Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное учреждение высшего образования

«Пермский национальный исследовательский политехнический университет»

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы»

**О Т Ч Ё Т**

**по лабораторной работе**

Дисциплина: «Основы алгоритмизации и программирования»

Выполнил

студент группы ИВТ 21-1Б

Лебедев Т.А.

Проверил

Яруллин Д.В.

Пермь, 2022

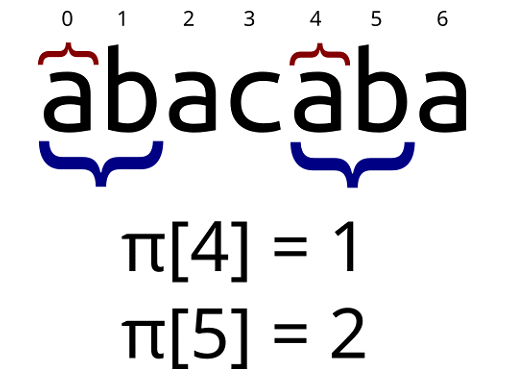
**Постановка задачи:**

Дана строка и подстрока, которую нужно найти. Выполнить задание в соответствии со своим вариантом, используя указанный метод поиска. Значения строки и подстроки любые (если иного не указано по заданию).  
Найти подстроку strl длиной k символов и заменить ее на подстроку, следующую после нее длиной k символов (например, подстрока ab в строке cvabghuj заменится: cvghghuj)

**Анализ задачи:**

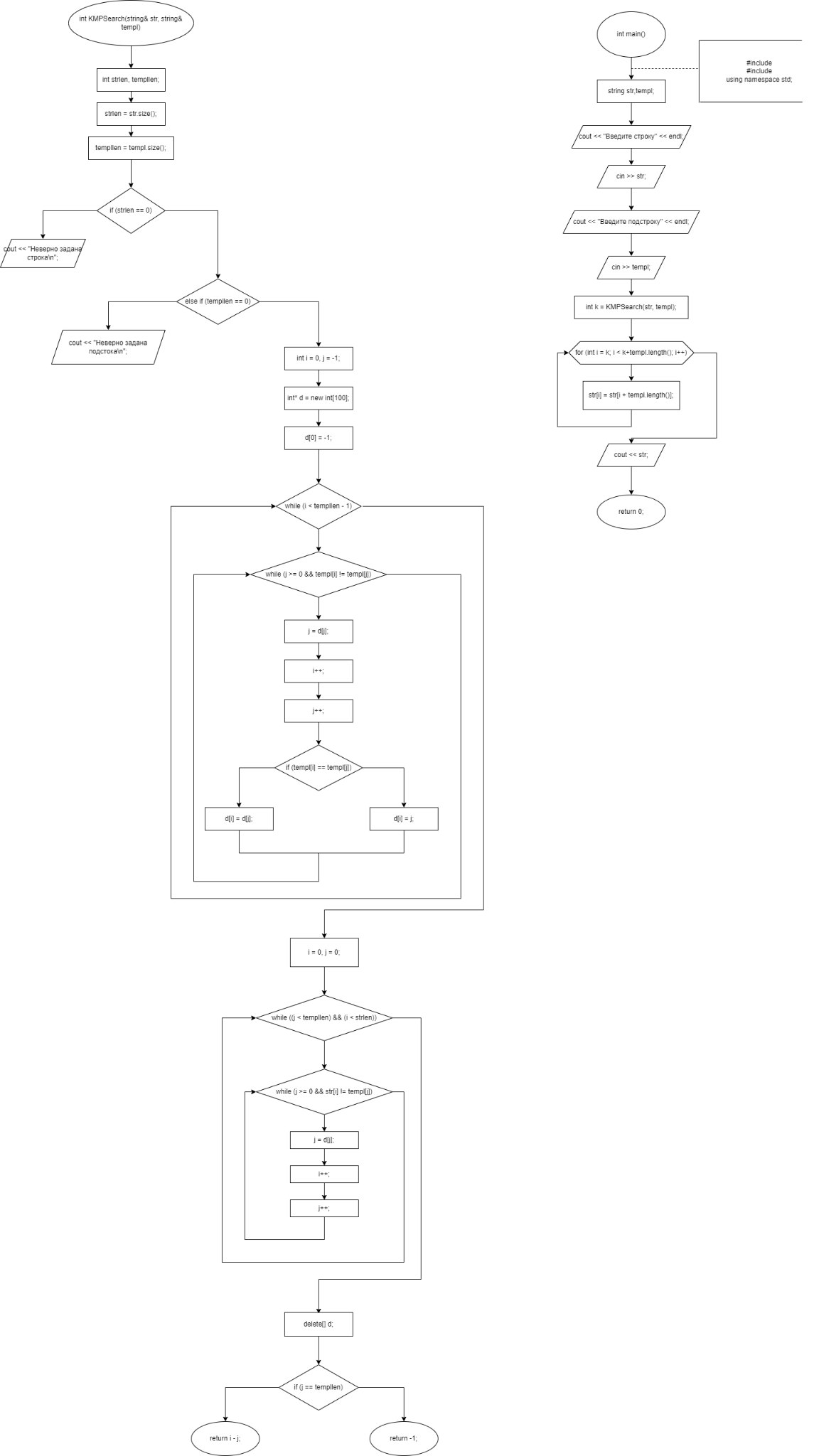
⦁ Поиск Кнута — Морриса — Пратта — эффективный алгоритм, осуществляющий поиск подстроки в строке. Время работы алгоритма линейно зависит от объёма входных данных, то есть разработать асимптотически более эффективный алгоритм невозможно.

⦁ Префикс-функция от строки s равна массиву π, где π[i] обозначает длину максимального префикса строки s[0..i], совпадающего с её суффиксом. Тривиальные случаи (префикс равен суффиксу и равен всей строке) не учитываются.



На изображении обозначены равные подстроки, длина которых равна значению префикс-функции в данной позиции. Префикс-функция от всей строки “abacaba” равна {0,0,1,0,1,2,3}. π[0]=π[1]=0 так как строки “a” и “ab” являются тривиальными, и поэтому не учитываются.

**Блок-схема:**



**Код программы:**

#include<iostream>

#include<string>

using namespace std;

int KMPSearch(string& str, string& templ)

{

int strlen, templlen;

strlen = str.size();

templlen = templ.size();

if (strlen == 0)

cout << "Неверно задана строка\n";

else if (templlen == 0)

cout << "Неверно задана подстока\n";

else

{

int i = 0, j = -1;

int\* d = new int[100];

d[0] = -1;

while (i < templlen - 1)

{

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

j = d[j];

i++;

j++;

if (templ[i] == templ[j])

d[i] = d[j];

else

d[i] = j;

}

i = 0, j = 0;

while ((j < templlen) && (i < strlen))

{

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

j = d[j];

i++;

j++;

}

delete[] d;

if (j == templlen)

return i - j;

else

return -1;

}

}

int main()

{

string str,templ;

setlocale(LC\_ALL, "ru");

cout << "Введите строку" << endl;

cin >> str;

cout << "Введите подстроку" << endl;

cin >> templ;

int k = KMPSearch(str, templ);

for (int i = k; i < k+templ.length(); i++)

{

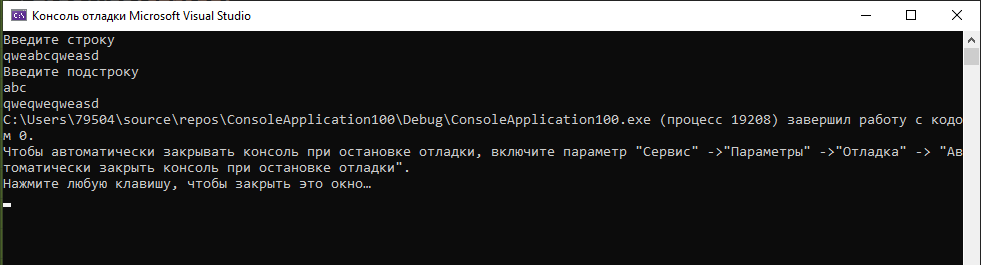
str[i] = str[i + templ.length()];

}

cout << str;

}

**Скриншоты результатов:**

****