СПб НИУ ИТМО

кафедра ИПМ

Алгоритмы и структуры данных

Лабораторная работа № 1

Хеш-функция

Вариант 5

Рехеширование с использованием случайных чисел

Работу выполнил:

Студент II курса

Группы № 2120

Журавлев Виталий

Санкт-Петербург

2014 г.

**Цель работы:**

Разработать программу, реализующую комбинированный способ организации таблицы идентификаторов. Для организации таблицы используется простейшая хэш-функция (**Сумма кодов первой и второй букв**), а при возникновении коллизий используется дополнительный метод размещения идентификаторов в памяти (**Рехеширование с использованием случайных чисел**). Если в качестве этого метода используется дерево или список, то они должны быть связаны с элементом главной хэш-таблицы. Программа должна сообщать среднее число коллизий и среднее количество сравнений, выполненных для поиска идентификатора.

**Описание метода размещения:**

**Метод рехеширования с использованием случайных чисел**

Метод организации таблиц идентификаторов, основанный на использовании хеш-адресации, заключается в помещении каждого элемента таблицы в ячейку, адрес которой возвращает хэш-функция, вычисленная для этого элемента.

Согласно этому методу, если для элемента A адрес h(A), вычисленный с помощью хеш-функции h, указывает на уже занятую ячейку, то необходимо вычислить значение функции n1=h1(A) и проверить занятость ячейки по адресу n1. Если и она занята, то вычисляется значение h2(A) и так до тех пор, пока либо не будет найдена свободная ячейка, либо очередное значение hi(A) совпадет с h(A). В последнем случае считается, что таблица идентификаторов заполнена, и места в ней больше нет — выдается информация об ошибке размещения идентификатора в таблице.

**Описание хэш-функции:**

В данной работе в качестве простейшей хэш-функции используется сумма кодов символом первой и второй буквы слова. Возвращает эта функция целое число из определенного диапазона (зависит от кодировки). Стоит отметить, что если в слове нет двух и более букв, т.е. слово состоит из одного символа, то необходимо вывести его код.

**Код программы**

using System;

using System.IO;

using System.Collections.Generic;

namespace alg\_lab1

{

 class Program

 {

 static void Main(string[] args)

 {

 Dictionary<Int64, String> HashTable = new Dictionary<Int64, String>();

 string filename = @"D:\Programming\C#\Projects\Alg\_lab1\hashfile.txt";

 string[] Words;

 string word;

 Int64 u = 0, Coll\_num = 0, Comp\_num = 0, Key, Key\_new;

 Int32 table\_size = 300;

 Words = File.ReadAllText(filename).Split(new[] { ' ', '.', ',' }, StringSplitOptions.RemoveEmptyEntries);

 Int32[] Rndm\_nums = new Int32[Words.Length];

 Random random = new Random();

 for (int i = 0; i < Words.Length; i++)

 {

 Rndm\_nums[i] = random.Next(1, table\_size);

 }

 for (int i = 0; i < table\_size; i++)

 { HashTable.Add(i, ""); }

 for (int i = 0; i < Words.Length; i++)

 {

 word = Words[i];

 Key = HashFunc(word);

 if (HashTable[Key] == "")

 {

 Comp\_num++;

 HashTable[Key] = word;

 }

 else

 {

 Coll\_num++;

 mark:

 Comp\_num++;

 Key\_new = ((Key + Rndm\_nums[u]) % table\_size);

 if (HashTable[Key\_new] == "")

 {

 HashTable[Key\_new] = word;

 }

 else

 {

 if (Key\_new == Key)

 {

 Console.WriteLine("Закончилось место!");

 }

 else

 {

 Key = Key\_new;

 u++;

 goto mark;

 }

 }

 }

 }

 Console.WriteLine("Коллизий : {0}\n", Coll\_num.ToString());

 Console.WriteLine("Сравнений: {0}\n", Comp\_num.ToString());

 Console.ReadLine();

 }

 static Int64 HashFunc(string word)

 {

 if (word.Length >= 2)

 {

 return Convert.ToInt64(word[0]) + Convert.ToInt64(word[1]);

 }

 else

 {

 return Convert.ToInt64(word);

 }

 }

 }

}

**Вывод:**

В ходе выполнения лабораторной работы была организована таблица идентификаторов с помощью заданной простейшей хэш-функции по заданному методу размещения (метод рехеширования с использованием случайных чисел). Этот метод размещения нельзя признать достаточно удачным, так как эффективность сильно зависит от заполненности таблицы идентификаторов и может возникнуть такая ситуация, когда не окажется ни одной свободной ячейки, адреса которых вычислены по заданной хэш-функции, тогда как в действительности в таблице идентификаторов может оказаться много пустых мест.