МИНОБРНАУКИ РОССИИ

Федеральное государствен

ное бюджетное образовательное учреждение

высшего образования

**«Челябинский государственный университет»**

**(ФГБОУ ВО «ЧелГУ»)**

Институт информационных технологий

Кафедра информационных технологий и экономической информатики

ОТЧЕТ

по лабораторной работе №6

Авторы отчета С.М. Панов ПрИ-201

подпись инициалы, фамилия группа

А.Д. Казбеков ПрИ-201

подпись инициалы, фамилия группа

В.И. Кочетков ПрИ-201

подпись инициалы, фамилия группа

Отчет защищен \_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_

дата оценка

Челябинск 2024 г.

**Цель работы:** понять и реализовать хеш-таблицы, а так же реализовать методы вычисления хеш значений.

Стек технологий:

1. C#

**Задание 1**

I

Реализовать

1. хеш-таблицу, использующую метод разрешения коллизий с помощью цепочек. (см. Рис. 1.1.1)

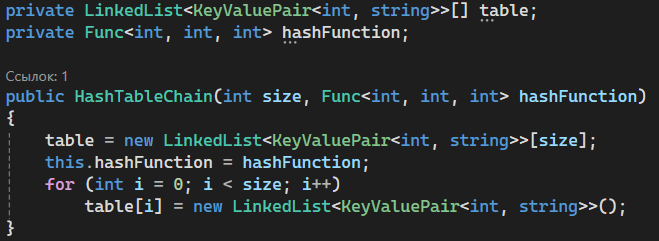


Рис. 1.1.1 Код реализации хеш-таблицы, используя метод разрешения коллизий с помощью цепочек

1. интерфейсные операции работы с таблицей  
   (Поиск элемента, Вставка элемента, Удаление элемента) (см. Рис. 1.1.2 – 1.1.4)

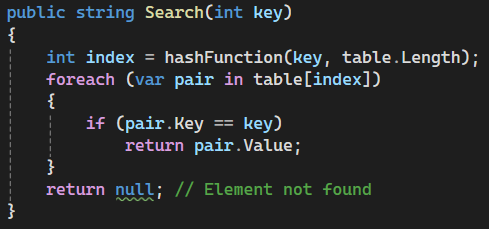


Рис. 1.1.2 Код поиска элемента

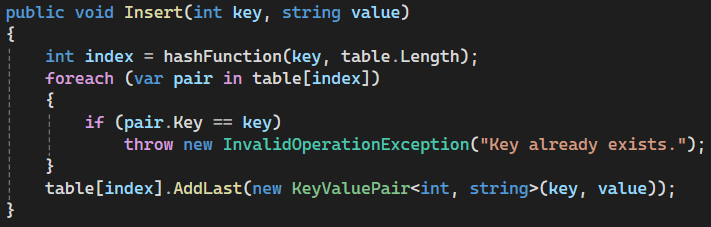


Рис. 1.1.3 Код вставки элемента

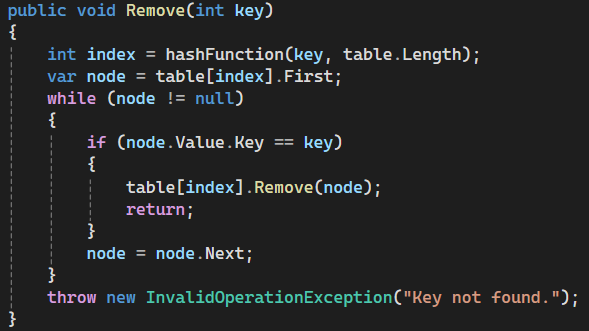


Рис. 1.1.4 Код удаления элемента

Методы вычисления хеша

* Метод деления (см Рис. 1.1.5)
* Метод умножения (см Рис. 1.1.6)
* XOR + Побитовый сдвиг (см Рис. 1.1.7) (Обеспечивает достаточно хорошее распределение и уменьшает кол-во коллизий)

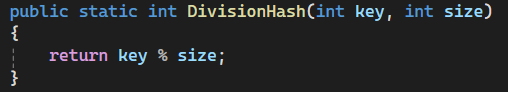


Рис. 1.1.5 Код метода деления

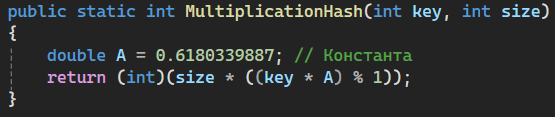


Рис. 1.1.6 Код метода умножения

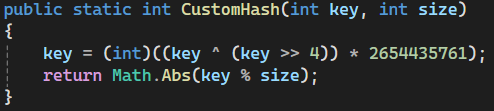


Рис. 1.1.7 Код своего метода вычисления хеша

II

После генерации 100000 элементов и вставки их в хеш-таблицу размером 1000, получаем следующие данные (см. Рис. 1.2.1)

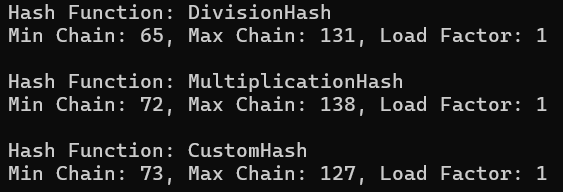


Рис. 1.2.1 Результат вставки 100000 элементов в хеш таблицу

Так же были созданы плохие методы вычисления хеша (см. Рис. 1.2.2), в результате (см. Рис. 1.2.3) получаем очень маленькое значение коэффициента заполнения

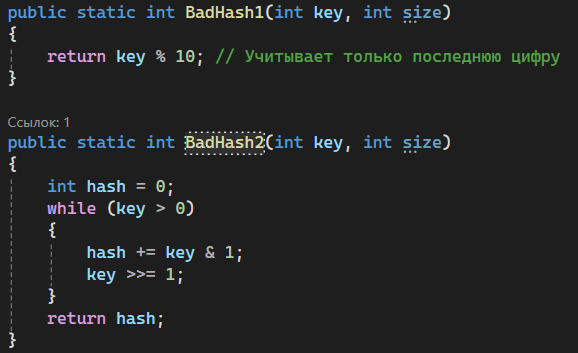


Рис. 1.2.2 Код плохих методов вычисления хеша

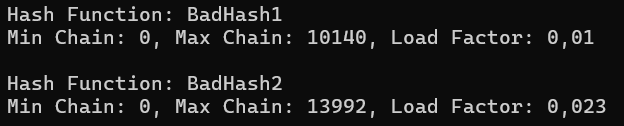


Рис. 1.2.3 Результат плохих методов вычисления хеша

Для первых трёх функций результаты примерно одинаковые и выявить кто из них лучше трудно.

**Задание 2**

I

Реализовать

1. хеш-таблицу, использующую метод открытой адресации для разрешения коллизий. (см. Рис. 2.1.1)

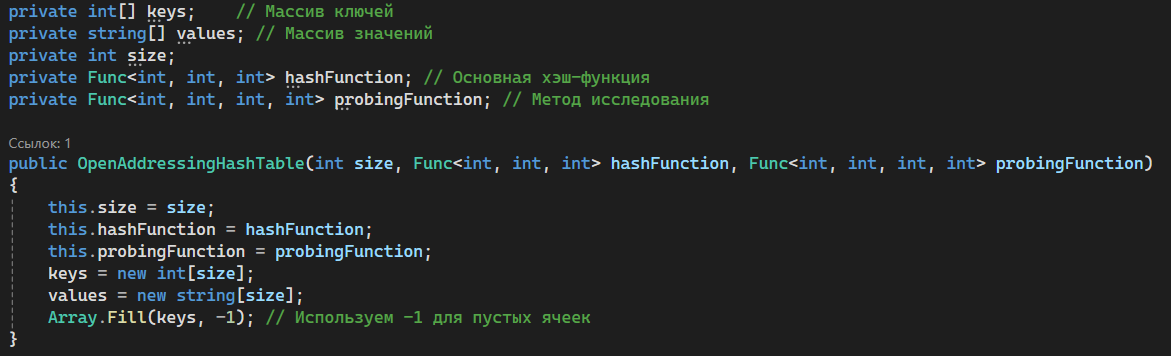


Рис. 2.1.1 Код хеш-таблицы, используя метод открытой адресации

1. интерфейсные операции работы с таблицей   
   (Поиск элемента, Вставка элемента, Удаление элемента). (см. Рис. 2.1.2 – 2.1.4)

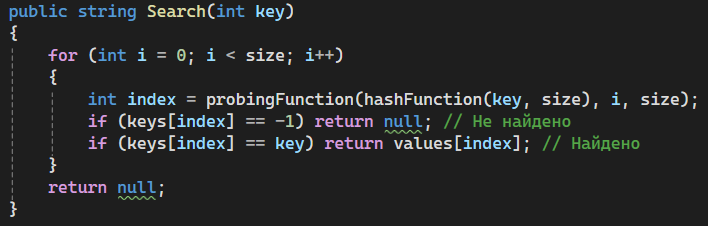


Рис. 2.1.2 Код поиска элемента

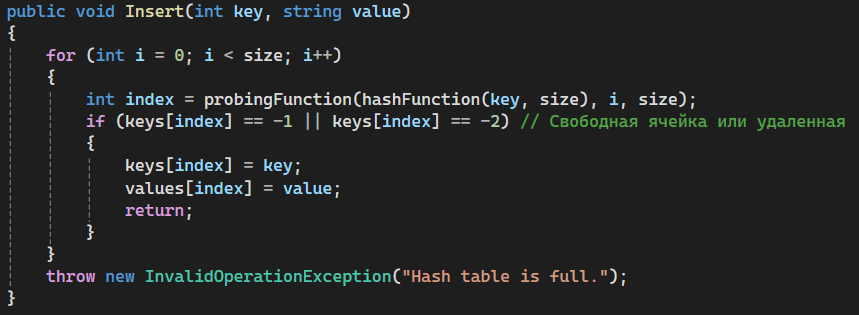


Рис. 2.1.3 Код вставки элемента

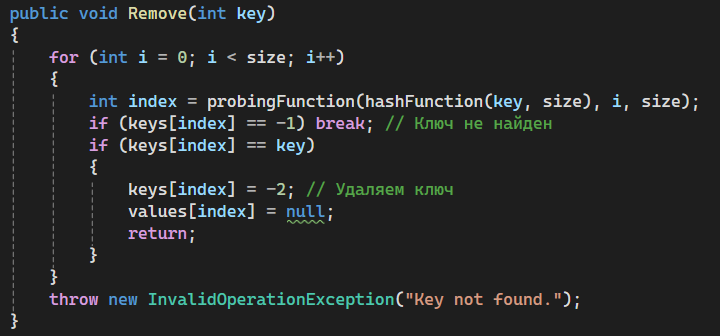


Рис. 2.1.4 Код удаления элемента

Методы исследования (см. Рис. 2.1.5):

* линейное исследование
* квадратичное исследование
* двойное хеширование
* модифицированное квадратичное исследование
* побитовое смешивание

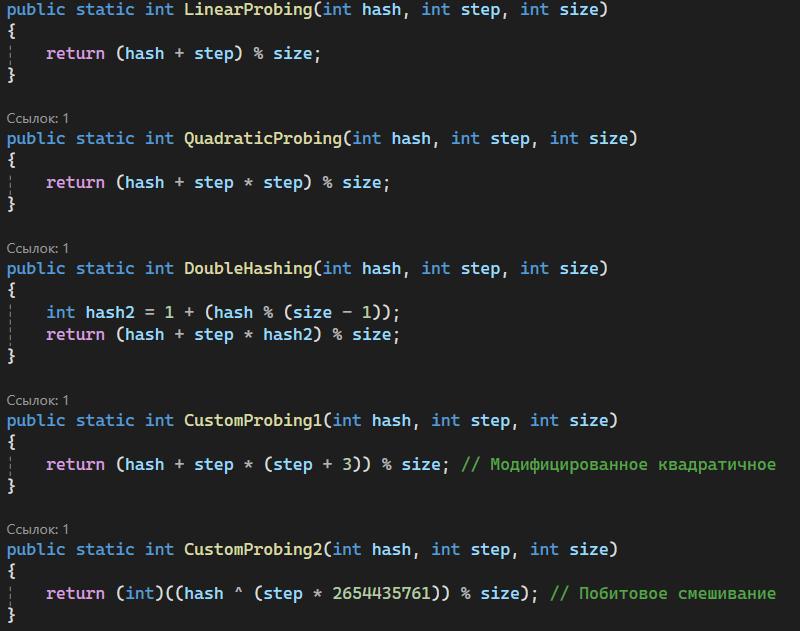


Рис. 2.1.5 Реализация методов исследования

II

После генерации 10000 элементов и вставки их в хеш-таблицу размером 10000, получаем следующие данные (см. Рис. 2.2.1)

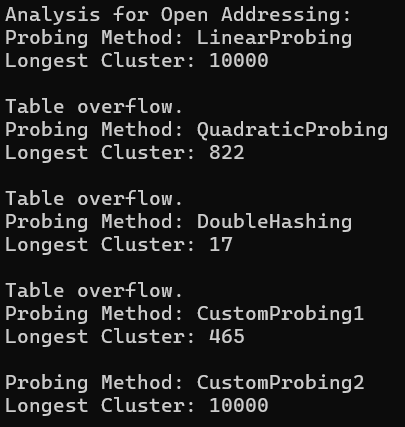


Рис. 2.2.1 Результат вставки 10000 элементов в хеш-таблицу

Линейное исследование и побитовое смешивание приводят к самой сильной кластеризации, что негативно влияет на производительность.  
Проведя несколько запусков получаем, что модифицированное квадратичное исследование имеет примерно в 2 раза меньшую кластеризацию исследование в отличии от обычной квадратичной

Двойное хеширование обеспечивает наилучшее распределение и минимальную кластеризацию, по сравнению с остальными методами и является самым лучшем методом.