**«Пермский национальный исследовательский  
политехнический университет»**

**О Т Ч Ё Т**

**по лабораторной работе №14**

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

Вариант 7

Выполнил работу

студент группы ИВТ-20-2б

Копытов А.А.

Проверила

Доцент кафедры ИТАС

Полякова О.А.

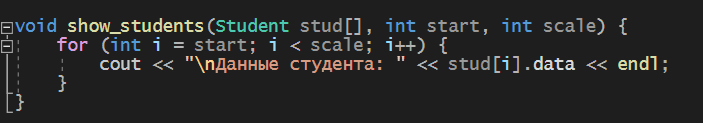
Пермь, 2021

**Постановка задачи**

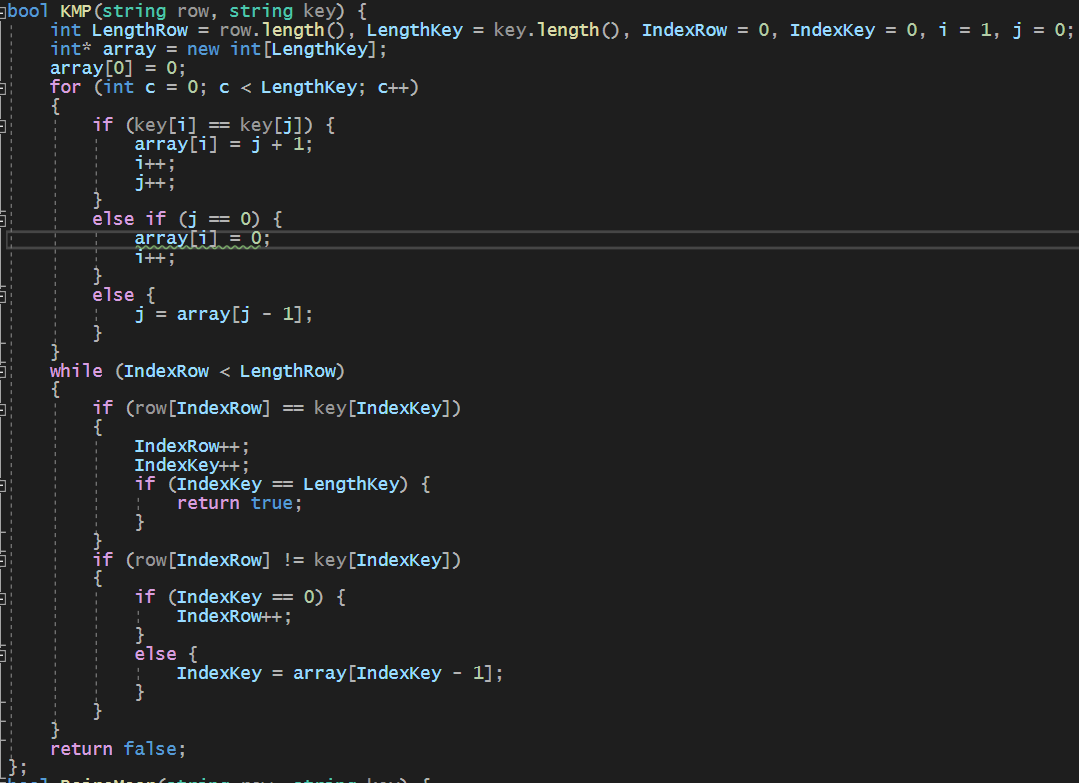
1. Дан список студентов, содержащий в себе их ФИО, паспортные данные и адрес.
2. Реализовать поиск по фамилии с помощью метода Бойлера—Мура и Кнута-Морриса-Пратта.

**Анализ задачи**

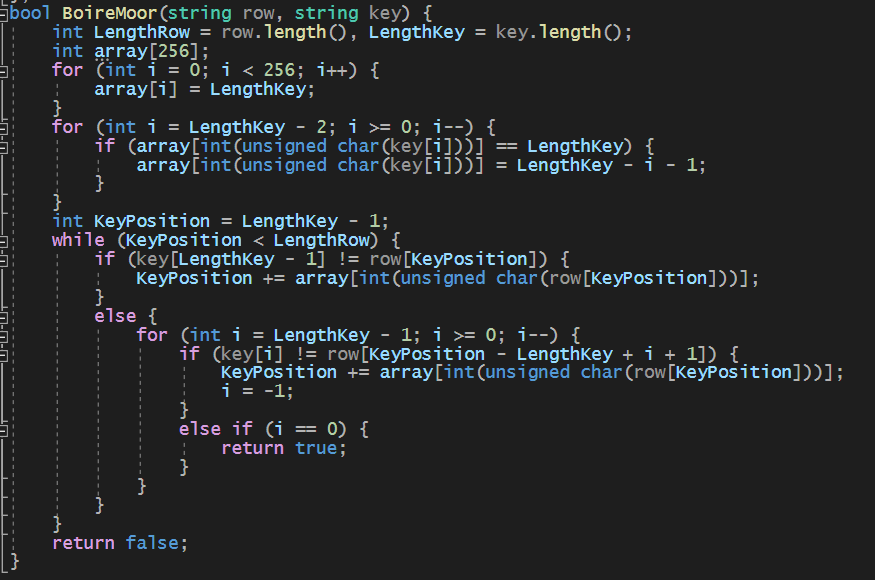
1. Для решения задачи необходимо…
   1. Создать функцию show\_student типа void, которая выводит данные студентов на экран. То есть пользователь вводит нужное количество студентов, и программа выводит нужное количество людей в консоль;



* 1. Реализовать функцию KMP типа bool, которая будет искать нужного студента по фамилии по методу Кнута—Морриса—Пратта;

****

* 1. Реализовать функцию BoireMoor типа bool, которая ищет нужного студента по методу Бойлера—Мура;



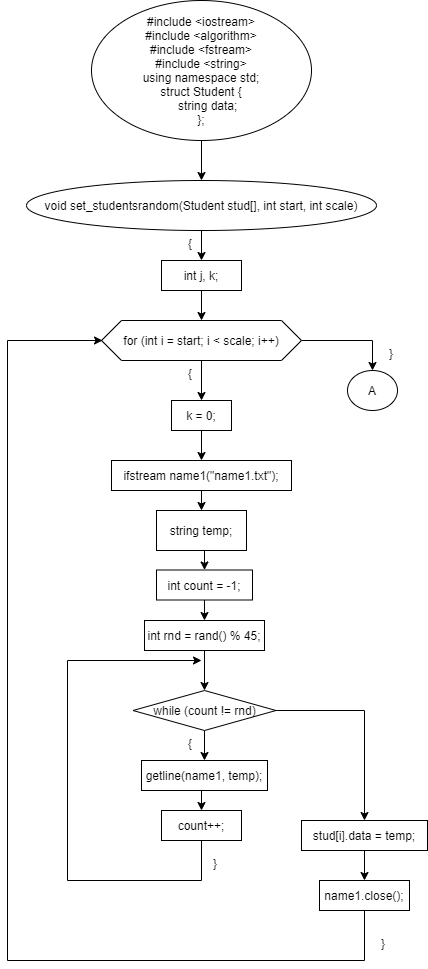
1. В программе были использованы следующие типы данных:
   1. Тип данных int для хранения количества элементов и для счетчика по циклу;

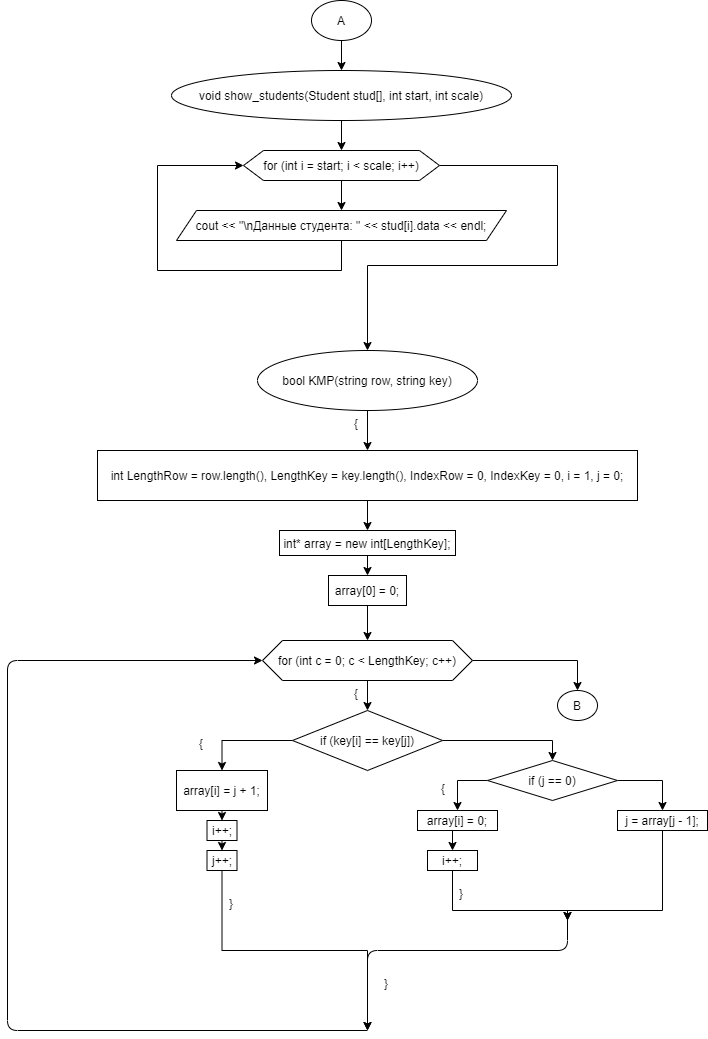


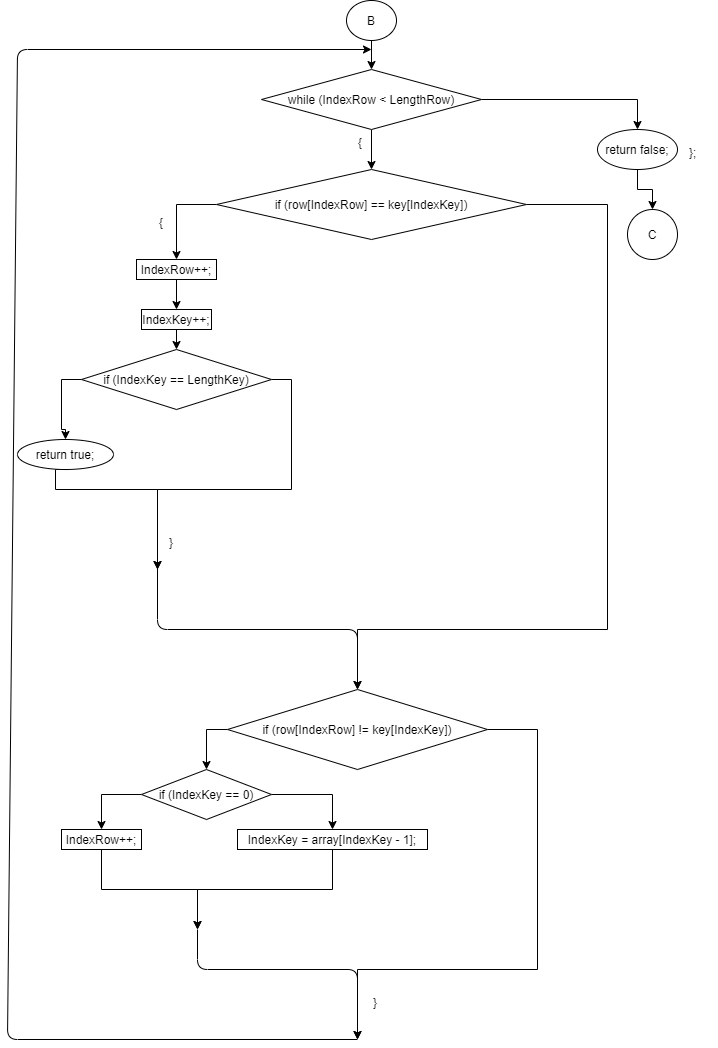
* 1. Тип данных string для хранения данных о студентах;

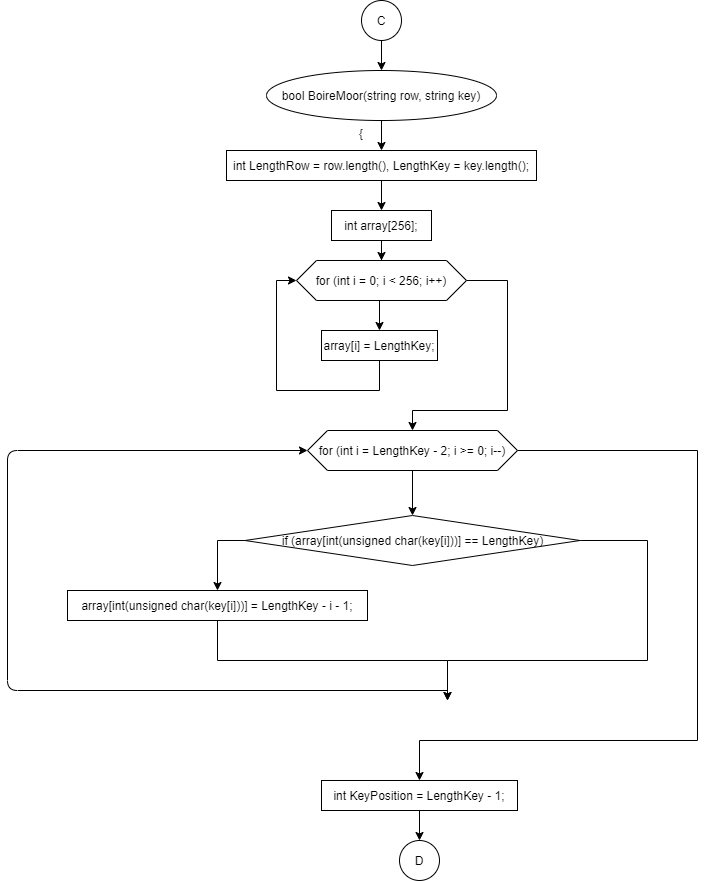


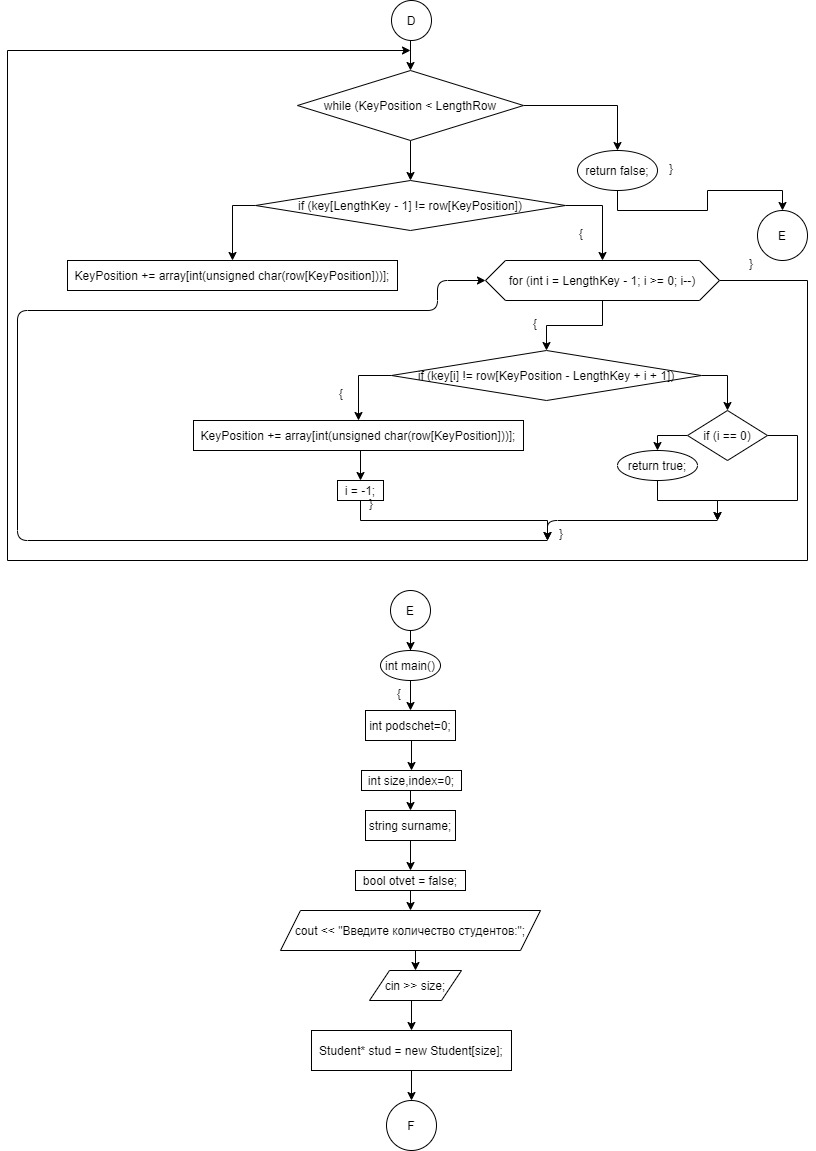
**Блок—схема программы**

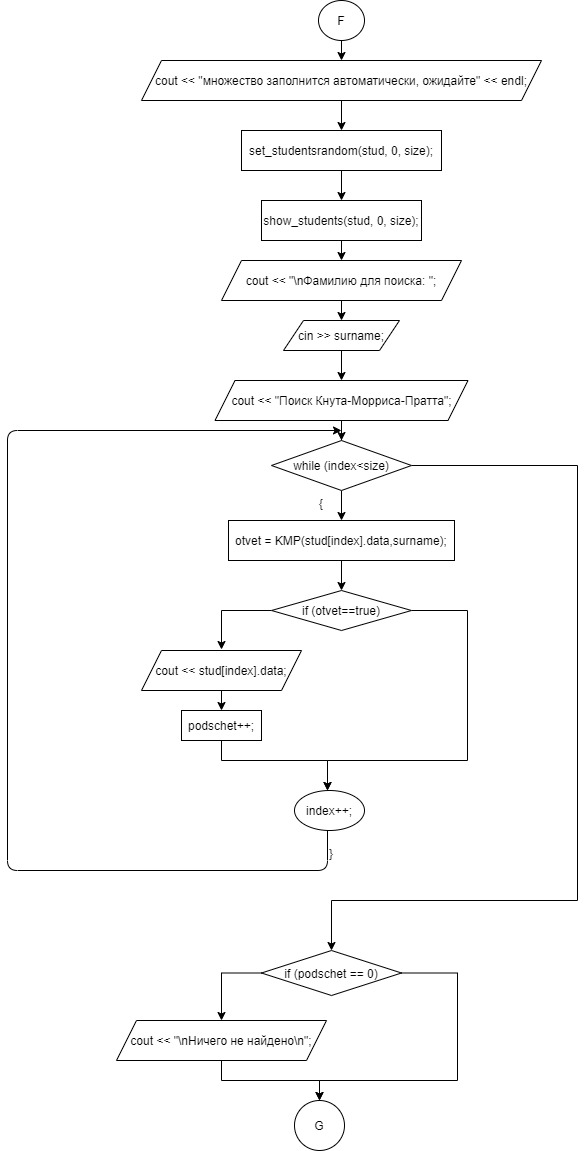


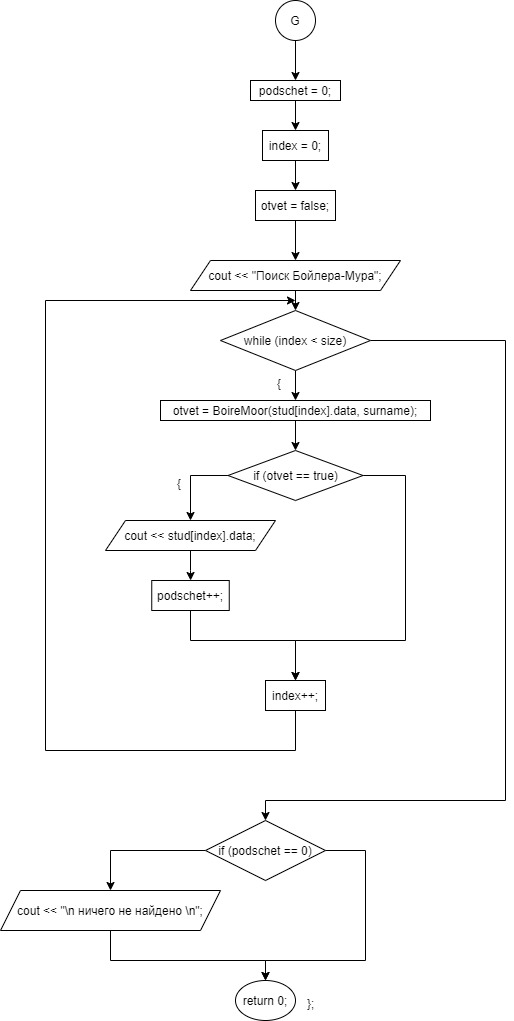












**Решение**

#include <iostream>

#include <algorithm>

#include <fstream>

#include <string>

using namespace std;

struct Student {

string data;

};

void set\_studentsrandom(Student stud[], int start, int scale) {

int j, k;

for (int i = start; i < scale; i++) {

k = 0;

ifstream name1("name1.txt");

string temp;

int count = -1;

int rnd = rand() % 45;

while (count != rnd)

{

getline(name1, temp);

count++;

}

stud[i].data = temp;

name1.close();

}

}

void show\_students(Student stud[], int start, int scale) {

for (int i = start; i < scale; i++) {

cout << "\nДанные студента: " << stud[i].data << endl;

}

}

bool KMP(string row, string key) {

int LengthRow = row.length(), LengthKey = key.length(), IndexRow = 0, IndexKey = 0, i = 1, j = 0;

int\* array = new int[LengthKey];

array[0] = 0;

for (int c = 0; c < LengthKey; c++)

{

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

array[i] = j + 1;

i++;

j++;

}

else if (j == 0) {

array[i] = 0;

i++;

}

else {

j = array[j - 1];

}

}

while (IndexRow < LengthRow)

{

if (row[IndexRow] == key[IndexKey])

{

IndexRow++;

IndexKey++;

if (IndexKey == LengthKey) {

return true;

}

}

if (row[IndexRow] != key[IndexKey])

{

if (IndexKey == 0) {

IndexRow++;

}

else {

IndexKey = array[IndexKey - 1];

}

}

}

return false;

};

bool BoireMoor(string row, string key) {

int LengthRow = row.length(), LengthKey = key.length();

int array[256];

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

array[i] = LengthKey;

}

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

if (array[int(unsigned char(key[i]))] == LengthKey) {

array[int(unsigned char(key[i]))] = LengthKey - i - 1;

}

}

int KeyPosition = LengthKey - 1;

while (KeyPosition < LengthRow) {

if (key[LengthKey - 1] != row[KeyPosition]) {

KeyPosition += array[int(unsigned char(row[KeyPosition]))];

}

else {

for (int i = LengthKey - 1; i >= 0; i--) {

if (key[i] != row[KeyPosition - LengthKey + i + 1]) {

KeyPosition += array[int(unsigned char(row[KeyPosition]))];

i = -1;

}

else if (i == 0) {

return true;

}

}

}

}

return false;

}

int main()

{

int podschet=0;

system("chcp 1251");

int size,index=0;

string surname;

bool otvet = false;

cout << "Введите количество студентов:";

cin >> size;

srand(time(NULL));

while (size <= 0)

{

cout << "Введите корректное количество студентов:";

cin >> size;

}

cout << endl;

Student\* stud = new Student[size];

cout << endl;

cout << "множество заполнится автоматически, ожидайте" << endl;

system("pause");

set\_studentsrandom(stud, 0, size);

show\_students(stud, 0, size);

cout << "\nФамилию для поиска: ";

cin >> surname;

cout << "Поиск Кнута-Морриса-Пратта";

cout << endl;

while (index<size) {

otvet = KMP(stud[index].data,surname);

if (otvet==true){

cout << stud[index].data;

cout << endl;

podschet++;

}

index++;

}

if (podschet == 0) {

cout << "\nНичего не найдено\n";

}

podschet = 0;

index = 0;

otvet = false;

cout << "Поиск Бойлера-Мура";

cout << endl;

while (index < size) {

otvet = BoireMoor(stud[index].data, surname);

if (otvet == true) {

cout << stud[index].data;

cout << endl;

podschet++;

}

index++;

}

if (podschet == 0) {

cout << "\n ничего не найдено \n";

}

delete[] stud;

};

**Скриншоты**

