структуры программы

1. Модуль hashtablelib (THashTable.h, TListHash.h) – реализация класса хэш-таблица методом открытого перемешивания и методом цепочек
2. Модуль hashtable(main.cpp) – реализация программы для тестирования динамической структуры хэш-таблица
3. Модуль hashtabletest(hashtable_test.cpp, listhash_tests.cpp) - тестирование класса хэш-таблица при помощи Google C++ Testing Framework.

4.2 Описание структур данных

Структура

Поля:

dataCount — число записей в таблице

tabSize — максимальное число записей в таблице

pRecs – массив записей таблицы

hashStep — шаг хэш-функции

Методы:

HashFunc(TKey k) — хэш функция

GCD(int n, int m) — наибольший общий делитель

Add(TTabRecord<ValType> *tr) — добавляет запись в таблицу

Delete(TKey k) — удаляет запись из таблицы

Search(TKey k) — поиск по ключу

operator[] (TKey k) — обращение по ключу, в случае отсутствия записи добавляет пустую запись с заданным ключом

4.3 Описание алгоритмов

Поиск

1. $s = h(\text{key})$ // применение функции расстановки
2. ЕСЛИ s занята и $K[s] == \text{key}$, ТО { Останов }
3. ЕСЛИ s свободна, ТО { Останов }
4. (!) Коллизия $\{ s = (s+p) \bmod M$ и переход к п. 2 }

Удаление

1. Поиск записи
2. ЕСЛИ запись найдена,
3. ТО { Отметить строку как пустую }

5. Заключение

В ходе работы был реализованы классы хэш-таблица с разрешением коллизий методом открытого перемешивания и методом цепочек , в них реализованы функции удаления, вставки и поиска, а так же добавлена индексация по ключу

6. Литература

1. Гергель В.П. Методические материалы по курсу «Методы программирования 2», Нижний Новгород, 2015.