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 №12**

Дисциплина: основы алгоритмизации и программирования

Тема: “Методы поиска” в C++

Вариант 3

Выполнил работу

студент группы ИВТ-20-2б

Солдатов А.П.

Проверил

Доцент кафедры ИТАС

Полякова О.А.

Пермь, 2021

****Цель задачи****

Цель – написать программу для осуществления метода поиска Кнута, Морриса и Пратта и метода Бойера и Мура.

Для достижения поставленной цели необходимо решить следующие задачи:

* Понять, как воспользоваться алгоритмами поиска, в условиях поставленной задачи
* Проанализировать алгоритмы Кнута, Морриса и Пратта и Бойера и Мура

Постановка задачи

Выполнить поиск элемента в массиве по ключу в соответствии с вариантом, используя методы Кнута, Мориса и Пратта и Бойера и Мура

Анализ задачи

1. Определить какие операции должны быть выполнены по заданию:

* Создание структуры данных, среди которых будет осуществлен поиск

struct info

{

string fio;

string date;

string numberpPass;

};

* Создание меню с возможностью выбора метода поиска и защитой от случайных ошибок

switch (a)

{

case 0:

boyer(st, key, n); break;

case 1:

knut(st, key, n); break;

default:

cout << "Неизвестная операция. Поиск не выполнен." << endl;

}

return 0;

do

{

cout << "Введите 0, если хотите произвести поиск алгоритмом Бойера-Мура.\nВведите 1, если хотите произвести поиск алгоритмом Кнута-Морриса-Пратта." << endl;

cin >> a;

} while (a != 0 || a != 1);

Реализация методов поиска в соответствующих функциях

1. Для решения задачи используются переменные:

Несколько целочисленных переменных для управления циклами, контроля размеров массивов и контроля длины строк

int i, j, k, d = str.size(), e = obr.size(), Pos, n;

Переменная типа string для хранения ключа

string key;

Переменная типа bool:

bool a, l, res = 0;

bool p = 0;

1. Ввод данных осуществляется посредством функции

getline(cin, key);

Вывод осуществляется через консоль

Для решения задачи будут использованы циклы. Например, цикл вывода нужного элемента структуры:

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

{

l = boyer\_moor(st[i].numberpPass, key);

if (l != 0)

{

cout << "Элемент найден под номером: " << i + 1 << endl;

cout << "ФИО: " << st[i].fio << endl;

cout << "Дата рождения: " << st[i].date << endl;

cout << "Номер паспорта: " << st[i].numberpPass << endl;

p = 1;

}

}

1. В программе использована защита от случайных ошибок в меню

do

{

cout << "Введите 0, если хотите произвести поиск алгоритмом Бойера-Мура.\nВведите 1, если хотите произвести поиск алгоритмом Кнута-Морриса-Пратта." << endl;

cin >> a;

} while (a != 0 || a != 1);

**Код**

// ФИО, дата\_рождения, № паспорта

// № паспорта

#include <iostream>

#include <string>

#include <windows.h>

using namespace std;

struct info

{

string fio;

string date;

string numberpPass;

};

bool knut\_morris\_pratt(string str, string obr)

{

int i, j, k;

bool res = 0;

int d = str.size();

int e = obr.size();

if (d == 0)

cout << "Неверно задана строка\n";

else if (e == 0)

cout << "Неверно задана подстрока\n";

else

{

j = 0;

k = -1;

}

int\* pf = new int[1000];

pf[0] = -1;

while (j < e - 1)

{

while (k >= 0 && obr[j] != obr[k])

k = pf[k];

j++;

k++;

if (obr[j] == obr[k])

pf[j] = pf[k];

else

pf[j] = k;

}

i = 0;

j = 0;

while (j < e && i < d)

{

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

j = pf[j];

i++;

j++;

}

delete[] pf;

if (j == e)

res = 1;

return res;

}

bool boyer\_moor(string str, string obr)

{

int d = str.size();

int e = obr.size();

if (d == 0)

cout << "Неверно задана строка\n";

else if (e == 0)

cout << "Неверно задана подстрока\n";

else

int i, Pos;

int bmt[256];

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

bmt[i] = e;

for (int i = e - 1; i >= 0; i--)

if (bmt[(short)(obr[i])] == e)

bmt[(short)(obr[i])] = e - i - 1;

int pos = e - 1;

while (pos < d)

if (obr[e - 1] != str[pos])

pos += bmt[(short)(str[pos])];

else

for (int i = e - 2; i >= 0; i--) {

if (obr[i] != str[pos - e + i + 1]) {

pos += bmt[(short)(str[pos - e + i + 1])] - 1;

break;

}

else

if (i == 0)

{

return true;

}

}

return false;

}

void boyer(info\* st, string key, int n)

{

bool l;

bool p = 0;

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

{

l = boyer\_moor(st[i].numberpPass, key);

if (l != 0)

{

cout << "Элемент найден под номером: " << i + 1 << endl;

cout << "ФИО: " << st[i].fio << endl;

cout << "Дата рождения: " << st[i].date << endl;

cout << "Номер паспорта: " << st[i].numberpPass << endl;

p = 1;

}

}

if (p == 0)

{

cout << "Такого элемента не найдено" << endl;

}

}

void knut(info\* st, string key, int n)

{

bool l;

bool p = 0;

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

{

l = knut\_morris\_pratt(st[i].numberpPass, key);

if (l != 0)

{

cout << "Элемент найден под номером: " << i + 1 << endl;

cout << "ФИО: " << st[i].fio << endl;

cout << "Дата рождения: " << st[i].date << endl;

cout << "Номер паспорта: " << st[i].numberpPass << endl;

p = 1;

}

}

if (p == 0)

{

cout << "Такого элемента не найдено" << endl;

}

}

int main()

{

setlocale(LC\_ALL, "rus");

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int n;

string key;

cout << "Кол-во людей: ";

do

{

cin >> n;

}while (n < 0);

cin.ignore();

info\* st = new info[n];

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

{

cout << "\nФИО: ";

getline(cin, st[i].fio);

cout << "Дата рождения: ";

getline(cin, st[i].date);

cout << "Номер паспорта: ";

getline(cin, st[i].numberpPass);

}

cout << "Ключ для поиска(номер паспорта): ";

getline(cin, key);

bool a;

switch (a)

{

case 0:

boyer(st, key, n); break;

case 1:

knut(st, key, n); break;

default:

cout << "Неизвестная операция. Поиск не выполнен." << endl;

}

return 0;

do

{

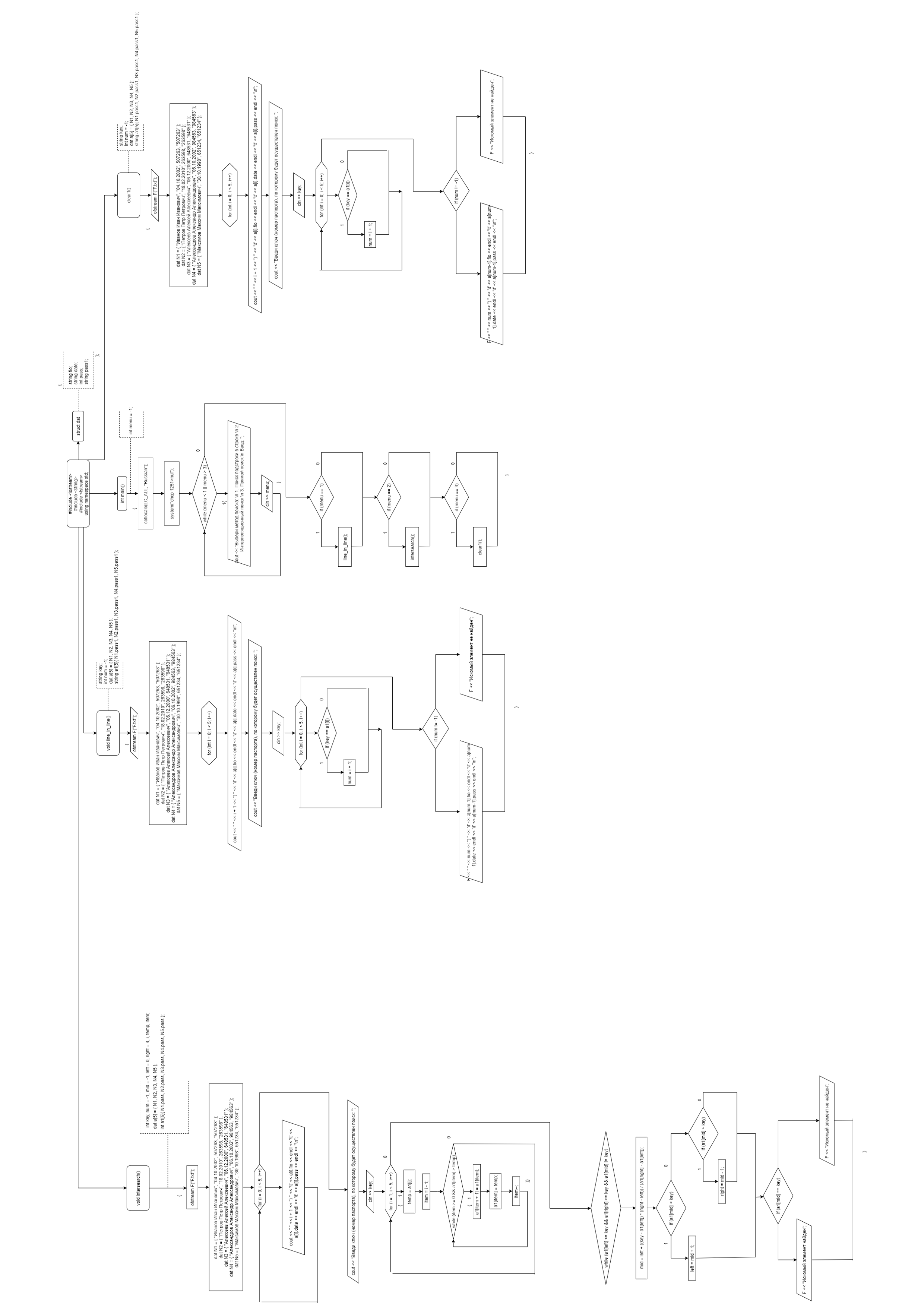
cout << "Введите 0, если хотите произвести поиск алгоритмом Бойера-Мура.\nВведите 1, если хотите произвести поиск алгоритмом Кнута-Морриса-Пратта." << endl;

cin >> a;

} while (a != 0 || a != 1);

}

**Блок-схема**



**Работа кода**

