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 теории алгоритмов и структуры данных

Тема: Методы поиска: метод Кнутта-Мориса Пратта, метод Бойера-Мура

Вариант: 20

Выполнил работу

студент группы РИС-20-1б

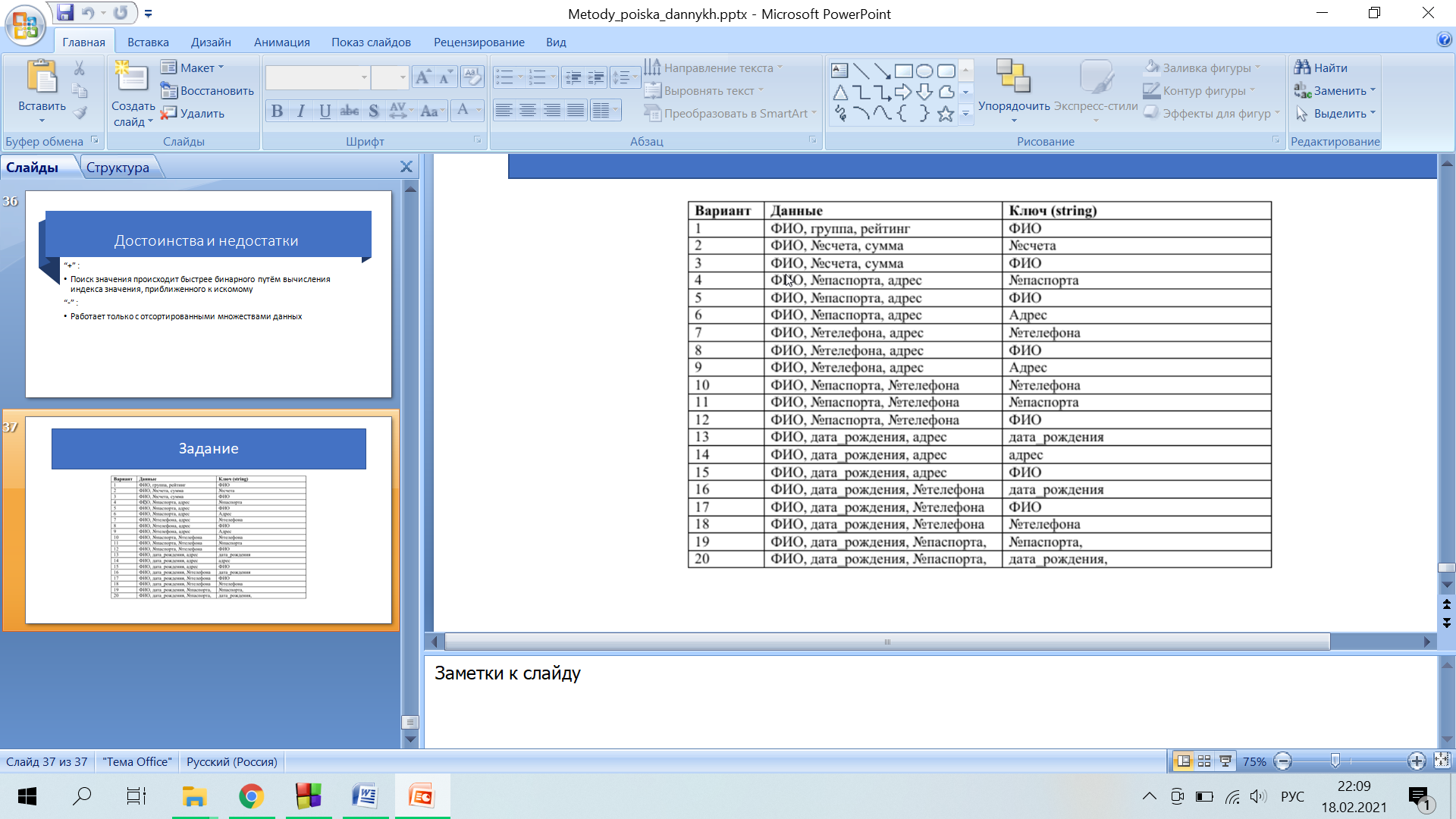
Рябов Никита Андреевич

Проверила

Доцент кафедры ИТАС Полякова О.А.

Пермь, 2021

**Постановка задачи**



**Анализ задачи**

*Какие предстоит выполнить действия.*

Считываем длину массива и считываем массив соответствующей длины

while (len<2 || len>100)

{

cout << "Введите размер таблицы\n";

cin >> len;

}

cin.get();

CITIZEN \*data\_base=new CITIZEN[len];

ifstream F1("F1.txt");

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

{

getline(F1,data\_base[i].fio);

getline(F1,data\_base[i].birthday);

getline(F1,data\_base[i].passport);

}

Запрашиваем у пользователя ключ поиска и получаем ответ от функций

cout << "Введите запрос\n";

while (user\_search.size()<8) cin >> user\_search;

int pos1=Linear(data\_base,user\_search);

int pos2=Binary(data\_base,user\_search);

int pos3=Interpol(data\_base,user\_search);

Вывод результата

if (pos1==-1 || pos2==-1 || pos3==-1) cout << "Элемент не найден\n";

else

{

cout << "Результат линейного поиска:\n";

Print(data\_base[pos1]);

cout << "Результат бинарного поиска:\n";

Print(data\_base[pos2]);

cout << "Результат интерполяционного поиска:\n";

Print(data\_base[pos3]);

cout << endl;

}

Функция, печатающая элемент структуры

void Print(CITIZEN a)

{

cout << "ФИО : " << a.fio << endl;

cout << "Дата рождения : " << a.birthday << endl;

cout << "Номер пасспорта: " << a.passport << endl;

}

Просто пробегает по массиву до первого совпадения. Если таких нет – выводит -1

int Linear(string s, string x)

{

bool flag;

for (int i=0;i<(len-1)\*8;i++)

{

flag=true;

for (int k=0;k<8;k++)

{

if (x[k]!=s[i+k])

{

flag=false;

break;

}

}

if (flag) return i/8;

}

return -1;

}

Функция Кнута-Морриса-Пратта. Строи пи-массив из длин наибольших равных пар суффиксов и префиксов. Затем пробегает по строке, как в линейном поиске, но смещается не на 1, а на значения из пи-массива.

int KMP(string s, string x)

{

int ls = s.size();

int lx = x.size();

int pi[lx];

pi[0] = 0;

int i = 1;

int j = 0;

int index\_str = 0;

int index\_pod = 0;

while (i<lx)

{

if (x[i] != x[j])

{

if (j == 0)

{

pi[i] = 0;

i++;

}

else j = pi[j - 1];

}

else

{

pi[i] = j + 1;

i++;

j++;

}

}

while (index\_str < ls)

{

if (s[index\_str] == x[index\_pod])

{

index\_str++;

index\_pod++;

if (index\_pod == lx) return (index\_str-8)/8;

}

else if (s[index\_str] != x[index\_pod])

{

if (index\_pod == 0) index\_str++;

else index\_pod = pi[index\_pod - 1];

}

}

return -1;

}

Функция Бойера-Мура. Строит таблицу смещения из индексов первых вхождений с конца для каждого символа. Затем пробегает по строке, как в линейном поиске, но смещается не на 1, а на значения из таблицы смещения.

*С каким типом данных действуем*

Динамический массив структур, часть данных конвертируется в одну строку

*В каком виде представлены данные*

Строки, содержащие любые символы

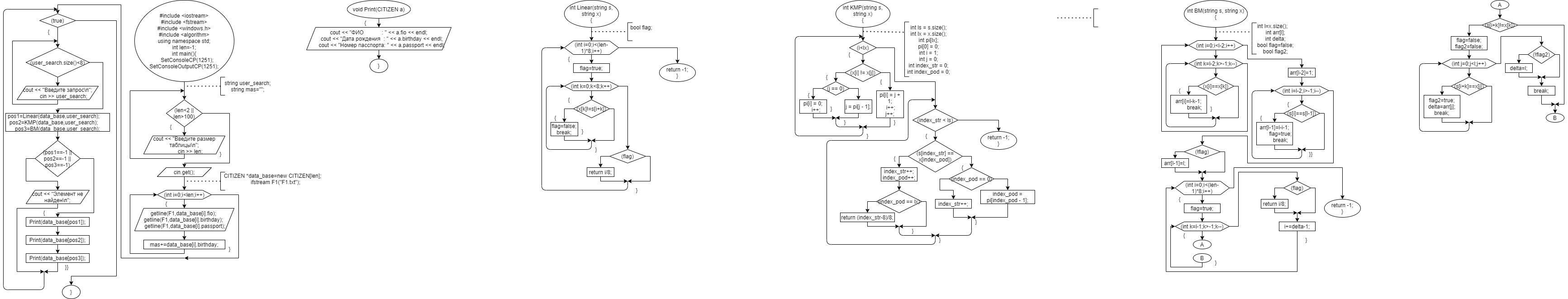
*Какие поля*

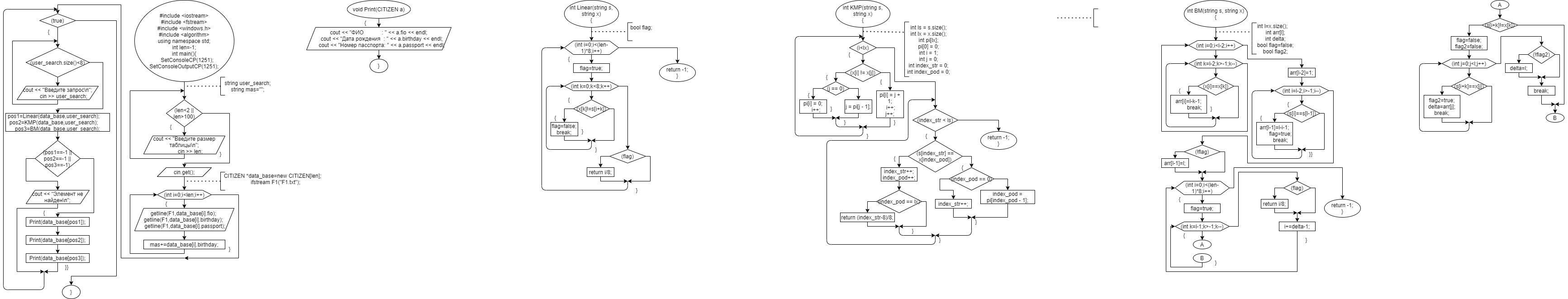
Структура содержит поле string для ФИО, поле string для даты рождения и поле string для номера пасспорта

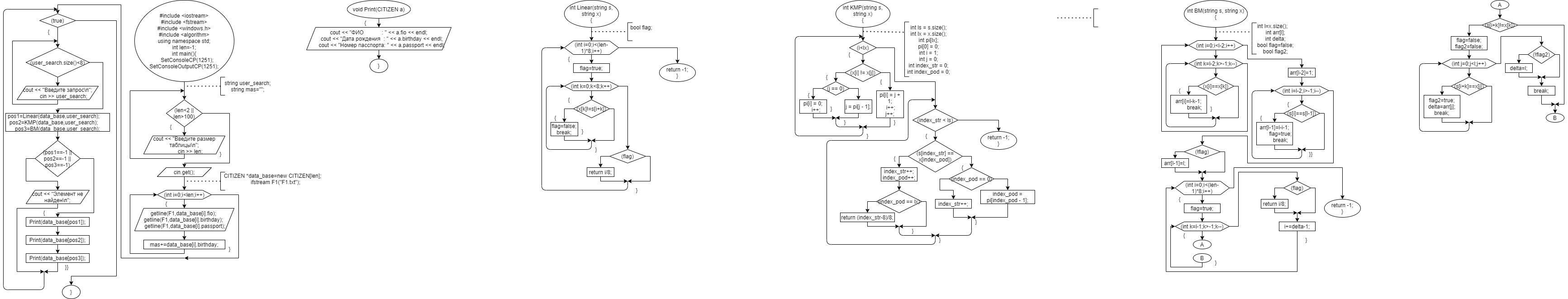
*Операторы ввода/вывода*

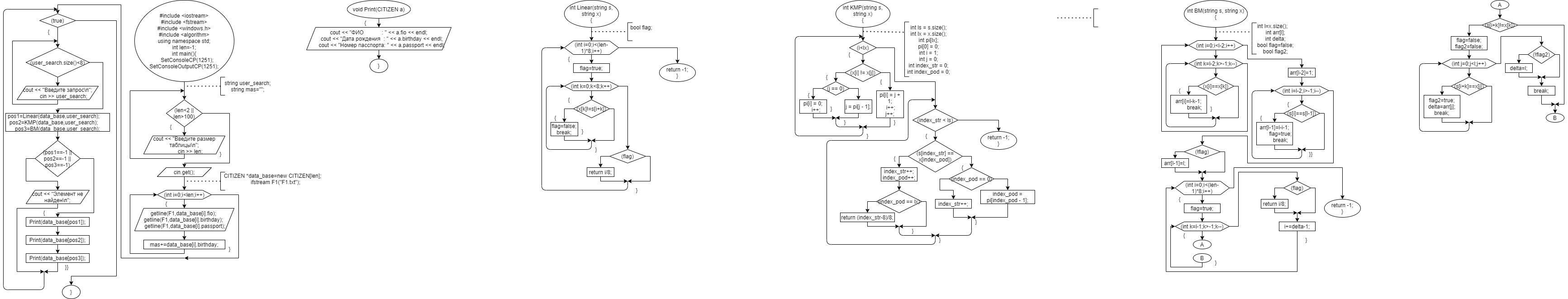
В основном используются операторы cin/cout. Исключение составляет ввод строк. Они могут содержать пробелы, и к ним применяется метод getline().

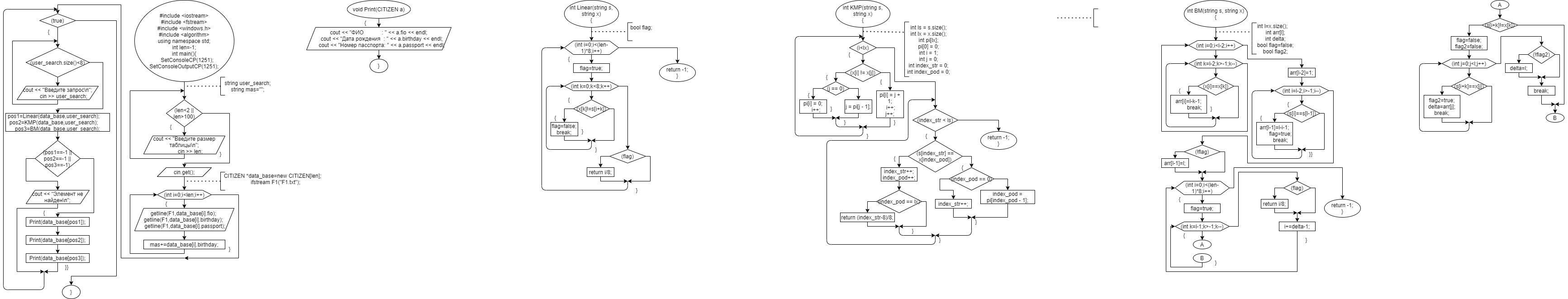
**Блок-схема**

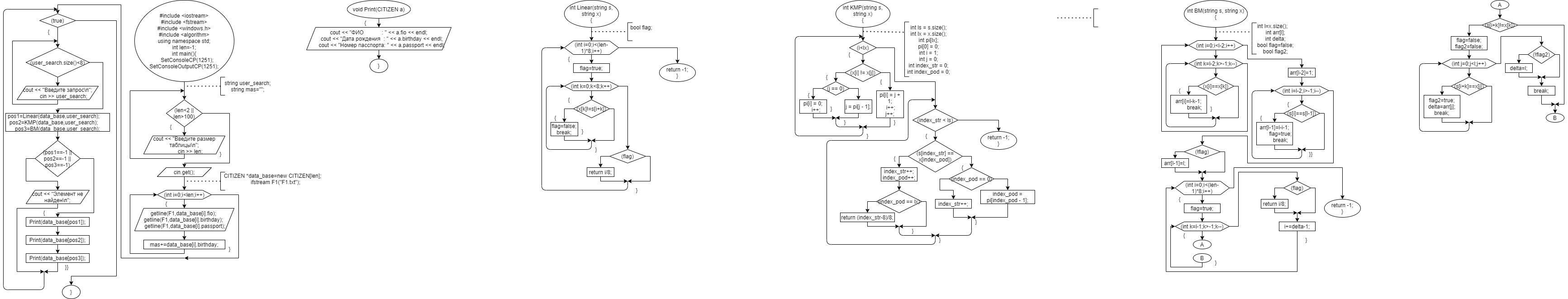
****

****

****

****

****

****

**Код**

#include <iostream>

#include <fstream>

#include <windows.h>

#include <algorithm>

using namespace std;

int len=-1;

struct CITIZEN

{

string fio;

string birthday;

string passport;

};

void Print(CITIZEN a)

{

cout << "ФИО : " << a.fio << endl;

cout << "Дата рождения : " << a.birthday << endl;

cout << "Номер пасспорта: " << a.passport << endl;

}

int Linear(string s, string x)

{

bool flag;

for (int i=0;i<(len-1)\*8;i++)

{

flag=true;

for (int k=0;k<8;k++)

{

if (x[k]!=s[i+k])

{

flag=false;

break;

}

}

if (flag) return i/8;

}

return -1;

}

int KMP(string s, string x)

{

int ls = s.size();

int lx = x.size();

int pi[lx];

pi[0] = 0;

int i = 1;

int j = 0;

int index\_str = 0;

int index\_pod = 0;

while (i<lx)

{

if (x[i] != x[j])

{

if (j == 0)

{

pi[i] = 0;

i++;

}

else j = pi[j - 1];

}

else

{

pi[i] = j + 1;

i++;

j++;

}

}

while (index\_str < ls)

{

if (s[index\_str] == x[index\_pod])

{

index\_str++;

index\_pod++;

if (index\_pod == lx) return (index\_str-8)/8;

}

else

{

if (index\_pod == 0) index\_str++;

else index\_pod = pi[index\_pod - 1];

}

}

return -1;

}

int BM(string s, string x)

{

int l=x.size();

int arr[l];

int delta;

bool flag=false;

bool flag2;

for (int i=0;i<l-2;i++) for (int k=l-2;k>-1;k--) if (x[i]==x[k])

{

arr[i]=l-k-1;

break;

}

arr[l-2]=1;

for (int i=l-2;i>-1;i--) if (s[i]==s[l-1])

{

arr[l-1]=l-i-1;

flag=true;

break;

}

if (!flag) arr[l-1]=l;

for (int i=0;i<(len-1)\*8;i++)

{

flag=true;

for (int k=l-1;k>-1;k--) if (s[i+k]!=x[k])

{

flag=false;

flag2=false;

for (int j=0;j<l;j++) if (s[i+k]==x[j])

{

flag2=true;

delta=arr[j];

break;

}

if (!flag2) delta=l;

break;

}

if (flag) return i/8;

i+=delta-1;

}

return -1;

}

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

string user\_search;

while (len<2 || len>100)

{

cout << "Введите размер таблицы\n";

cin >> len;

}

cin.get();

CITIZEN \*data\_base=new CITIZEN[len];

string mas="";

ifstream F1("F1.txt");

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

{

getline(F1,data\_base[i].fio);

getline(F1,data\_base[i].birthday);

getline(F1,data\_base[i].passport);

mas+=data\_base[i].birthday;

}

while (true)

{

user\_search="1";

cout << "Введите запрос\n";

while (user\_search.size()<8) cin >> user\_search;

int pos1=Linear(mas,user\_search);

cout << 1;

int pos2=KMP(mas,user\_search);

cout << 2;

int pos3=BM(mas,user\_search);

cout << 3;

if (pos1==-1 || pos2==-1 || pos3==-1) cout << "Элемент не найден\n";

else

{

cout << "Результат прямого поиска:\n";

Print(data\_base[pos1]);

cout << "Результат метода КМП:\n";

Print(data\_base[pos2]);

cout << "Результат метода БМ:\n";

Print(data\_base[pos3]);

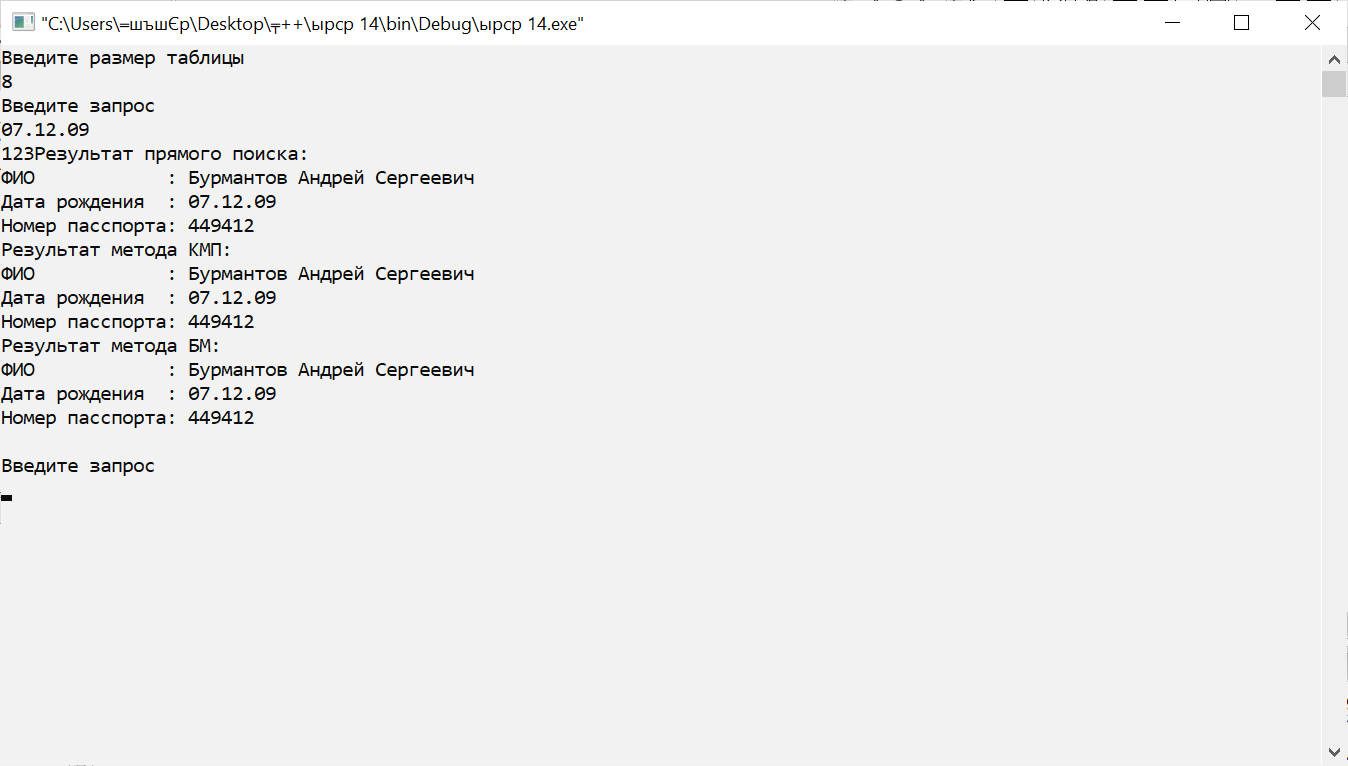
cout << endl;

}

}

}

**Скриншоты**

****