Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«**Пермский национальный исследовательский политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЕТ**

Дисциплина: «Основы алгоритмизации и программирования»

Тема: «Поиск данных с помощью хэш - таблиц»

Семестр 2

Выполнил работу

Студент группы РИС-22-1Б

Желнин Н.А.

Проверил

Доцент кафедры ИТАС

Полякова О.А.

Г. Пермь-2023

# Постановка задачи

Реализовать хэш - таблицу с элементами, содержащими ФИО, дату рождения и адрес. Найти индекс элемента по ключу (в данном случае — по адресу)

# Алгоритм решения

1. Создаем строки, содержащие данные для создания элемента.
2. Создаем структуру человека.
3. Создаем структуру хэш - таблицы.
4. Создаем функцию поиска элемента по ключу.
5. Соединяем все данные (ФИО, дату рождения и адрес) в один элемент - человек.
6. Реализуем хэш — функцию в соответствии с вариантом задания.

# 3. Код программы

#include<iostream>

#include<string>

#include<random>

#include<time.h>

using namespace std;

string name[10] = {"Иван", "Андрей", "Антон", "Амин", "Роберт", "Георгий", "Борис", "Владислав", "Дмитрий", "Никита"};

string patronymic[10] = {"Иванович", "Андреевич", "Антонович", "Сергеевич", "Робертович", "Георгиевич", "Борисович", "Владиславович", "Дмитриевич", "Ильич"};

string surname[10] = {"Карпов", "Афанасьев", "Власов","Маслов","Исаков","Тихонов","Аксёнов","Гаврилов","Родионов","Котов"};

string dates[10] = {"12.12.2011","18.05.1030","09.01.2000","21.03.2003","19.10.2010","23.09.2009","24.10.2010","25.11.2011","26.12.2012","27.01.1988"};

string adress[10] = {"Уральская 2","Петрова 23","Солнечная 34","Еремова 8","Васина 43","Кутакбетова 333","Магомедовна 777","Щукина 25","Язь 93","Волжская 12"};

int coll\_num = 0;

struct Human

{

Human()

{

full\_name = "NULL";

date = "NULL";

adress = "NULL";

}

string full\_name;

string date;

string adress;

};

struct hash\_table

{

Human\*array;

hash\_table(int size)

{

array = new Human[size];

}

~hash\_table()

{

delete[]array;

}

void find\_index(string adress, const int size);

void add(Human tmp, const int size);

};

void show\_table(hash\_table\*table, const int size);

void show(const Human tmp);

Human creator();

string get\_random\_name(); string get\_random\_birthday(); string get\_random\_adress();

void print\_Humans(const Human\* const array, const int size);

int get\_random\_num();

int get\_sum\_id(string str);

int hash\_function(string str, const int size);

void fill\_creator(Human\*array, const int count);

void hash\_table::find\_index(string adress, const int size)

{

int hash = hash\_function(adress, size);

int index = hash;

while (array[index].adress != adress && index < size)

{

index++;

}

if (index >= size)

{

index = 0;

while (array[index].adress != adress && index < hash)

{

index++;

}

if (index >= hash)

{

cout << "Человека с адресом: " << adress << " нет" << endl;

}

else

{

cout << "Человек с адресом: " << adress << " найден с индексом: " << index << endl;

}

}

else

{

cout << "Человек с адресом: " << adress << " найден с индексом: " << index << endl;

}

}

void hash\_table::add(Human tmp, const int size)

{

int index = hash\_function(tmp.adress, size);

int hash = index;

if (array[index].full\_name == "NULL")

{

array[index] = tmp;

return;

}

else

{

while (index < size)

{

if (array[index].full\_name == "NULL")

{

array[index] = tmp;

return;

}

index++;

coll\_num++;

}

if (index >= size)

{

index = 0;

coll\_num++;

while (index < hash)

{

if (array[index].full\_name == "NULL")

{

array[index] = tmp;

return;

}

index++;

coll\_num++;

}

}

}

}

Human creator()

{

Human tmp;

tmp.full\_name = get\_random\_name();

tmp.date = get\_random\_birthday();

tmp.adress = get\_random\_adress();

return tmp;

}

string get\_random\_name()

{

return(surname[get\_random\_num()] + " " + name[get\_random\_num()] + " " + patronymic[get\_random\_num()]);

}

string get\_random\_birthday()

{

return dates[get\_random\_num()];

}

string get\_random\_adress()

{

return adress[get\_random\_num()];

}

void show(const Human tmp)

{

cout << "ФИО: " << tmp.full\_name<<endl;

cout << "Дата рождения: " << tmp.date << endl;

cout << "Адрес: " << tmp.adress << endl;

}

void print\_Humans(const Human\* const array, const int size)

{

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

{

show(array[i]);

}

}

void fill\_creator(Human\*array, const int count)

{

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

{

array[i] = creator();

}

}

int get\_random\_num()

{

return rand() % 10;

}

int get\_sum\_id(string str)

{

int sum = adress[get\_random\_num()].length();

return sum;

}

int hash\_function(string str, const int size)

{

double a = 0.618\*get\_sum\_id(str);

double c = size \* (a - static\_cast<int>(a));

return static\_cast<int>(c);

}

void show\_table(hash\_table\*table, const int size)

{

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

{

show(table->array[i]);

}

}

int main()

{

srand(time(0));

setlocale(LC\_ALL, "Rus");

int size;

cout << "Введите количество элементов в массиве(не менее 100): ";

cin >> size;

cout << endl;

Human\*array = new Human[size];

hash\_table table(size);

fill\_creator(array, size);

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

{

table.add(array[i], size);

}

show\_table(&table, size);

table.find\_index("Еремова 8", size);

cout << endl << "===============" << endl;

cout << "Кол-во колллизий для " << size << " записей равно " << coll\_num << endl;

delete[] array;

return 0;

}

4. Вывод в консоль

