Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Пермский национальный исследовательский**

**политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**О Т Ч Ё Т**

**по лабораторной работе №14**

Дисциплина: основы алгоритмизации и программирования

Тема: “Поиск данных в динамических структурах по заданному ключу”

Вариант 16

Выполнил:

студент группы ИВТ-20-2Б Тедеев А.З.

Проверила: доцент кафедры ИТАС

Полякова О.А.

Пермь, 2021

**Постановка задачи**

1. Создать динамический массив из записей (в соответствии с вариантом), содержащий не менее 100 элементов. Для заполнения элементов массива использовать ГСЧ.

2. Выполнить поиск элемента в массиве по ключу в соответствии с вариантом. Для поиска использовать метод Кнута-Морриса-Пратта, Бойера-Мура и подстроки в строке.

2

**Анализ задачи**

**1.** Для решения задачи необходимо:

**1.1.** Организовать функцию KnutMorrisPratt(), которая отвечает за поиск методом Кнута-Морриса-Пратта.

**1.2.** Организовать функцию BoyerMur(), которая отвечает за поиск методом Бойера-Мура.

**1.3.** Организовать функцию SearchInSubstrInString(), которая отвечает за прямой поиск.

**1.4.** Организовать функцию ConvertData(), которая преобразует ключ в строку, а также все строки из массива структур в большую строку.

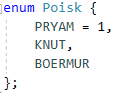
**1.5.** Организовать функцию CheckingResult, которая проверяет, был ли найден ключ среди элементов массива структур.

**1.6.** Организовать структуру Date для хранения даты рождения.

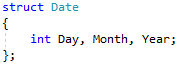
**1.7.** Организовать структуру Human для организации массива структур с методами Random() для генерации данный в поля и Show() для вывода данных в консоль.

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

**2.1.** enum для повышения читаемости кода, так как используются функции switch.

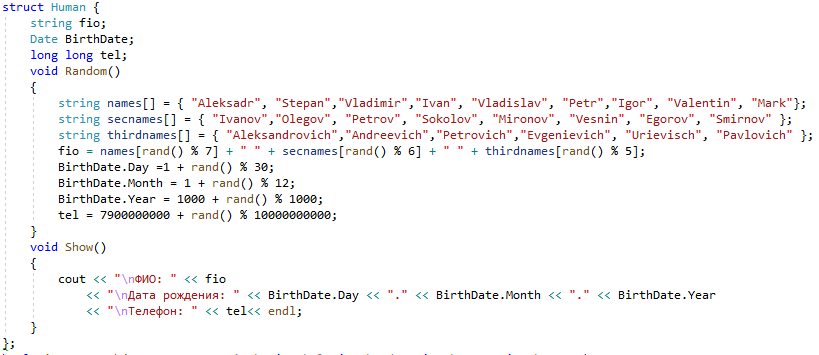


**2.2.** Структура Date для хранения дат рождения.



**2.3.** Структура Human для хранения данных людях.

3



Поле fio типа string для хранения имени, фамилии и отчества. Поле BirthDate типа Date для хранения даты рождения (в свою очередь структура Date хранит дни, недели, годы в разных полях типа int) и поле tel типа long long для хранения номера телефона. Метод Random(), в котором генерируются случайным образом в каждом из полей данные: под поле fio используются три массива строк, в которые соответственно вписаны имена, фамилии и отчества, при заполнении поля fio используется функция rand(), которая и перебирает данные.

**2.5.** В главной функции используется многократно функция switch под «меню», например, под выбор метода поиска, выбора удаления или добавления элемента и т.д.

cout << "\nКаким методом поиска воспользоваться? " << endl

<< "1) Прямой поиск;" << endl

<< "2) Поиск Кнута-Морриса-Пратта;" << endl

<< "3) Поиск Бойера-Мура; " << endl << endl << ">";

cin >> method;

switch (method) {

case PRYAM:

...

break;

case KNUT:

...

break;

case BOERMUR:

...

break;

default:

...

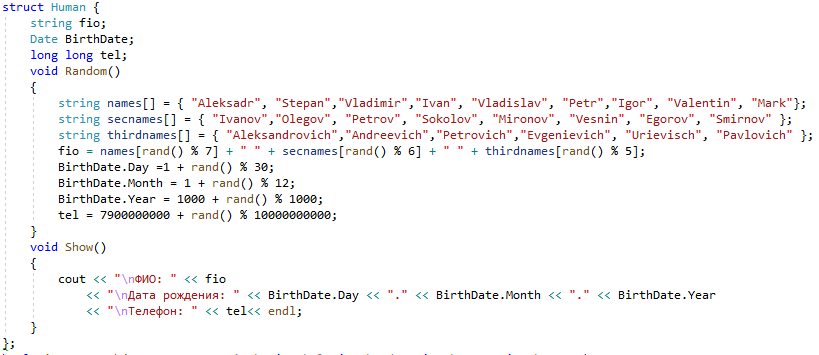
break;

}

4

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

**3.1.** Для обработки данных используется массив структур Human.

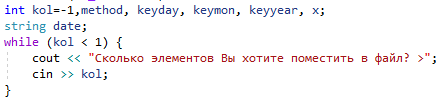


**3.2.** Для хранения данных используется вектор типа Human

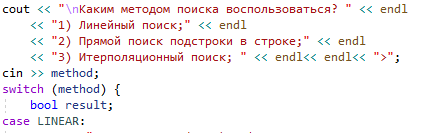
vector<Human> spisok(kol);

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

**4.1.** Для ввода количества элементов используется переменная kol. Ввод осуществляется командой cin.



**4.2.** Переменная method необходима для предоставления пользователю выбора метода поиска в массиве структур. Ввод осуществляется командой cin.

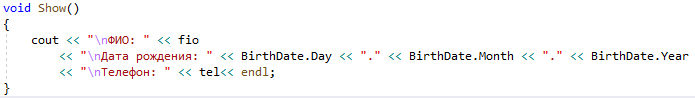


**4.3.** Переменные keyday, keymon, keyyear типа int необходимы для хранения ключа (т.е. даты). Вводятся через функцию cin по порядку.



5

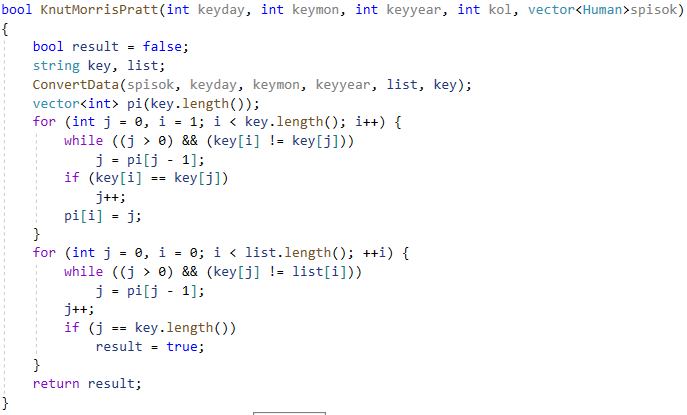
**4.4.** Вывод на консоль осуществляется с помощью метода Show() структуры Human в цикле for.



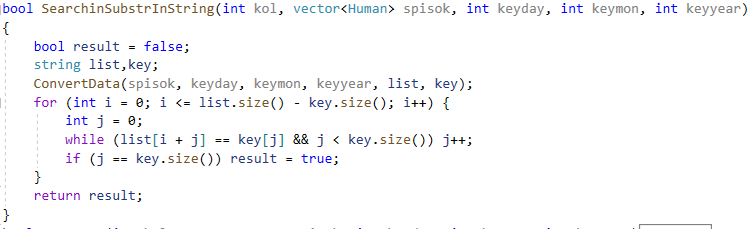


**5.** Поставленные задачи будут решены следующими действиями:

**5.1.** Вызов функции KnutMorrisPratt() типа bool присваивается переменной result. В функции KnutMorrisPratt() производится поиск элемента по ключу методом Кнута-Морриса-Пратта.

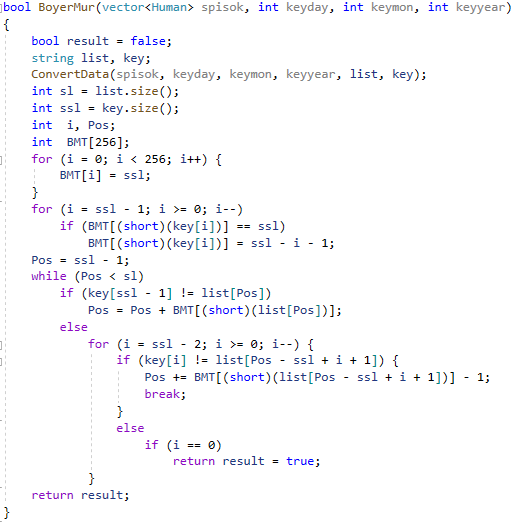


**5.2.** Вызов функции SearchInSubstrInString() типа bool присваивается переменной result. В функции SearchInSubstrInString() производится поиск элемента по ключу методом прямого поиска.

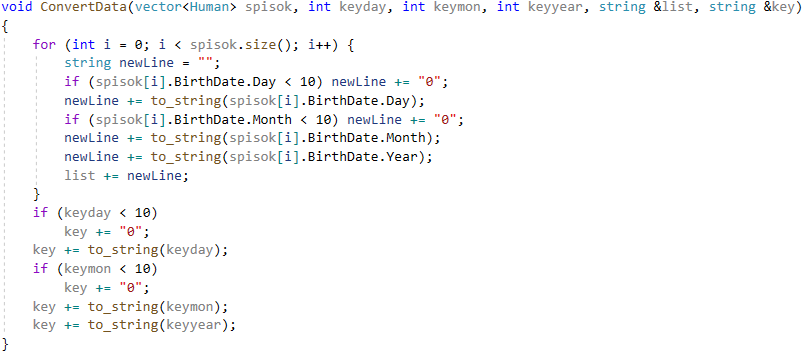


6

**5.3.** Вызов функции BoyerMur() типа bool присваивается переменной result. В функции BoyerMur() производится поиск элемента по ключу методом прямого поиска.



**5.4.** Функция ConvertData() типа void, в которой производится преобразование ключа в строку и преобразование всех дат рождения в одну большую строк – подготовительный этап для всех поисковых функций.



7

**5.5.** Меню с выбором метода поиска выполняется с помощью функции switch

cout << "\nКаким методом поиска воспользоваться? " << endl

<< "1) Прямой поиск;" << endl

<< "2) Поиск Кнута-Морриса-Пратта;" << endl

<< "3) Поиск Бойера-Мура; " << endl << endl << ">";

cin >> method;

switch (method) {

case PRYAM:

...

break;

case KNUT:

...

break;

case BOERMUR:

...

break;

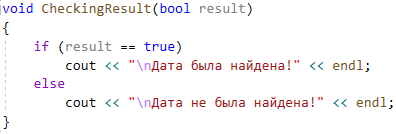
default:

...

break;

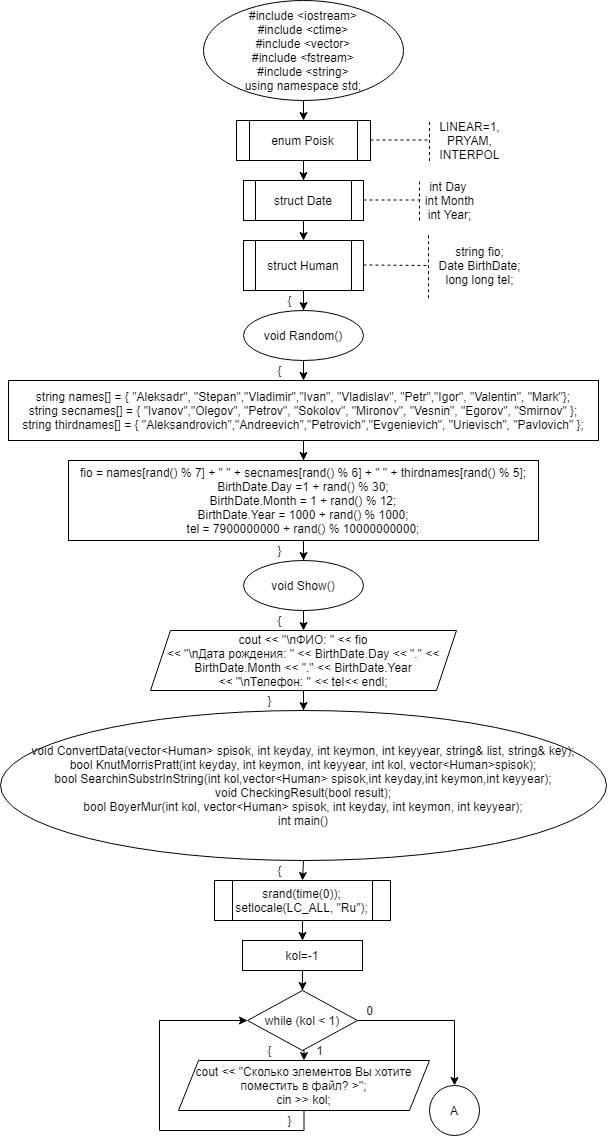
}

**5.6.** Если результат поиска положителен, то после выполнения функции CheckingResult() пользователь выбирает функцию добавления или удаления по ключу.

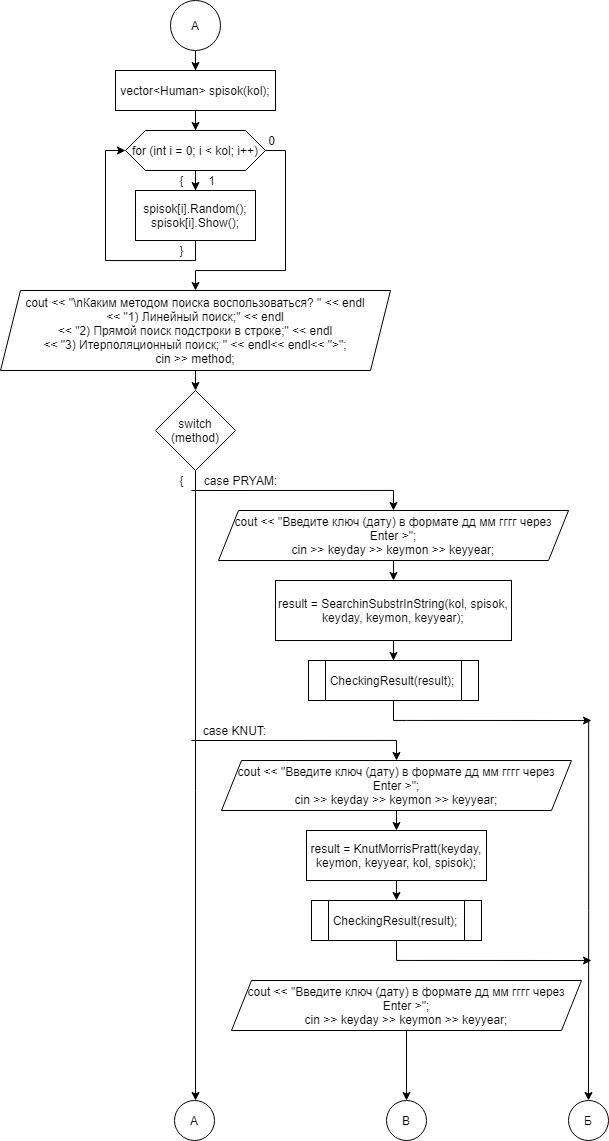


8

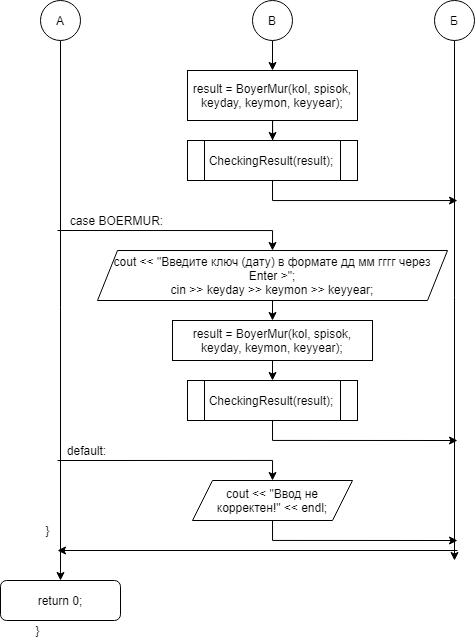
**Блок-схема**



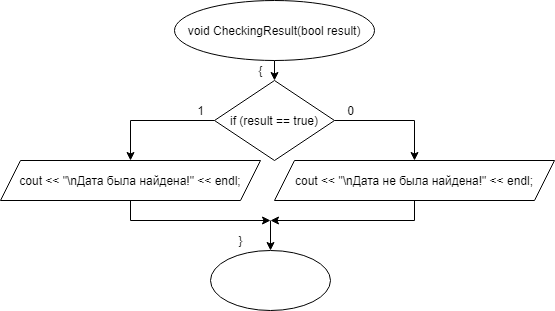
9



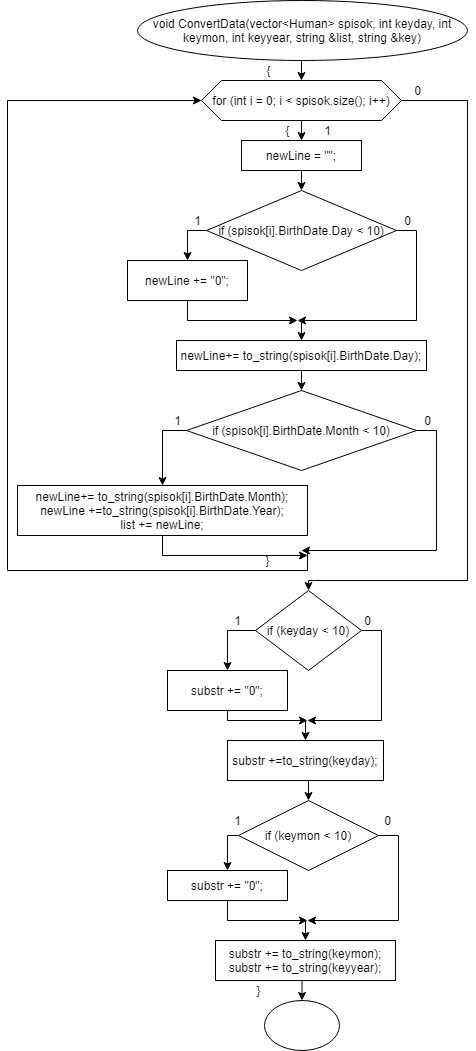
10



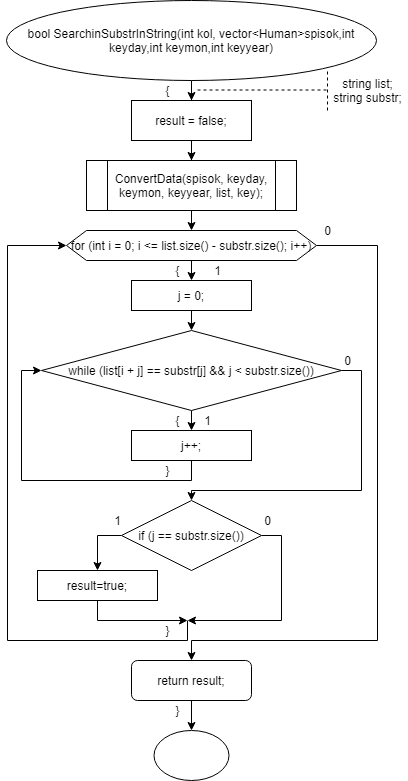
11



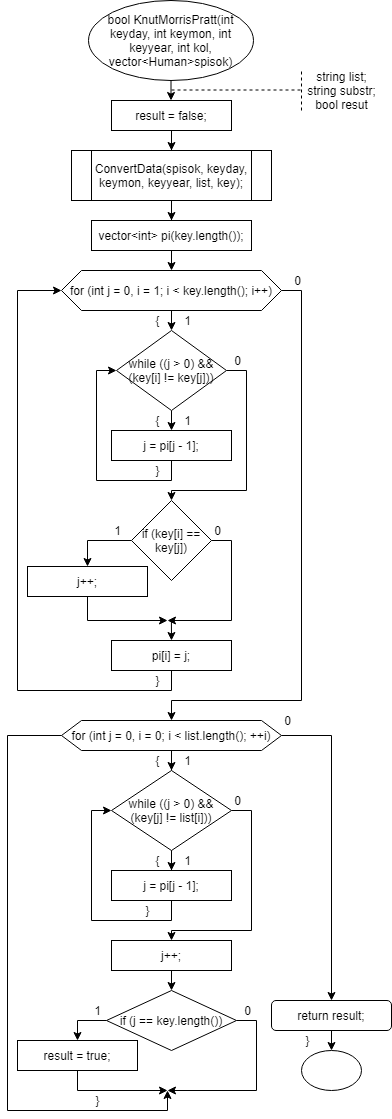
12



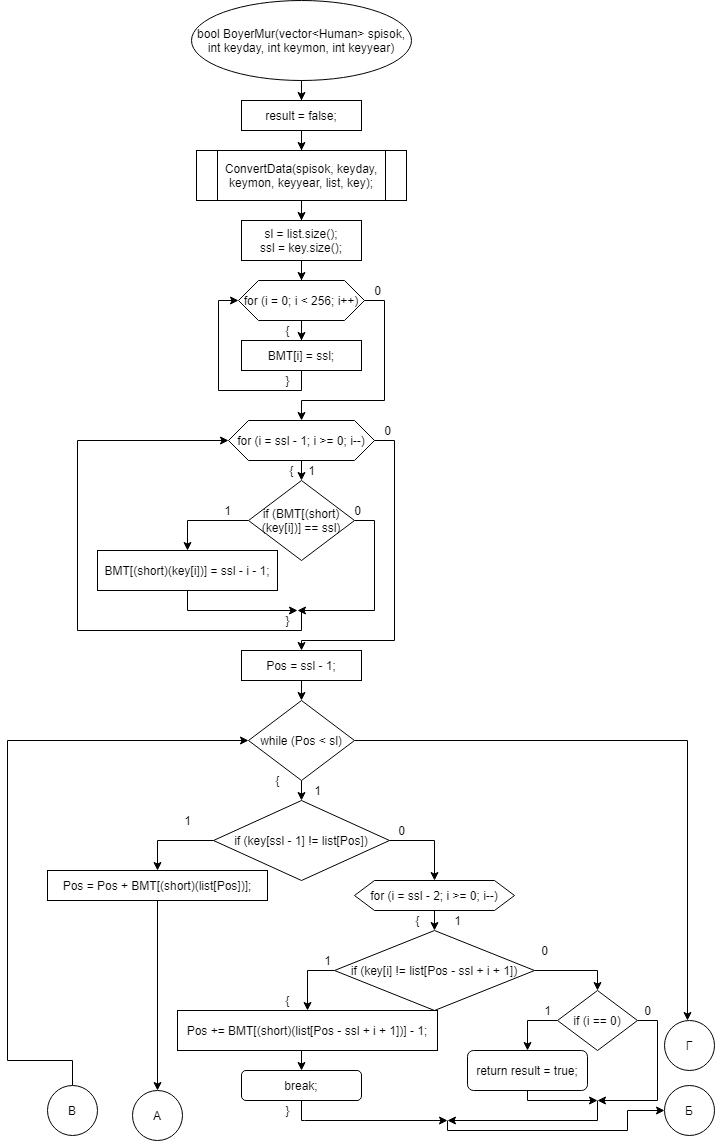
13



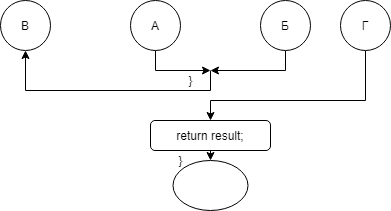
14



15



16



17

**Код**

#include <iostream>

#include <string>

#include <ctime>

#include <vector>

using namespace std;

enum Poisk {

PRYAM = 1,

KNUT,

BOERMUR

};

struct Date

{

int Day, Month, Year;

};

struct Human {

string fio;

Date BirthDate;

long long tel;

void Random()

{

string names[] = { "Aleksadr", "Stepan","Vladimir","Ivan", "Vladislav", "Petr","Igor", "Valentin", "Mark" };

string secnames[] = { "Ivanov","Olegov", "Petrov", "Sokolov", "Mironov", "Vesnin", "Egorov", "Smirnov" };

string thirdnames[] = { "Aleksandrovich","Andreevich","Petrovich","Evgenievich", "Urievisch", "Pavlovich" };

fio = names[rand() % 7] + " " + secnames[rand() % 6] + " " + thirdnames[rand() % 5];

BirthDate.Day = 1 + rand() % 30;

BirthDate.Month = 1 + rand() % 12;

BirthDate.Year = 1000 + rand() % 1000;

tel = 7900000000 + rand() % 10000000000;

}

void Show()

{

cout << "\nФИО: " << fio

<< "\nДата рождения: " << BirthDate.Day << "." << BirthDate.Month << "." << BirthDate.Year

<< "\nТелефон: " << tel << endl;

}

};

void ConvertData(vector<Human> spisok, int keyday, int keymon, int keyyear, string& list, string& key);

bool KnutMorrisPratt(int keyday, int keymon, int keyyear, int kol, vector<Human>spisok);

bool SearchinSubstrInString(int kol,vector<Human> spisok,int keyday,int keymon,int keyyear);

void CheckingResult(bool result);

bool BoyerMur(vector<Human> spisok, int keyday, int keymon, int keyyear);

int main()

{

srand(time(0));

setlocale(LC\_ALL, "Ru");

int kol = -1, method, keyday, keymon, keyyear, x;

bool result = false;

while (kol < 1) {

cout << "Введите количество элементов >";

cin >> kol;

}

vector<Human> spisok(kol);

18

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

spisok[i].Random();

spisok[i].Show();

}

cout << "\nКаким методом поиска воспользоваться? " << endl

<< "1) Прямой поиск;" << endl

<< "2) Поиск Кнута-Морриса-Пратта;" << endl

<< "3) Поиск Бойера-Мура; " << endl << endl << ">";

cin >> method;

switch (method) {

case PRYAM:

cout << "\nВведите ключ (дату) через точки в формате дд мм гггг через Enter >";

cin >> keyday >> keymon >> keyyear;

result = SearchinSubstrInString(kol, spisok, keyday, keymon, keyyear);

CheckingResult(result);

break;

case KNUT:

cout << "Введите ключ (дату) в формате дд мм гггг через Enter >";

cin >> keyday >> keymon >> keyyear;

result = KnutMorrisPratt(keyday, keymon, keyyear, kol, spisok);

CheckingResult(result);

break;

case BOERMUR:

cout << "\nВведите ключ (дату) через точки в формате дд мм гггг через Enter >";

cin >> keyday >> keymon >> keyyear;

result = BoyerMur(spisok, keyday, keymon, keyyear);

CheckingResult(result);

break;

default:

cout << "Ввод не корректен!" << endl;

break;

}

}

bool KnutMorrisPratt(int keyday, int keymon, int keyyear, int kol, vector<Human>spisok)

{

bool result = false;

string key, list;

ConvertData(spisok, keyday, keymon, keyyear, list, key);

vector<int> pi(key.length());

for (int j = 0, i = 1; i < key.length(); i++) {

while ((j > 0) && (key[i] != key[j]))

j = pi[j - 1];

if (key[i] == key[j])

j++;

pi[i] = j;

}

for (int j = 0, i = 0; i < list.length(); ++i) {

while ((j > 0) && (key[j] != list[i]))

j = pi[j - 1];

j++;

if (j == key.length())

result = true;

}

return result;

}

void CheckingResult(bool result)

{

if (result == true)

cout << "\nДата была найдена!" << endl;

19

else

cout << "\nДата не была найдена!" << endl;

}

bool SearchinSubstrInString(int kol, vector<Human> spisok, int keyday, int keymon, int keyyear)

{

bool result = false;

string list,key;

ConvertData(spisok, keyday, keymon, keyyear, list, key);

for (int i = 0; i <= list.size() - key.size(); i++) {

int j = 0;

while (list[i + j] == key[j] && j < key.size()) j++;

if (j == key.size()) result = true;

}

return result;

}

bool BoyerMur(vector<Human> spisok, int keyday, int keymon, int keyyear)

{

bool result = false;

string list, key;

ConvertData(spisok, keyday, keymon, keyyear, list, key);

int sl = list.size();

int ssl = key.size();

int i, Pos;

int BMT[256];

for (i = 0; i < 256; i++) {

BMT[i] = ssl;

}

for (i = ssl - 1; i >= 0; i--)

if (BMT[(short)(key[i])] == ssl)

BMT[(short)(key[i])] = ssl - i - 1;

Pos = ssl - 1;

while (Pos < sl)

if (key[ssl - 1] != list[Pos])

Pos = Pos + BMT[(short)(list[Pos])];

else

for (i = ssl - 2; i >= 0; i--) {

if (key[i] != list[Pos - ssl + i + 1]) {

Pos += BMT[(short)(list[Pos - ssl + i + 1])] - 1;

break;

}

else

if (i == 0)

return result = true;

}

return result;

}

void ConvertData(vector<Human> spisok, int keyday, int keymon, int keyyear, string &list, string &key)

{

for (int i = 0; i < spisok.size(); i++) {

string newLine = "";

if (spisok[i].BirthDate.Day < 10) newLine += "0";

newLine += to\_string(spisok[i].BirthDate.Day);

if (spisok[i].BirthDate.Month < 10) newLine += "0";

newLine += to\_string(spisok[i].BirthDate.Month);

newLine += to\_string(spisok[i].BirthDate.Year);

list += newLine;

}

20

if (keyday < 10)

key += "0";

key += to\_string(keyday);

if (keymon < 10)

key += "0";

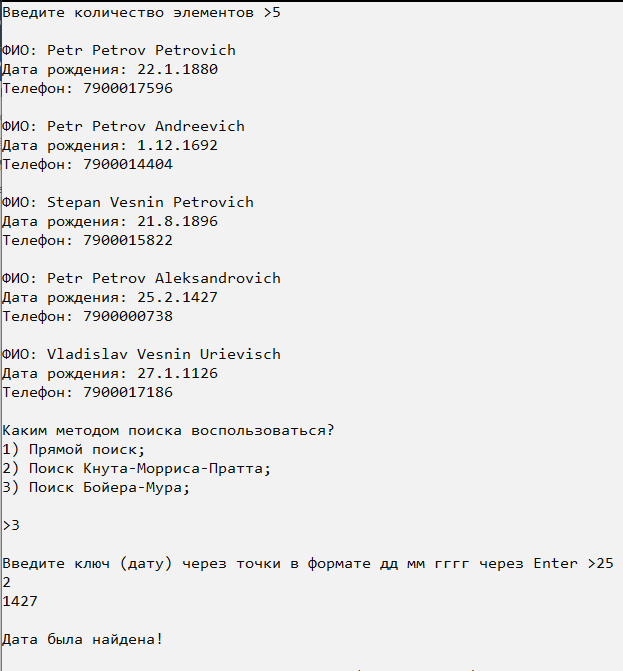
key += to\_string(keymon);

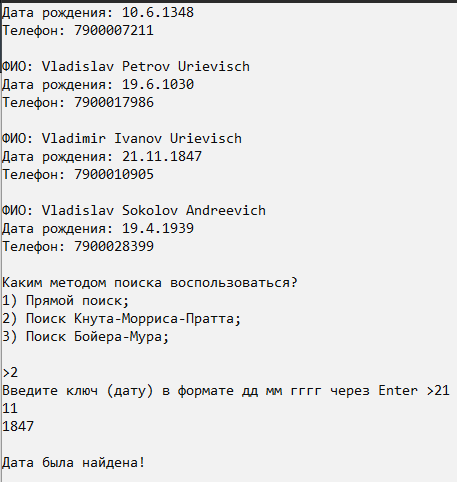
key += to\_string(keyyear);

}

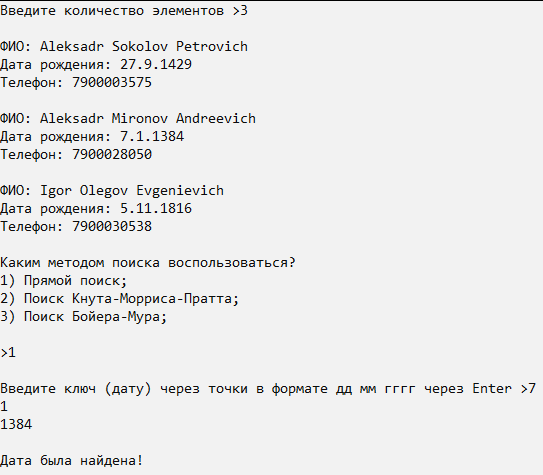
21

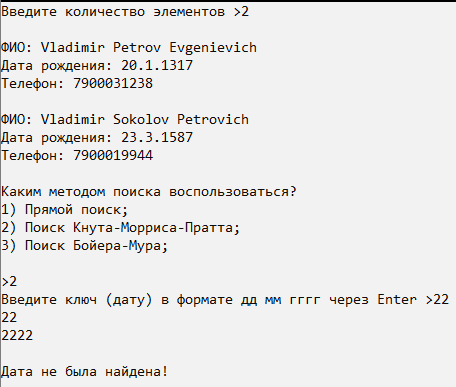
**Скриншоты**





22





23