**Задание:** разработать программу, демонстрирующую основные этапы работы с хэш-таблицами – формирование хэша, получение данных, разрешение коллизий.

**Требования:**

* Программа на языке С++.
* Теоретические оценки временной и пространственной сложности алгоритмов.
* Экспериментальные оценки временной и пространственной сложности алгоритмов.
* Рабочая версия программы на языке С++ (продемонстрировать на компьютере).
	+ Формирование хэш-таблицы (ввод данных из файла)
	+ Поиск данных в таблице
	+ Удаление элементов из таблицы
	+ Разрешение коллизий

Метод хэширования: деление.

Тип данных: числа.

Метод разрешения коллизий: открытая адресация.

#include<iostream>

#include<fstream>

#include<string>

#include<windows.h>

#include<ctime>

using namespace std;

int h(int, int);

void input(char[7]);

void add(int \*, int);

void find(int \*, int);

void del(int \*, int);

int create\_table(int \*);

void output\_in\_file(int \*, int);

int main()

{

 setlocale(LC\_ALL, "Russian");

 int smth, n, \*mas;

 for (;;)

 {

 system("cls");

 cout << "МЕНЮ" << endl << endl << "1. Создание хеш-таблицы" << endl;

 cout << "2. Добавление элемента" << endl << "3. Поиск элемента" << endl;

 cout << "4. Удаление элемента" << endl << "5. Выход из программы" << endl;

 cin >> smth;

 switch (smth)

 {

 case 1: n=create\_table(mas);

 break;

 case 2: add(mas, n);

 break;

 case 3: find(mas, n);

 break;

 case 4: del(mas, n);

 break;

 case 5: exit(0);

 break;

 default:

 cout << "Вводить нужно цифру от 1 до 5 включительно!" << endl;

 system("pause");

 break;

 }

 }

 return 0;

}

int h(int x, int n)

{

 return x % n;

}

void add(int \*mas, int n)

{

 int key, adress;

 system("cls");

 cout << "Введите число: ";

 cin >> key;

 adress = h(key, n);

 bool flag = false;

 do

 {

 if ((mas[adress] == -1))

 {

 mas[adress] = key;

 break;

 }

 else

 adress += 1;

 } while (adress<n);

 if (adress >= n){

 cout << "Массив переполнен" << endl;

 }

 else

 {

 cout << "Данные добавлены в таблицу" << endl;

 cout << adress << " " << mas[adress] << endl;

 }

 output\_in\_file(mas, n);

 system("pause");

}

void find(int \*mas, int n)

{

 int key, adress;

 system("cls");

 cout << "Введите число: ";

 cin >> key;

 adress = h(key, n);

 bool flag = false;

 do

 {

 if (mas[adress] == key)

 {

 mas[adress] = -1;

 flag = true;

 break;

 }

 else

 {

 adress += 1;

 }

 } while (adress<n);

 if (flag == false)

 cout << "Элемент не найден" << endl;

 else

 {

 cout << "Элемент найден!" << endl;

 cout << adress << " " << mas[adress] << endl;

 }

 system("pause");

}

void del(int \*mas, int n)

{

 int key, adress;

 system("cls");

 cout << "Введите число: ";

 cin >> key;

 adress = h(key, n);

 bool flag = false;

 do

 {

 if (mas[adress] == key)

 {

 mas[adress] = -1;

 flag = true;

 break;

 }

 else

 {

 adress += 1;

 }

 } while (adress<n);

 if (adress >= n)

 flag = false;

 if (flag == false)

 cout << "Элемент не найден" << endl;

 else

 {

 cout << "Элемент удален" << endl;

 cout << adress << " " << mas[adress] << endl;

 }

 output\_in\_file(mas, n);

 system("pause");

}

int create\_table(int \*mas)

{

 string file;

 int n, key, adress;

 cout << "Имя файла: ";

 cin >> file;

 ifstream f(file);

 f >> n;

 mas = new int[n];

 for (int i = 0; i < n; i++)

 mas[i] = -1;

 for (int i = 0; i < n; i++)

 {

 f >> key;

 adress = h(key, n);

 mas[adress] = key;

 }

 output\_in\_file(mas, n);

 return n;

}

void output\_in\_file(int \*mas, int n)

{

 ofstream ofs("Table.txt");

 for (int i = 0; i<n; i++)

 {

 ofs << i << " " << mas[i] << endl;

 }

 ofs.close();

}