**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина) Кафедра МО ЭВМ**

**ОТЧЕТ**

**по лабораторной работе №4**

**по дисциплине «Построение и анализ алгоритмов»**

**Тема: Алгоритм Кнута-Морриса-Пратта**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент гр. 7383 | |  | Сычевский Р.А. |  |
| Преподаватель | |  | Жангиров Т.Р. |  |
|  |  |
|  |  |  |  |  |

Санкт-Петербург

2019

**Содержание**

Цель работы 3

Реализация задачи 4

Исследование алгоритма 5

Тестирование 6

1. Процесс тестирования 6

2. Результаты тестирования 6

Вывод 7

Приложение А. Тестовые случаи 8

Приложение Б. Исходный код 9

**Цель работы**

Цель работы: познакомиться с алгоритмом поиска подстроки в строке Кнута-Морриса-Пратта, создать программу, осуществляющую поиск подстроки в строке с помощью алгоритма Кнута-Морриса-Пратта и проверку строк на циклический сдвиг.

Формулировка задачи: Реализуйте алгоритм КМП и с его помощью для заданных шаблона P (|P|≤15000) и текста T (|T|≤5000000) найдите все вхождения P в T. Определить, является ли А циклическим сдвигом В (это значит, что А и В имеют одинаковую длину и А состоит из суффикса В, склеенного с префиксом В). Например, defabc является циклическим сдвигом abcdef. Вариант 2. Оптимизация по памяти: программа должна требовать O(m) памяти, где m - длина образца. Это возможно, если не учитвать память, в которой хранится строка поиска.

**Реализация задачи**

Программу было решено писать на языке программирования C++.

Для реализации поставленной задачи были созданы функции

prefix\_function и find\_shift.

Функция vector<int> prefix\_function(string s, vector<int> &pi) осуществляет вычисление префикс функции для алгоритма Кнута-Морриса-Пратта. В выходном массиве символы с результатом, равным длине строки p являются концом ее вхождения в данную строку.

Функция int find\_shift(string s, string b) возвращает индекс начала строки a в строке или -1, если строка не является сдвигом.

Исходный код программы представлен в приложении Б.

**Исследование алгоритма**

Сложность алгоритма Кнута-Морриса-Пратта линейная и совпадает со сложностью вычисления префикс-функции. Сложность алгоритма поиска цикла равно O(n2), где n - длина строк.

**Тестирование**

1. **Процесс тестирования**

Программа собрана в операционной системе Ubuntu 18.04.2 LTS bionic компилятором g++ version 7.3.0 (Ubuntu 7.3.0-27ubuntu1~18.04). В других ОС

* компиляторах тестирование не проводилось.
  1. **Результаты тестирования**
* результате тестирования программы ошибок выявлено не было. Тестовые случаи представлены в приложении А.

**Вывод**

* + ходе выполнения данной работы был изучен метод поиска подстроки
* строке с помощью алгоритма Кнута-Морриса-Пратта. Была написана программа, применяющая алгоритм Кнута-Морриса-Пратта для поиска подстроки в строке, а также осуществляющая поиск циклического сдвига в двух строках. Сложность алгоритма Кнута-Морриса-Пратта составляет O(n), а алгоритма поиска цикла - O(n2).

**ПРИЛОЖЕНИЕ А. ТЕСТОВЫЕ СЛУЧАИ**

|  |  |  |
| --- | --- | --- |
| Входные данные | Результат |  |
|  |  |  |
| ab | 0,2 |  |
| abab |  |
|  |  |
|  |  |  |
| aa | 1,2,3 |  |
| caaaa |  |
|  |  |
|  |  |  |
| caa | 0 |  |
| caaaa |  |
|  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| Входные данные | Результат |  |
|  |  |  |
| defabc | 3 |  |
| abcdef |  |
|  |  |
|  |  |  |
| aaa | -1 |  |
| aa |  |
|  |  |
|  |  |  |
| abcdef | -1 |  |
| cdefba |  |
|  |  |
|  |  |  |

**ПРИЛОЖЕНИЕ Б. ИСХОДНЫЙ КОД**

#include <iostream>

#include <vector>

using namespace std;

vector<int> prefix\_function (string s, vector<int> &pi) {

    for (int i=1; i<s.length(); ++i) {

        int j = pi[i-1];

        while (j > 0 && s[i] != s[j])

            j = pi[j-1];

        if (s[i] == s[j]) ++j;

        pi[i] = j;

        pi.push\_back(0);

    }

    return pi;

}

int find\_shift(string s, string f){

    int flag = 0;

    for(int i = 0; i < s.length(); i++){

        flag = 1;

        if (s[0] == f[i]){

for(int j = 0; j < s.length(); j++)

{

if (s[j] != f[(i + j) % s.length()])

{

flag = 0;

break;

}

}

if (flag)

return i;

}

    }

    return -1;

}

int main(){

    string f;

    cin >> f;

    int size = f.length();

    string s;

    cin >> s;

cout << "Выберите задание:" << endl;

cout << "1 - поиск подстрки в стрке" << endl;

cout << "2 - поиск сдвига" << endl;

int task;

cin >> task;

vector<int> pi;

switch(task){

case 1:{

f = f + "&";

     f = f + s;

     pi.push\_back(0);

     pi.push\_back(0);

     pi = prefix\_function(f, pi);

int flag = 0;

for(int i = 0; i < f.length(); i++){

         if(pi[i] == size){

             if(!flag){

                 cout << i-(2\*size);

                 flag++;

             }else

                 cout << "," << i-(2\*size);

     }

     }

cout << endl;

     if(!flag)

cout << -1 << endl;

break;

}

case 2:{

if(f.length() != s.length()){

         cout << -1 << endl;

return 0;

}

cout << find\_shift(s, f) << endl;

break;

}

}

return 0;

}