Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

**«Пермский национальный исследовательский политехнический университет»**

Электротехнический факультет

Кафедра **«Информационные технологии и автоматизированные системы»**

направление подготовки: 09.03.04 - «Программная инженерия»

Лабораторная работа

По теме **«Сложные поиски»**

Выполнял:

студент группы РИС-24-1б

Морозова Н.С.

Проверял:

Доцент кафедры ИТАС

Полякова О.А.

Постановка задачи:

1. Дана некоторая срока и подстрока.
2. Двумя методами поиска определить, присутствует ли эта подстрока в данной строка.
3. Вывести результат.

Анализ задачи:

1. Предусмотреть ручной ввод строки и образа для двух методов поиска.
2. Для каждого метода написать функцию.
3. Применить их на данных строках.
4. Вывести ответ на вопрос: есть ли данный образ в нашей строке? И если есть, вывести номер, с какого символа в строке находится данный образ.

Решение

Код

#include <iostream>

#include <string>

#include <Windows.h>

using namespace std;

const int CHAR\_NUM = 256;

void calcBadT(const char\* pattern, const int patSize, int badT[CHAR\_NUM])

{

for (int i = 0; i < CHAR\_NUM; i++)

badT[i] = -1;

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

badT[(unsigned char)pattern[i]] = i;

}

void Boyer\_Moore(const char\* text, const char\* pattern)

{

int patSize = strlen(pattern);

int textSize = strlen(text);

if (patSize == 0 || textSize == 0 || patSize > textSize)

{

cout << "Некорректный ввод" << endl;

return;

}

int badT[CHAR\_NUM];

calcBadT(pattern, patSize, badT);

bool found = false;

int shift = 0;

while (shift <= (textSize - patSize))

{

int j = patSize - 1; //с последнего символа

while (j >= 0 && pattern[j] == text[shift + j])

j--;

if (j < 0)

{

cout << "Подстрока найдена по номеру: " << shift+1 << endl;

found = true;

shift += (shift + patSize < textSize) ? patSize - badT[(unsigned char)text[shift + patSize]] : 1;

}

else

shift += max(1, j - badT[(unsigned char)text[shift + j]]);

}

if (!found)

cout << "Подстрока не найдена" << endl;

}

void computeLPS(const char\* pat, int pat\_size, int\* lps)

{

int len = 0; // Длина предыдущего наибольшего суффикса

lps[0] = 0;

for (int i = 1; i < pat\_size; )

{

if (pat[i] == pat[len])

{

len++;

lps[i] = len;

i++;

}

else

{

if (len != 0)

len = lps[len - 1]; // Откатываемся назад

else

lps[i] = 0;

i++;

}

}

}

void Knut\_Moris\_Pratt(const char\* str, const char\* pat)

{

int str\_size = strlen(str);

int pat\_size = strlen(pat);

if (pat\_size == 0 || str\_size == 0 || pat\_size > str\_size)

{

cout << "Некорректный ввод" << endl;

return;

}

// Выделяем память для LPS-массива

int\* lps = new int[pat\_size];

computeLPS(pat, pat\_size, lps);

int str\_idx = 0, pat\_idx = 0;

bool found = false;

while (str\_idx < str\_size)

{

if (pat[pat\_idx] == str[str\_idx])

{

str\_idx++;

pat\_idx++;

// Полное совпадение

if (pat\_idx == pat\_size)

{

cout << "Подстрока найдена по номеру: " << str\_idx - pat\_idx + 1 << endl;

found = true;

pat\_idx = lps[pat\_idx - 1]; // Сдвигаем паттерн

}

}

else

{

if (pat\_idx != 0)

pat\_idx = lps[pat\_idx - 1]; // Используем LPS для сдвига

else

str\_idx++; // Просто двигаемся по тексту

}

}

if (!found)

cout << "Подстрока не найдена" << endl;

delete[] lps; // Освобождаем память

}

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

char str[100], patr[100];

cout << "Введите строку:" << endl;

cin >> str;

cout << "Введите образ для поиска методом Бойера-Мура:" << endl;

cin >> patr;

Boyer\_Moore(str, patr);

cout << "Введите строку:" << endl;

cin >> str;

cout << "Введите образ для поиска методом Кнута-Мориса-Пратта:" << endl;

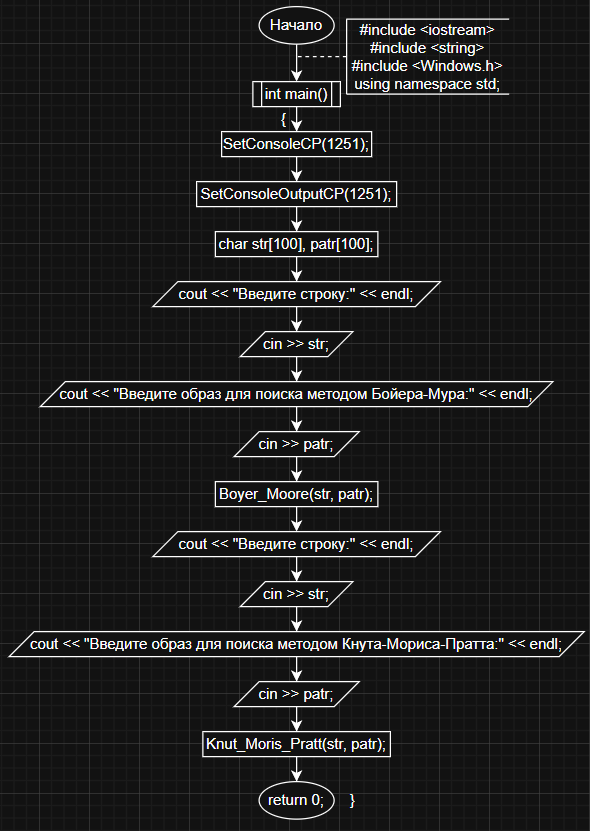
cin >> patr;

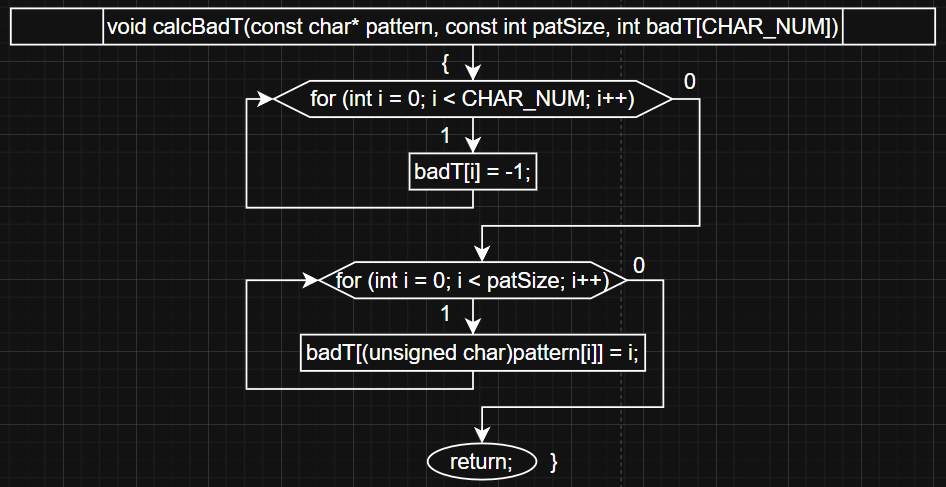
Knut\_Moris\_Pratt(str, patr);

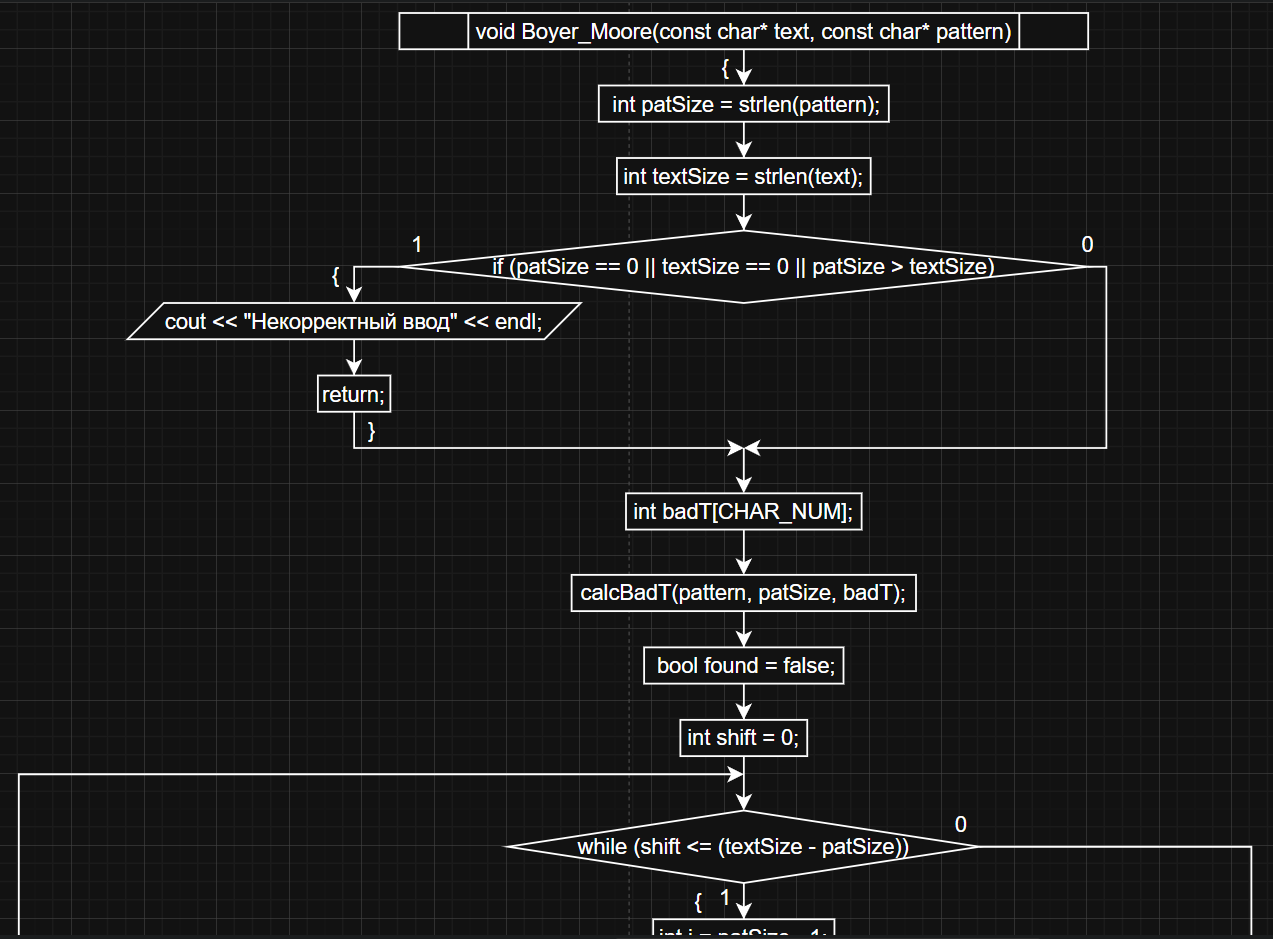
return 0;

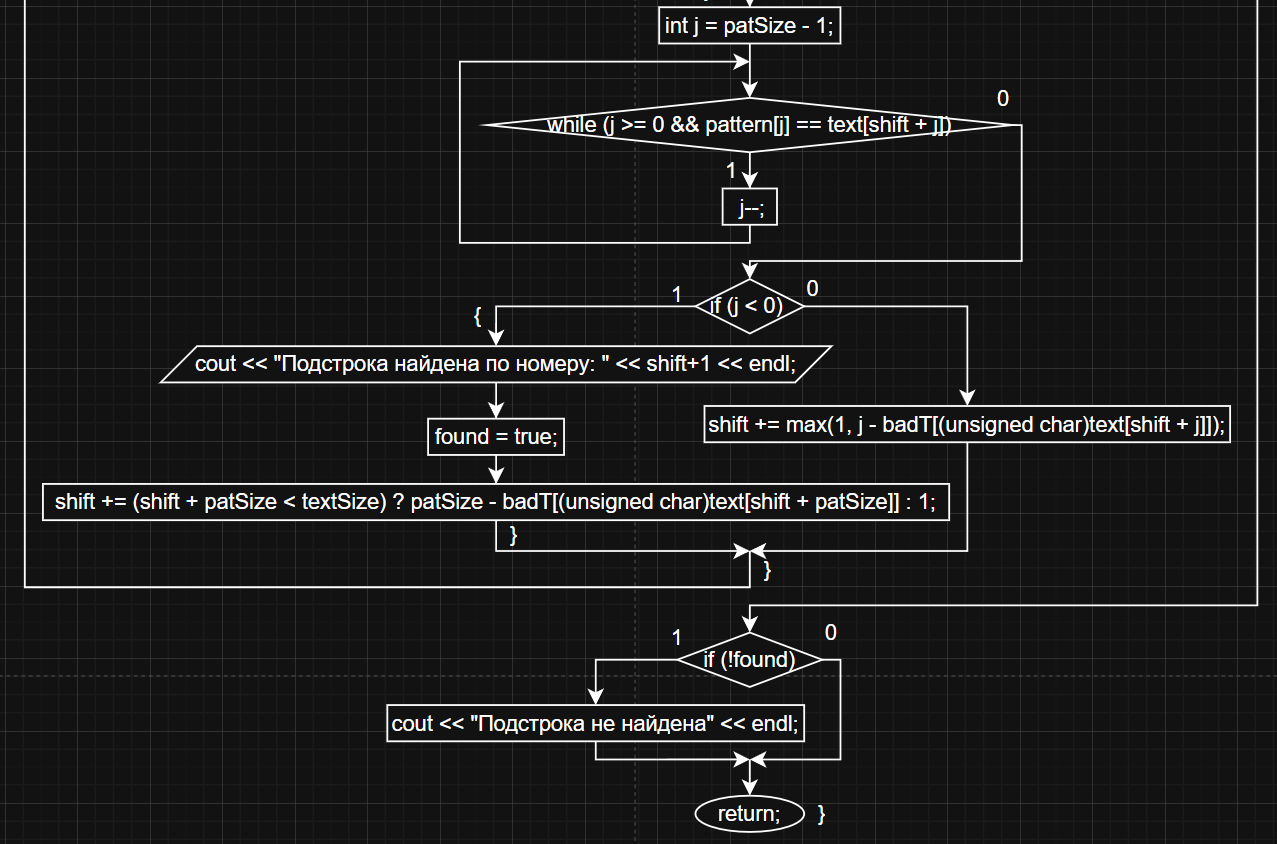
}

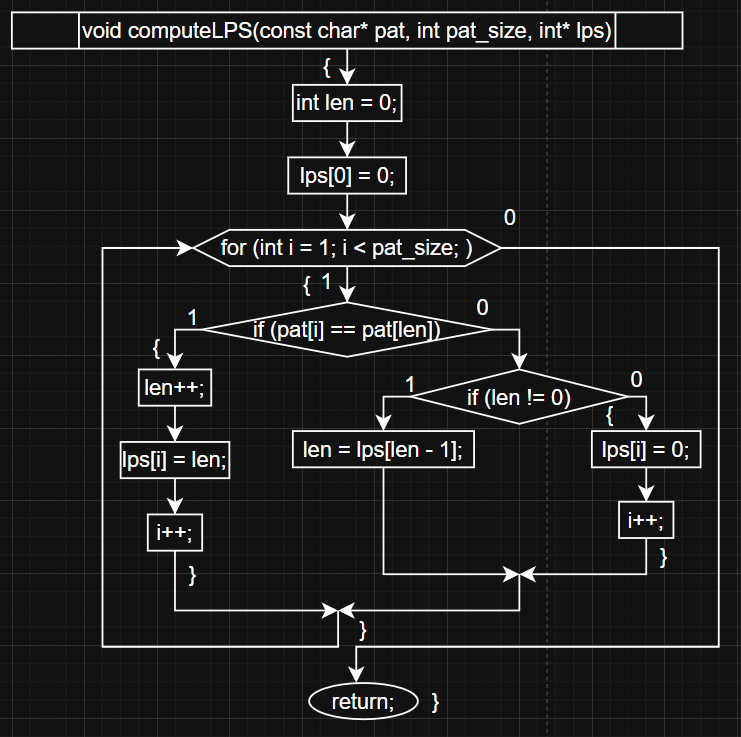
Блок-схема

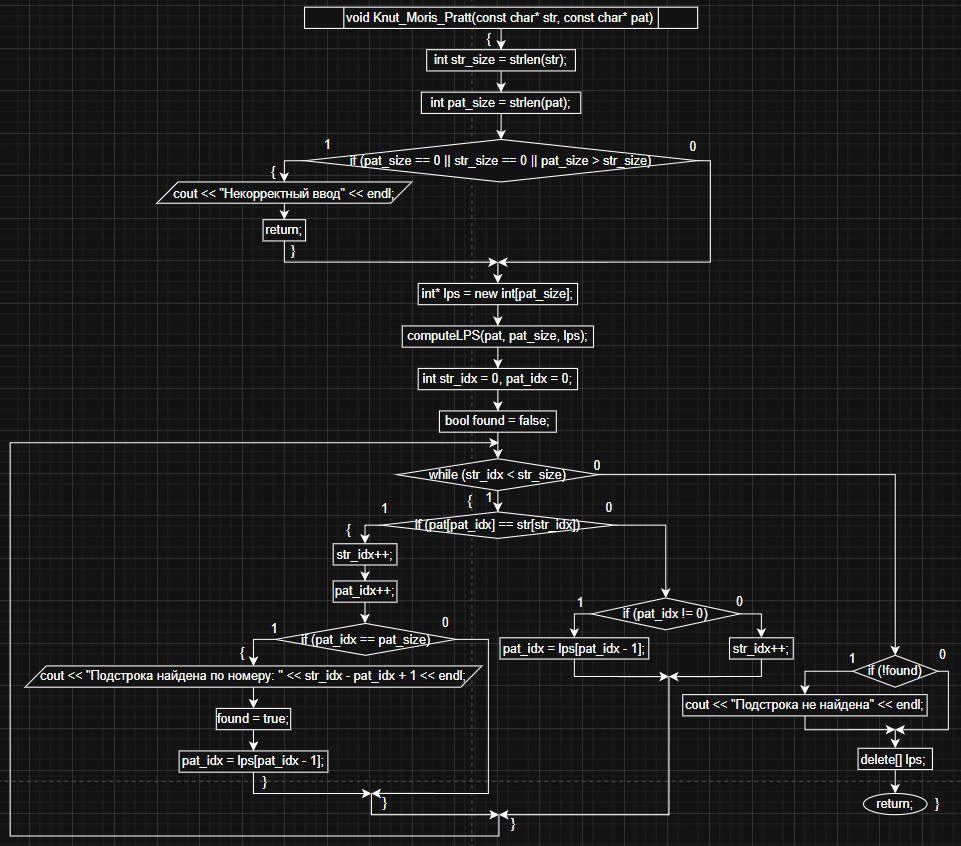












Результат работы программы:

