МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Ижевский государственный технический университет имени М. Т. Калашникова»

Факультет «Информатика и вычислительная техника»

Кафедра «Программное обеспечение»

Алгоритмы и структуры данных

Лабораторная работа №3

«Хеш-таблица»

Вариант № 1

Выполнил:

Студент группы Б03-191-2з

Закиров Айдар Ринатович

Проверил:

Старший преподаватель

[Еланцев Михаил Олегович](https://istu.ru/staff/1092)

Ижевск 2020

**Задача:**

Реализовать хеш-таблицу с открытым хешированием.

Хэш-таблица должна поддерживать следующие операции:

Ключом хэш-таблицы является целочисленное значение

1. Инициализация

2) Вставка элемента (целочисленный ключ + значение)

3) Получение значения элемента по ключу

4) Удаление элемента по ключу

5) Освобождение памяти от структуры данных

**Решение:**

Для реализации хеш-тфблицы была составлена программа на языке программирования C#:

**Исходный код программы:**

using System;

using System.Collections;

namespace HashTable

{

class Program

{

//Инициализируем хеш таблицу

static Hashtable hashtable = new Hashtable();

//Метод добавления нового ключа и элемента

static void AddNewKeyAndElemToHashTable()

{

Console.Write("Введите через пробел знаение Ключа(должен быть целочисленным) и Элемента: ");

string[] str = Console.ReadLine().Split(' ');

hashtable.Add(Convert.ToInt32(str[0]), str[1]);

}

//Получение значения элемента по ключу

public static void PrintElemутеByItsKey()

{

Console.WriteLine("Введите значение ключа");

var key = Convert.ToInt32(Console.ReadLine());

if (hashtable.ContainsKey(key) == true) Console.WriteLine($"Элемент с ключем {key}: {hashtable[key]}");

else

{

Console.WriteLine("Элемента с таким ключем несуществует! Попробуйте ещё раз.");

PrintElemутеByItsKey();

}

}

//Удаление элемента по ключу

static void RemoveElemутеByItsKey()

{

Console.WriteLine("Введите значение ключа");

var key = Convert.ToInt32(Console.ReadLine());

if (hashtable.ContainsKey(key))

{

hashtable.Remove(key);

Console.WriteLine($"Элемент с ключем {key} удален!");

}

else

{

Console.WriteLine("Элемента с таким ключем несуществует! Попробуйте ещё раз.");

RemoveElemутеByItsKey();

}

}

static void ClearToHashtable()

{

Console.WriteLine("Занято памяти до очистки: " + GC.GetTotalMemory(false));

hashtable.Clear();

GC.Collect();

Console.WriteLine("Занято памяти после очистки: " + GC.GetTotalMemory(true));

}

static void Comands()

{

int comand = Convert.ToInt32(Console.ReadLine());

switch (comand)

{

case 0:

AddNewKeyAndElemToHashTable();

Comands();

break;

case 1:

RemoveElemутеByItsKey();

Comands();

break;

case 2:

PrintElemутеByItsKey();

Comands();

break;

case 3:

ClearToHashtable();

break;

default:

Console.WriteLine("Вы выбрали неизвестную команду ппробуйте ещё раз!");

Comands();

break;

}

}

static void Main(string[] args)

{

Console.WriteLine("Добавить новый элемент в таблицу - введите цифру 0\n" +

"Удалить элемент по его ключу - введите циру 1\n" +

"Вывести в консоль значение элемента по его ключу - введите цифру 2\n" +

"Очистить хеш таблицу - нажмите 3");

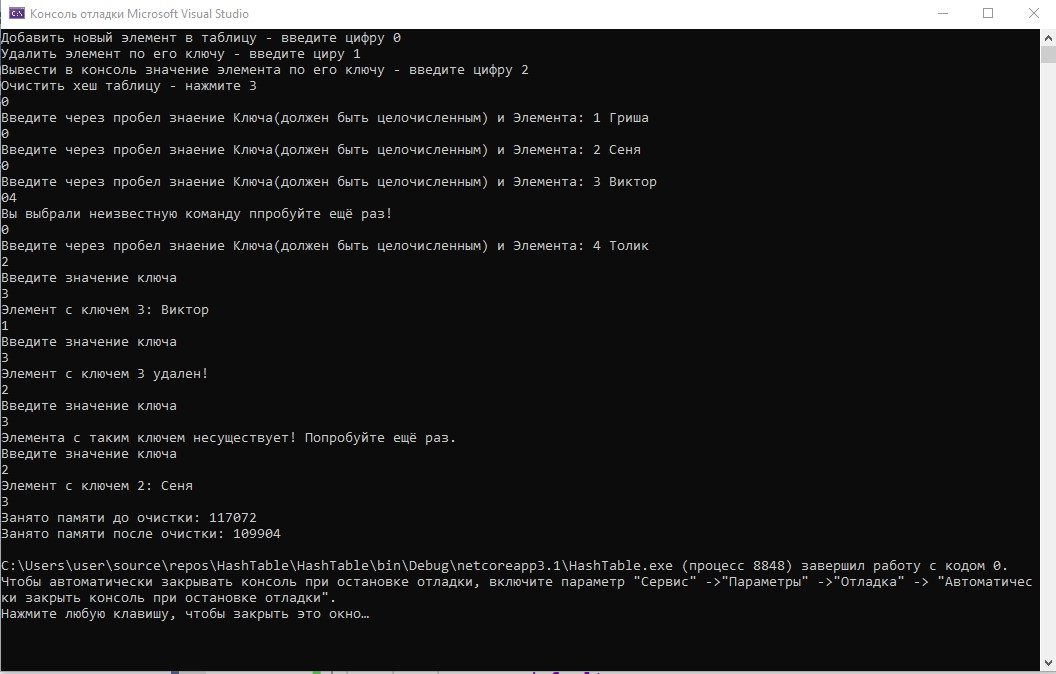
Comands();

}

}

}

**Вывод программы:**



После запуска программа выводит список команд для проведения нужных операций со структурой данных.

Вводим значение «0» для вызова метода AddNewKeyAndElemToHashTable() добавляющего новый элемент с ключом в хеш-таблицу. Вводим данные 4 раза:

ключ «1» значение «Гриша»

ключ «2» значение «Сеня»

ключ «3» значение «Виктор»

ключ «4» значение «Толик»

Проверяем наличие элемента в списке вводя команду «2», данная команда запускает метод PrintElemутеByItsKey(), который выводит в консоль значение элемента по его ключу который вводит пользователь. Вводим значение ключа «3». Элемент с ключом «3» имеет значение «Виктор».

С помощью команды «1» вызываем метод RemoveElemутеByItsKey(), который осуществляет поиск и удаление элемента с указанным нами ключом из списка, в данном случае это элемент с ключом «3» и значением «Виктор». Затем проверяем наш список для того что бы убедится, что элемент удалён, получаем сообщение о том, что такого элемента не существует.

Вводим команду «3» запускающую метод ClearToHashtable(). Данный метод вызывает сборщик мусора, показывает количество занятых программой байт перед очисткой и после удаления данных хеш-таблицы.

**Вывод:**

В данной лабораторной работе были изучены методы работы с хеш-таблицами. Реализованы согласно варианту такие структуры как Хэш-таблица с открытым хешированием.