Министерство образования и науки Российской Федерации  
Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский  
политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЁТ**

**по лабораторной работе №17**

Дисциплина: «Основы теории алгоритмов и структуры данных»

Тема: Методы быстрого поиска. Хеширование

Вариант 19

Выполнил:

Студент группы РИС-20-1б

Ремянников Александр Владимирович

Проверила:

Доцент кафедры ИТАС

Полякова О. А.

**Пермь, 2021**

**Цель работы**

Изучить построение функции хеширования и алгоритмов хеширования данных.

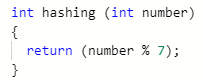
**Постановка задачи**

(Вариант 19) Выполнить поиск элемента в массиве по ключу в соответствии с вариантом. Для поиска использовать хеш-таблицу.

**Анализ задачи**

**1.** Для решения задачи необходимо:

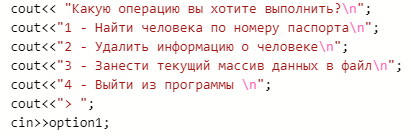
**1.1.** Реализовать функцию хеширования.



**1.2.** Реализовать код, использующий функцию хеширования для создания таблицы хешей.



**1.3.** Реализовать меню для взаимодействия с пользователем.

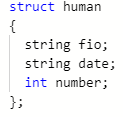


**1.4.** Реализовать код для поиска элемента массива по его хеш-функции.



**2.** В ходе работы были использованы следующие типы данных:

**2.1.** Для хранения информации о людях используется структура human.

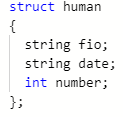


**2.2.** Для получения информации из файла используется класс ifstream.



**3.** Для решения задачи данные были представлены в следующем виде:

**3.1.** Информация о людях была предствленна в виде массива структур human.



**4.** Для операций ввода и вывода использовались следующие операторы и функции:

**4.1.** Для операции ввода из файла использовалась функция getline.



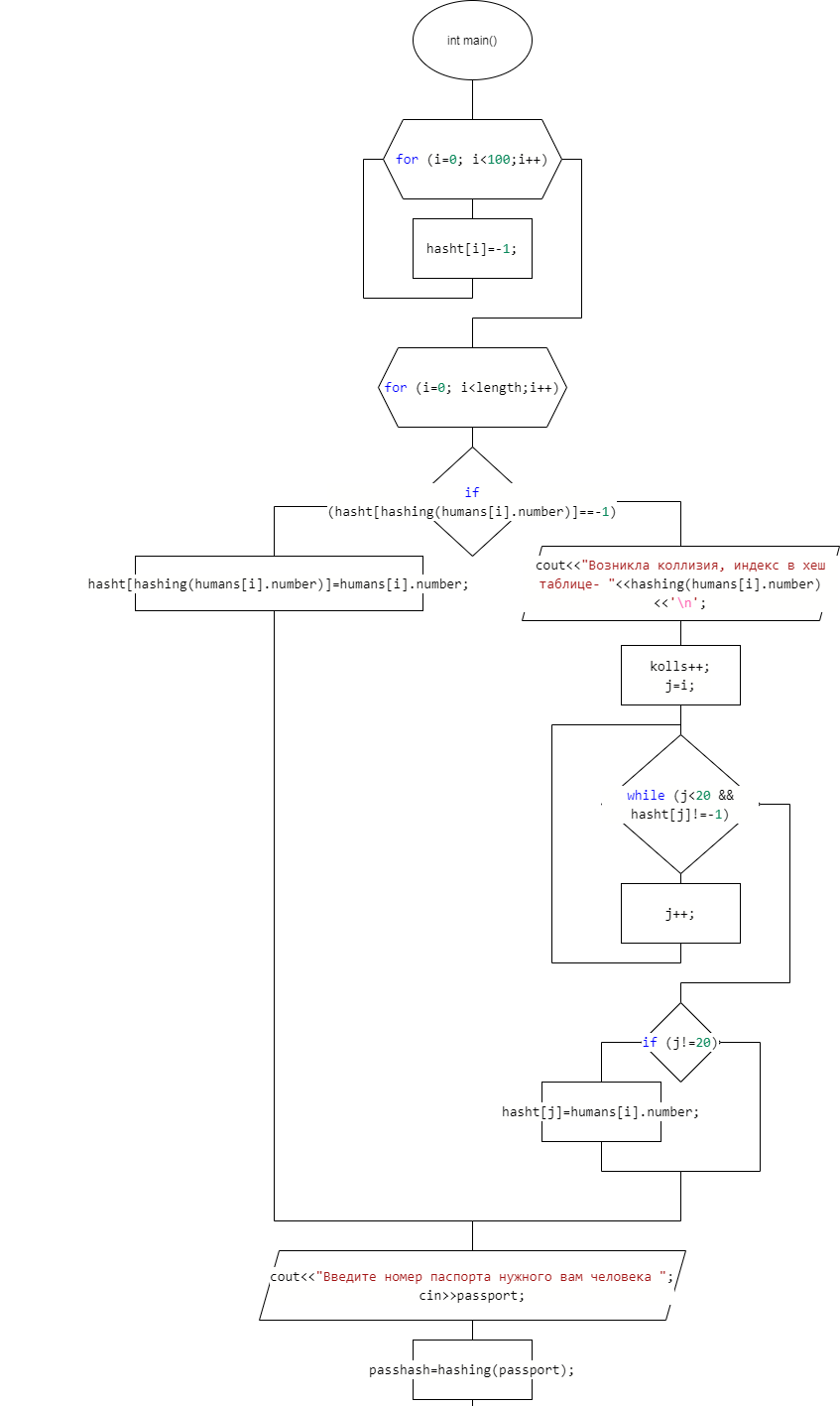
**4.2.** Для операции вывода в файл и на консоль использовался оператор cout.

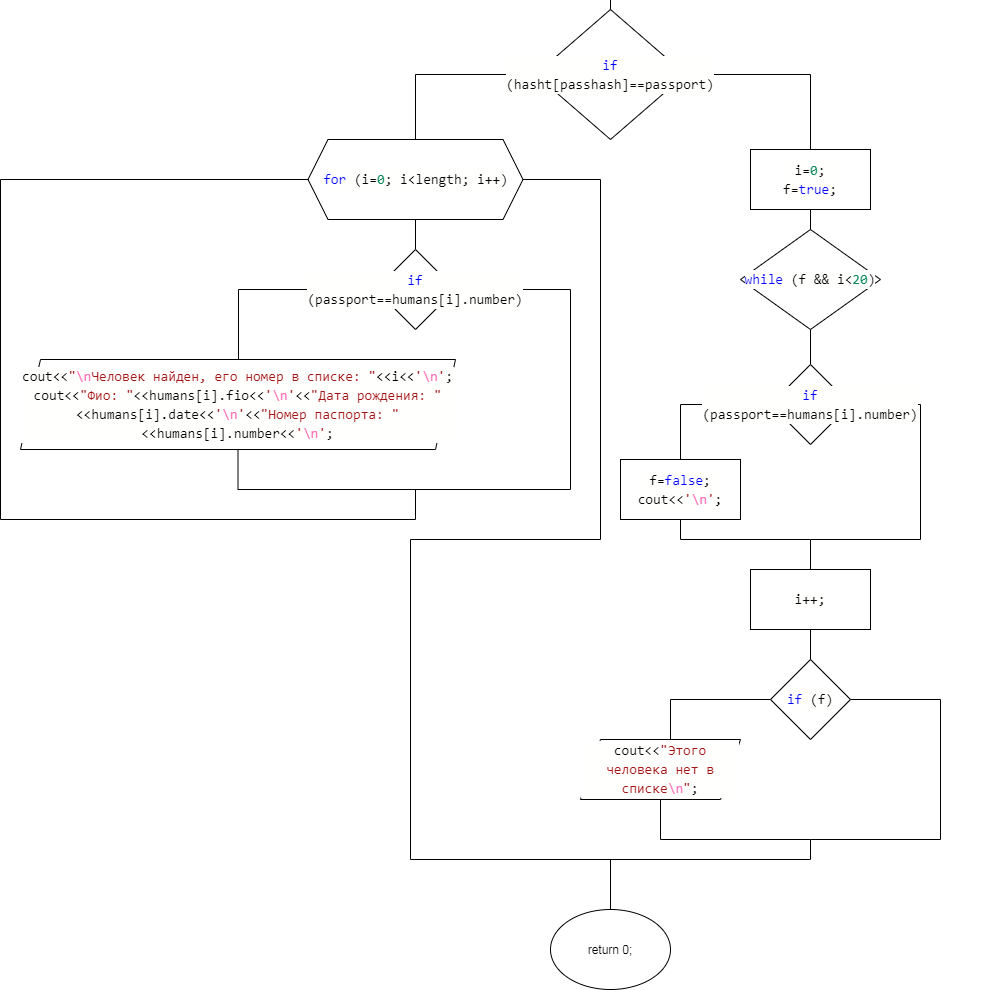


**5.** Поставленные задачи будут решены следующими действиями:

**5.1.** Код для создания хеш-таблицы будет применен для операции поиска по номеру паспорта. 

**Блок-схема программы**





**Решение**

#include <bits/stdc++.h>

using namespace std;

struct human

{

string fio;

string date;

int number;

};

int hashing (int);

int main()

{

// Описание переменных

ifstream input ("F.txt");

string row, row2;

human humans[100], humans2[100], y;

int i=0, j, number=0, passport, left=0, right=4, mid, g, option1=0, option2, length, position, hasht[177], h, kolls=0, passhash=-1;

bool f,fl=false;

//

// Читаем информацию из файла

while (!input.eof())

{

getline (input, row);

humans[i].fio=row;

getline (input, row);

humans[i].date=row;

getline (input, row);

humans[i].number=(row[0]-'0')\*100000+(row[1]-'0')\*10000+(row[2]-'0')\*1000+(row[3]-'0')\*100+(row[4]-'0')\*10+(row[5]-'0');

i++;

}

input.close();

length=i;

ofstream output("F.txt");

//

// Создание таблицы хешей

for (i=0; i<100;i++)

{

hasht[i]=-1;

}

for (i=0; i<length;i++)

{

if (hasht[hashing(humans[i].number)]==-1)

{

hasht[hashing(humans[i].number)]=humans[i].number;

}

else

{

cout<<"Возникла коллизия, индекс в хеш таблице- "<<hashing(humans[i].number)<<'\n';

kolls++;

j=i;

while (j<20 && hasht[j]!=-1)

{

j++;

}

if (j!=20)

{

hasht[j]=humans[i].number;

}

}

}

cout<<'\n';

//

// Выводим массив в консоль

cout<<"Ваш массив:\n";

for (i=0;i<length;i++)

{

cout<<"Фио: "<<humans[i].fio<<'\n'<<"Дата рождения: "<<humans[i].date<<'\n'<<"Номер паспорта: "<<humans[i].number<<'\n'<<'\n';

}

//

// Вывод таблицы хешей

cout<<"Общее кол-во коллизий: "<<kolls<<'\n';

for (i=0;i<7+kolls;i++)

{

if (hasht[i]!=-1)

{

cout<<"Индекс: "<<i<<" Номер паспорта: "<<hasht[i]<<'\n';

}

else

{

cout<<"Индекс: "<<i<<" Нет значения\n";

}

}

cout<<'\n';

//

while (option1!=4)

{

// Вывод пунктов меню и ввод номера операции

cout<< "Какую операцию вы хотите выполнить?\n";

cout<<"1 - Найти человека по номеру паспорта\n";

cout<<"2 - Удалить информацию о человеке\n";

cout<<"3 - Занести текущий массив данных в файл\n";

cout<<"4 - Выйти из программы \n";

cout<<"> ";

cin>>option1;

//

if (option1<=4 && option1>=1)

{

// Выполнение операции

switch (option1)

{

case 1:

// Поиск по номеру паспорта

cout<<"Введите номер паспорта нужного вам человека ";

cin>>passport;

passhash=hashing(passport);

if (hasht[passhash]==passport)

{

for (i=0; i<length; i++)

{

if (passport==humans[i].number)

{

cout<<"\nЧеловек найден, его номер в списке: "<<i<<'\n';

cout<<"Фио: "<<humans[i].fio<<'\n'<<"Дата рождения: "<<humans[i].date<<'\n'<<"Номер паспорта: "<<humans[i].number<<'\n';

}

}

}

else

{

i=0;

f=true;

while (f && i<20)

{

if (passport==humans[i].number)

{

cout<<"\nЧеловек найден, его номер в списке: "<<i<<'\n';

cout<<"Фио: "<<humans[i].fio<<'\n'<<"Дата рождения: "<<humans[i].date<<'\n'<<"Номер паспорта: "<<humans[i].number<<'\n';

f=false;

cout<<'\n';

}

i++;

}

if (f)

{

cout<<"Этого человека нет в списке\n";

}

}

break;

//

case 2:

// Удаление элемента

// Вывод пунктов меню и ввод номера операции

cout<<" 1 - Удаление по номеру паспорта\n";

cout<<" 2 - Удаление с указанной позиции\n";

cout<<" > ";

cin>>option2;

//

// Варианты удаления элемента

switch (option2)

{

case 1:

// По номеру паспорта

cout<<"Введите номер паспорта человека, информацию о котором вы хотите удалить ";

cin>>row;

fl=false;

for (i=0;i<length;i++)

{

if (humans[i].number==(row[0]-'0')\*100000+(row[1]-'0')\*10000+(row[2]-'0')\*1000+(row[3]-'0')\*100+(row[4]-'0')\*10+(row[5]-'0'))

{

fl=true;

break;

}

}

if (fl)

{

passhash=hashing((row[0]-'0')\*100000+(row[1]-'0')\*10000+(row[2]-'0')\*1000+(row[3]-'0')\*100+(row[4]-'0')\*10+(row[5]-'0'));

if (hasht[passhash]==passport)

{

i=0;

}

else

{

i=0;

f=true;

while (f && i<20)

{

if (passport==humans[i].number)

{

f=false;

cout<<'\n';

}

i++;

}

if (f)

{

cout<<"Этого человека нет в списке\n";

}

else

{

i-=3;

}

}

cout<< "Вы собиратесь удалить элемент с индексом "<<i<<'\n';

cout<<"Фио: "<<humans[i].fio<<'\n'<<"Дата рождения: "<<humans[i].date<<'\n'<<"Номер паспорта: "<<humans[i].number<<'\n'<<'\n';

for (;i<length;i++)

{

humans[i].fio=humans[i+1].fio;

humans[i].date=humans[i+1].date;

humans[i].number=humans[i+1].number;

}

humans[i].fio="";

humans[i].date="";

humans[i].number=0;

length--;

}

else

{

cout<<"В списке нет человека с заданным номером паспорта\n";

}

break;

//

case 2:

// По заданной позиции

cin>>position;

if (position<length)

{

cout<< "Вы собиратесь удалить элемент с индексом "<<i<<'\n';

cout<<"Фио: "<<humans[i].fio<<'\n'<<"Дата рождения: "<<humans[i].date<<'\n'<<"Номер паспорта: "<<humans[i].number<<'\n'<<'\n';

for (i=position;i<length;i++)

{

humans[i].fio=humans[i+1].fio;

humans[i].date=humans[i+1].date;

humans[i].number=humans[i+1].number;

}

humans[i].fio="";

humans[i].date="";

humans[i].number=0;

length--;

}

else

{

cout<<"В списке нет элемента с заданным номером";

}

//

default:

// Неверный номер

cout<<"Выбран несуществующий вариант";

break;

//

}

//

// Выводим массив в консоль

cout<<"Ваш массив:\n";

for (i=0;i<length;i++)

{

cout<<"Фио: "<<humans[i].fio<<'\n'<<"Дата рождения: "<<humans[i].date<<'\n'<<"Номер паспорта: "<<humans[i].number<<'\n'<<'\n';

}

//

//

break;

//

case 3:

// Запись массива в файл

cout<<"Проводим запись в файл...\n";

for (i=0;i<length-1;i++)

{

output<<humans[i].fio;

output<<endl;

output<<humans[i].date;

output<<endl;

output<<humans[i].number;

output<<endl;

}

output<<humans[i].fio;

output<<endl;

output<<humans[i].date;

output<<endl;

output<<humans[i].number;

output.close();

break;

//

case 4:

// Выход из программы

for (i=0;i<length-1;i++)

{

output<<humans[i].fio;

output<<endl;

output<<humans[i].date;

output<<endl;

output<<humans[i].number;

output<<endl;

}

output<<humans[i].fio;

output<<endl;

output<<humans[i].date;

output<<endl;

output<<humans[i].number;

output.close();

break;

//

default:

// Неверный номер

cout<<"Выбран несуществующий вариант";

break;

//

}

//

}

else

{

option1=4;

cout<<"\nВыбран неверный вариант";

}

}

return 0;

}

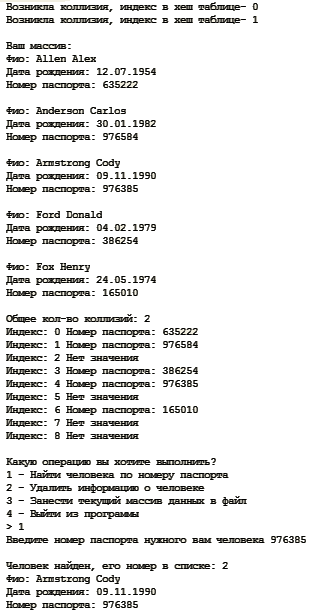
int hashing (int number)

{

return (number % 7);

}

**Скриншоты результатов работы программы**

****

**GitHub**

[**https://github.com/sugarrrfqs/Lab17**](https://github.com/sugarrrfqs/Lab17)