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**

**«Сложные поиски»**

Выполнил студент гр. РИС-24-1б

Иванов Семен Сергеевич

Проверил:

Доц. каф. ИТАС

Ольга Андреевна Полякова

(оценка) (подпись)

(дата)

**Постановка задачи**

Реализовать поиск при помощи алгоритмов Бойера-Мура и Кнута-Морриса-Пратта.

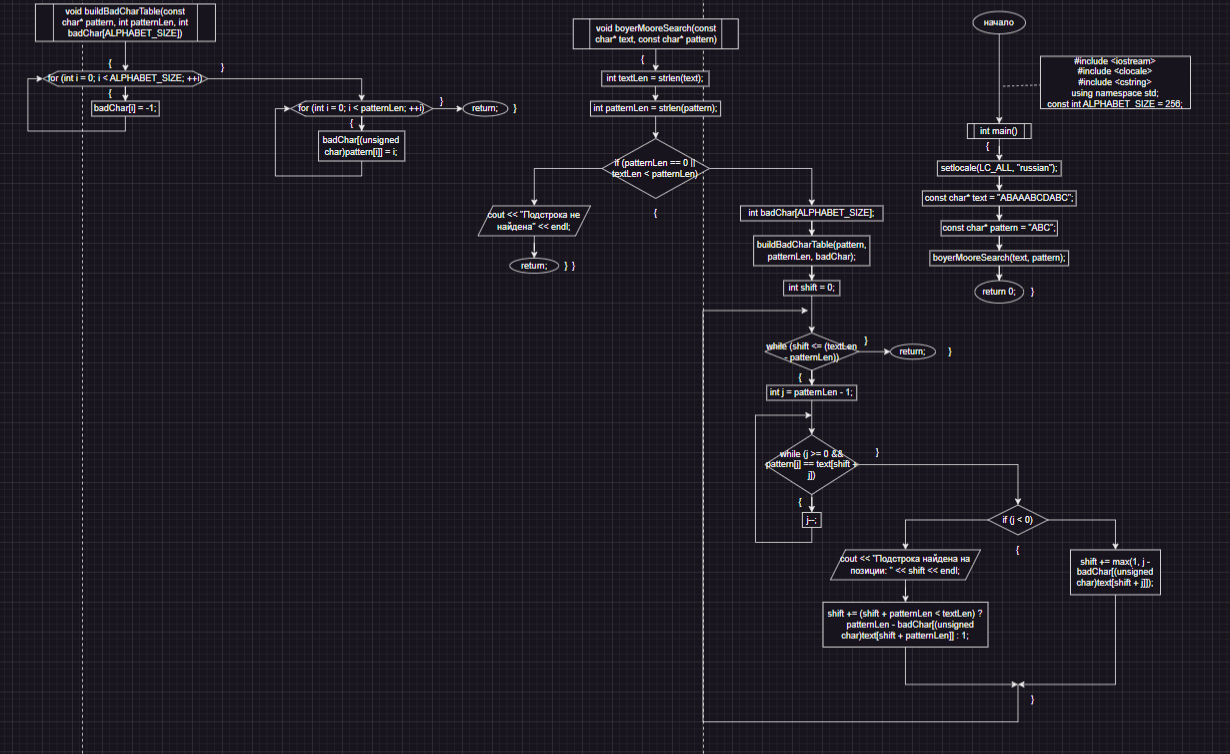
**Анализ задачи**

1. Создать строку символов и искомое их сочетание.

2. Написать функцию поиска при помощи одного из алгоритмов.

3. Вызвать функцию в main().

**Блок-схема(Бойер-Мур)**

****

**Код(Бойер-Мур)**

#include <iostream>

#include <clocale>

#include <cstring>

using namespace std;

const int ALPHABET\_SIZE = 256;

void buildBadCharTable(const char\* pattern, int patternLen, int badChar[ALPHABET\_SIZE]) {

for (int i = 0; i < ALPHABET\_SIZE; ++i) {

badChar[i] = -1;

}

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

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

}

}

void boyerMooreSearch(const char\* text, const char\* pattern) {

int textLen = strlen(text);

int patternLen = strlen(pattern);

if (patternLen == 0 || textLen < patternLen) {

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

return;

}

int badChar[ALPHABET\_SIZE];

buildBadCharTable(pattern, patternLen, badChar);

int shift = 0;

while (shift <= (textLen - patternLen)) {

int j = patternLen - 1;

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

j--;

}

if (j < 0) {

cout << "Подстрока найдена на позиции: " << shift << endl;

shift += (shift + patternLen < textLen) ?

patternLen - badChar[(unsigned char)text[shift + patternLen]] : 1;

}

else {

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

}

}

}

int main() {

setlocale(LC\_ALL, "russian");

const char\* text = "ABAAABCDABC";

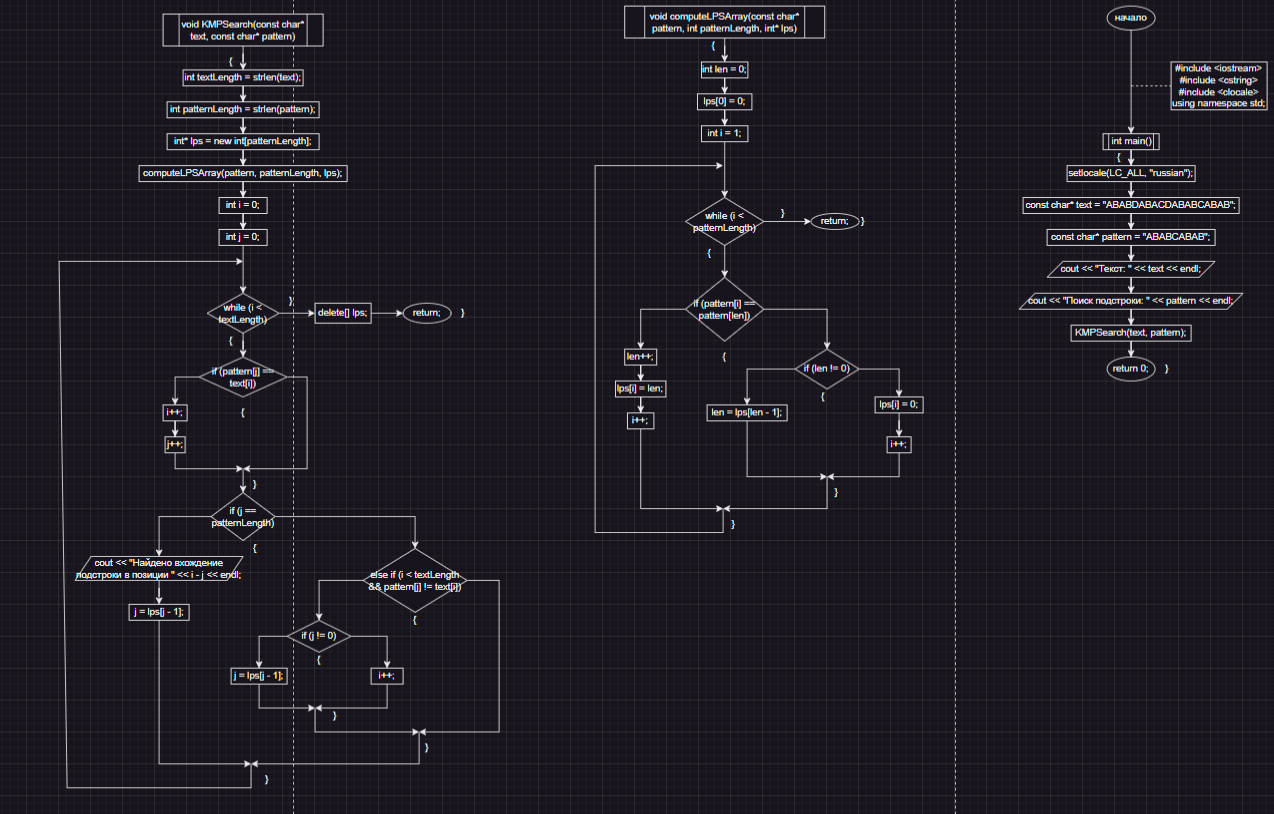
const char\* pattern = "ABC";

boyerMooreSearch(text, pattern);

return 0;

}

**Блок-схема(Кнут-Морис-Пратт)**

****

**Код(Кнут-Морис-Пратт)**

#include <iostream>

#include <cstring>

#include <clocale>

using namespace std;

void computeLPSArray(const char\* pattern, int patternLength, int\* lps) {

int len = 0;

lps[0] = 0;

int i = 1;

while (i < patternLength) {

if (pattern[i] == pattern[len]) {

len++;

lps[i] = len;

i++;

}

else {

if (len != 0) {

len = lps[len - 1];

}

else {

lps[i] = 0;

i++;

}

}

}

}

void KMPSearch(const char\* text, const char\* pattern) {

int textLength = strlen(text);

int patternLength = strlen(pattern);

int\* lps = new int[patternLength];

computeLPSArray(pattern, patternLength, lps);

int i = 0;

int j = 0;

while (i < textLength) {

if (pattern[j] == text[i]) {

i++;

j++;

}

if (j == patternLength) {

cout << "Найдено вхождение подстроки в позиции " << i - j << endl;

j = lps[j - 1];

}

else if (i < textLength && pattern[j] != text[i]) {

if (j != 0) {

j = lps[j - 1];

}

else {

i++;

}

}

}

delete[] lps;

}

int main() {

setlocale(LC\_ALL, "russian");

const char\* text = "ABABDABACDABABCABAB";

const char\* pattern = "ABABCABAB";

cout << "Текст: " << text << endl;

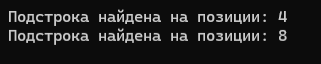
cout << "Поиск подстроки: " << pattern << endl;

KMPSearch(text, pattern);

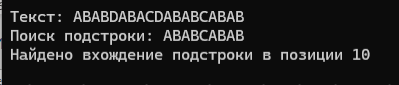
return 0;

}

**Скриншот решения(Бойер-Мур)**

****

**Скриншот решения(Кнут-Морис-Пратт)**

****

**github**

https://github.com/Lagus645/HardSearch