|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | |  |  | | МИНОБРНАУКИ РОССИИ | | | | Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | | |  | | | | | |
|  | | |
|  | Институт информационных технологий (ИТ) |
|  | Кафедра математического обеспечения и стандартизации информационных технологий (МОСИТ) |

|  |  |
| --- | --- |
| **ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №5** | |
| **по дисциплине** | |
| **«Структуры и алгоритмы обработки данных»**  **По теме: «Хеширование для организации быстрого поиска данных»**  **Вариант 7(23)** | |
|  | |
| Выполнил студент группы ИВБО-06-17 | Ушакова А.С. |
| Принял преподаватель | Скворцова Л.А. |

Москва 2019

**Оглавление**

[1. Задание 1. 12](#_Toc26288846)

[1.1. Реализация хеш функции 12](#_Toc26288848)

[1.2. Таблица тестов 12](#_Toc26288859)

[1.3. Текст исходного кода (листинг) программы 13](#_Toc26288860)

[Листинг программы представлен в приложении 1. 13](#_Toc26288861)

[2. Задание 2. 13](#_Toc26288862)

[2.1. Вариант задания 13](#_Toc26288865)

[2.2. Структура класса 13](#_Toc26288866)

[2.3. Таблица тестов 15](#_Toc26288915)

[2.4. Текст исходного кода (листинг) программы 17](#_Toc26288916)

[3. Контрольные прогоны программы 17](#_Toc26288918)

[Вывод 19](#_Toc26288924)

[Приложение 1 19](#_Toc26288926)

1. Задание 1.

1. Разработайте шаблонный класс – хеш таблица, для поиска информации в коллекциях данных с помощью хеш функции. Хеш функцию подберите самостоятельно.

* 1. Реализация хеш функции

int hashFunc(tKey str){

//сумма чисел в коде специальности

int asc = 0;

for (int i=0; i<str.length();i++){

if(isdigit(str[i])){

asc +=str[i];

}

}

return asc % SIZE;

}

* 1. Таблица тестов

|  |  |  |
| --- | --- | --- |
| **Номер теста** | **Входные данные** | **Эталон результата** |
| **1** | 102 | 7 |
| **2** | 111 | 7 |
| **3** | 001 | 5 |

* 1. Текст исходного кода (листинг) программы

Листинг программы представлен в приложении 1.

1. Задание 2.

Разработайте приложение, которое использует хеш таблицу для организации прямого доступа к элементам множества (массив данных), структура элементов которого приведена в варианте. Множество реализуйте через класс - шаблон с операциями вставки, удаления, поиска, вывода и включите в него хеш таблицу. Предусмотрите возможность создания наследников хеш-таблицы.

Выведите список индексов, которые формируются при вставке элементов в таблицу.

* 1. Вариант задания

|  |  |
| --- | --- |
| Метод хеширования (способ реализации коллизий) | Структура элемента множества(ключ – подчеркнутое поле) |
| Цепное хеширование | Специализация вуза: код специальности, название вуза |

* 1. Структура класса

template<typename tKey, typename tValue>

class Hash\_Table{

template<typename Key, typename Value>

struct Univer{

Univer(Value n= 0, Key c=0,Univer\* nxt=nullptr):name(n),code(c),next(nxt){}

Univer \*next;

Value name;

Key code;

};

Univer<tKey,tValue> \*table[SIZE];

//хеш функция

int hashFunc(tKey str){

//сумма чисел в коде специальности

int asc = 0;

for (int i=0; i<str.length();i++){

if(isdigit(str[i])){

asc +=str[i];

}

}

return asc % SIZE;

}

/\* Univer\* head;

Univer\* tail;

\*/

public:

//иницилизация

Hash\_Table(){

for (int i=0; i<SIZE; i++){

table[i] = NULL;

}

}

//удаление

~Hash\_Table(){

for(int i=0;i<SIZE;i++){

delete table[i];

}

}

//вставка

void push(tValue name, tKey code);

//поиск по коду специализации

void findEl();

//удаляет по ключу

void del(tKey elem);

//вывод

void output();

//вывод индексов

void Indexes();

};

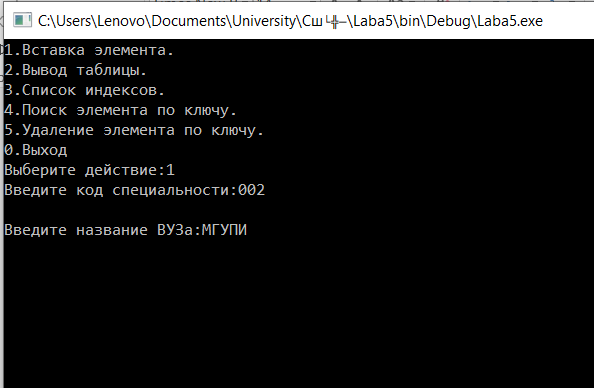
* 1. Таблица тестов

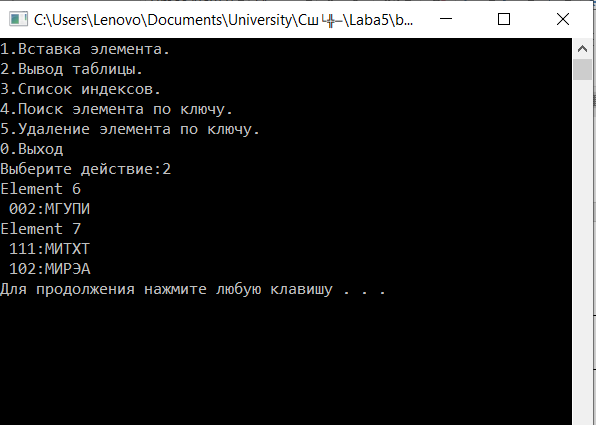
|  |  |  |
| --- | --- | --- |
| **Номер теста** | **Входные данные** | **Эталон результата** |
| **1** | 1. 002 МГУПИ  111 МИТХТ  102 МИРЭА  2. 002  3. 111 | Вывод таблицы  Element 6  002:МГУПИ  Element 7  111:МИТХТ  102:МИРЭА  Список индексов  6 7 7  Поиск элемента по ключу  ВУЗ:МГУПИ  Удаление элемента по ключу  Element 6  002:МГУПИ  Element 7  102:МИРЭА |
| **2** | 1. 002 МГУПИ  102 МИТХТ  201 МИРЭА  2. 201  3. 002 | Вывод таблицы  Element 6  002:МГУПИ  Element 7  102:МИТХТ  201:МИРЭА  Список индексов  6 7 7  Поиск элемента по ключу  ВУЗ:МИРЭА  Удаление элемента по ключу  Element 7  102:МИТХТ  201:МИРЭА |

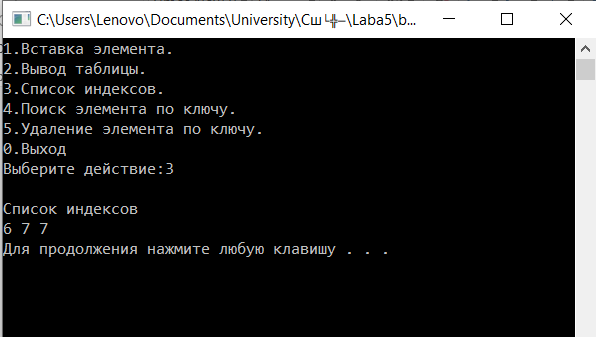
* 1. Текст исходного кода (листинг) программы

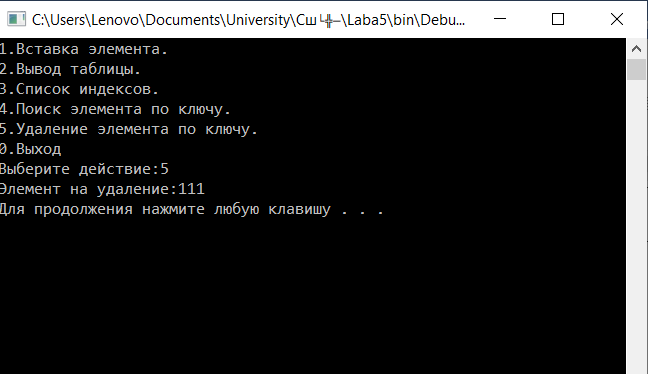
Листинг программы представлен в приложении 1.

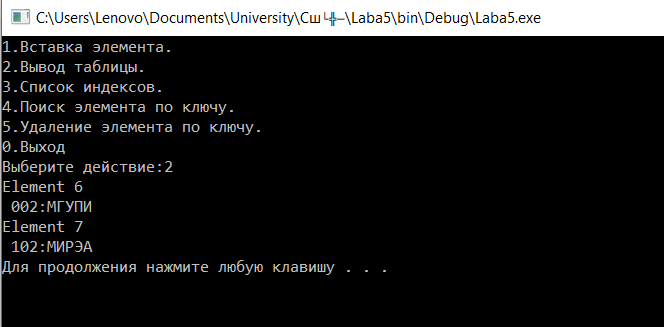
1. Контрольные прогоны программы











Вывод

В результате выполнения лабораторной работы были получены навыки по созданию и работе с хеш функциями и хеш таблицами.

Приложение 1

Hash\_Table.h

#ifndef HASH\_TABLE\_H

#define HASH\_TABLE\_H

#include <iostream>

#include <string>

#include <ctype.h>

#define SIZE 28

using namespace std;

template<typename tKey, typename tValue>

class Hash\_Table{

template<typename Key, typename Value>

struct Univer{

Univer(Value n= 0, Key c=0,Univer\* nxt=nullptr):name(n),code(c),next(nxt){}

Univer \*next;

Value name;

Key code;

};

Univer<tKey,tValue> \*table[SIZE];

//хеш функция

int hashFunc(tKey str){

//сумма чисел в коде специальности

int asc = 0;

for (int i=0; i<str.length();i++){

if(isdigit(str[i])){

asc +=str[i];

}

}

return asc % SIZE;

}

/\* Univer\* head;

Univer\* tail;

\*/

public:

//иницилизация

Hash\_Table(){

for (int i=0; i<SIZE; i++){

table[i] = NULL;

}

}

//удаление

~Hash\_Table(){

for(int i=0;i<SIZE;i++){

delete table[i];

}

}

//вставка

void push(tValue name, tKey code);

//поиск по коду специализации

void findEl();

//удаляет по ключу

void del(tKey elem);

//вывод

void output();

//вывод индексов

void Indexes();

};

//ВСТАВКА

template<typename tKey, typename tValue>

void Hash\_Table<tKey,tValue>::push(tValue name, tKey code){

//получение хеш кода

int index = hashFunc(code);

//создание нового элемента

Univer<tKey,tValue>\* univ = new Univer<tKey,tValue>(name, code);

//элемент хеш-таблицы

Univer<tKey,tValue>\* place = table[index];

if(!place){

table[index] = univ;

table[index]->next = nullptr;

return;

}

while(place->next){

place=place->next;

}

place->next = univ;

}

//ВЫВОД

template<typename tKey, typename tValue>

void Hash\_Table<tKey,tValue>::output(){

Univer<tKey,tValue>\* ptr;

for(int i=0;i<SIZE;i++){

if(table[i]){

ptr=table[i];

cout<<"Element "<<i<<" "<<endl;

while(ptr){

cout<<" "<<ptr->code<<":"<<ptr->name<<endl;

ptr=ptr->next;

}

}

}

}

//ПОИСК по ключу

template<typename tKey, typename tValue>

void Hash\_Table<tKey,tValue>::findEl(){

Univer<tKey,tValue>\* ptr;

string num;

cout<<"\nВведите код специальности:\n";

cin>>num;

int index = hashFunc(num);

ptr=table[index]; //указатель на текущий элемент

if(ptr){

while(ptr && (ptr->code != num) ){

ptr=ptr->next;

}

if(ptr){

cout<<"ВУЗ:"<<ptr->name<<endl;

} else{

cout<<"Element not found\n";

return;

}

} else{

cout<<"Element not found\n";

return;

}

}

//УДАЛЕНИЕ

template<typename tKey, typename tValue>

void Hash\_Table<tKey,tValue>::del(tKey elem){

Univer<tValue, tKey>\*pt, \*pt1 = nullptr, \*pt2 = nullptr;

int index = hashFunc(elem);

pt=table[index];

pt1=pt;

while(pt->code!=elem && pt){

pt=pt->next;

}

//элемент в начале

if(pt && pt1==pt){

pt1=pt->next;

delete pt;

table[index] = pt1;

return;

}

//элемент не найден

if(!pt){

cout<<"Error\n";

return;

}

//элемент в конце

if(!pt->next){

Univer<tValue,tKey>\* temp2 = pt1;

//поиск указателя на предыдущий элемент

while(temp2 -> next && temp2->next !=pt){

temp2=temp2->next;

}

table[index] = pt1; //указатель на предыдущий элемент

free(pt);

table[index]->next =0;

}

//элемент в середине

else{

Univer<tValue,tKey>\* temp1 = table[index];

while(temp1 -> next && temp1->next != pt){

temp1=temp1->next;

}

pt1=temp1;

pt1->next = pt->next;

delete pt;

}

}

//вывод списка индексов

template<typename tKey, typename tValue>

void Hash\_Table<tKey,tValue>::Indexes(){

Univer<tKey,tValue>\* ptr;

for(int i=0;i<SIZE;i++){

if(table[i]){

ptr=table[i];

while(ptr){

cout<<hashFunc(ptr->code)<<" ";

ptr=ptr->next;

}

}

}

}

#endif // HASH\_TABLE\_H

main.cpp

#include <iostream>

#include <windows.h>

#include <Hash\_Table.h>

using namespace std;

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

Hash\_Table<string,string> HT;

int var;

while (var){

system("cls");

cout<<"1.Вставка элемента.\n2.Вывод таблицы.\n3.Список индексов.\n4.Поиск элемента по ключу.\n5.Удаление элемента по ключу.\n0.Выход\n";

cout<<"Выберите действие:";

cin>>var;

switch(var){

case 1:{

string code,name;

cout<<"Введите код специальности:";

cin>>code;

cout<<"\nВведите название ВУЗа:";

cin>>name;

cout<<endl;

HT.push(name,code);

system("pause");

break;

}

case 2:{

HT.output();

system("pause");

break;

}

case 3:{

cout<<"\nСписок индексов\n";

HT.Indexes();

cout<<endl;

system("pause");

break;

}

case 4:{

HT.findEl();

system("pause");

break;

}

case 5:{

string el;

cout<<"Элемент на удаление:";

cin>>el;

HT.del(el);

system("pause");

break;

}

case 0:{

exit(0);

system("pause");

break;

}

default:{

cout<<"Выберите операцию из списка!\n";

system("pause");

break;

}

}

}

return 0;

}