**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра математического обеспечения и применения ЭВМ**

**ОТЧЕТ**

**по лабораторной работе №**4

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

**Тема: Алгоритмы на графах**

|  |  |  |
| --- | --- | --- |
| Студент гр. 8304 |  | Самакаев Д.И. |
| Преподаватель |  | Размочаева Н.В. |

Санкт-Петербург

2019

**Вариант 2.**

## Цель работы.

Построение и анализ алгоритма Кнута-Морриса-Пратта на основе решения задачи о нахождении циклического сдвига строки.

## Основные теоретические положения.

Заданы две строки A (∣A∣≤5000000) и B (∣B∣≤5000000).  
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.  
Вход:  
Первая строка - A   
Вторая строка - B   
Выход:  
Если A вляется циклическим сдвигом B, индекс начала строки B в A, иначе вывести −1. Если возможно несколько сдвигов вывести первый индекс.

## Описание алгоритма.

Для вычисления циклического сдвига к строке, в которой производится поиск, приписывается её копия и к полученной строке применяется алгоритм КМП. Вычисляется префикс для второй строки. По памяти алгоритм работает за О(2m), где m – длина строки, в которой производится поиск.

**Функции и структуры данных.**

std::vector<size\_t> get\_prefix(std::string sample) – функция вычисляет префикс для строки – образца и возвращает его.

int KMPSearch(std::string text, std::string sample) – функция при помощи алгоритма Кнута\_Морриса\_Пратта вычисляет является ли строка sample циклическим сдвигом строки text, и, если да, возвращает индекс смещения строки sample относительно строки text.

Реализован файловый и консольный вводы и выводы.

**Вывод промежуточной информации.**

Во время основной части работы алгоритма происходит вывод префикса.

**Тестирование.**

Таблица 1 – Результаты тестирование

|  |  |
| --- | --- |
| **Ввод** | **Вывод** |
| defabc  abcdef | 3 |
| efccaa  cbaads | -1 |
| caa  aac | 2 |
| GggHgg  gggggg | -1 |

# Вывод.

В ходе работы был построен и анализирован алгоритм Кнута-Морриса-Пратта на основе решения задачи о нахождении циклического сдвига строки.

**ПРИЛОЖЕНИЕ А.  
ИСХОДНЫЙ КОД**

#include <iostream>

#include <vector>

#include <string>

#include <algorithm>

#include <fstream>

std::vector<size\_t> get\_prefix(std::string sample) {

std::vector<size\_t> prefix(sample.length(), 0);

for (size\_t i = 1; i < sample.length(); i++) {

size\_t j = 0;

while (i + j < sample.length() && sample[j] == sample[i + j]) {

prefix[i + j] = std::max(prefix[i + j], j + 1);

j++;

}

}

return prefix;

}

int KMPSearch(std::string text, std::string sample) {

std::vector<size\_t> found;

std::vector<size\_t> prefix = get\_prefix(sample);

for (size\_t i = 0; i < prefix.size(); i++)

std::cout << prefix[i];

std::cout<<std::endl;

size\_t i = 0;

size\_t j = 0;

text += text;

while (i < text.length()) {

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

i++;

j++;

if (j == sample.length())

return i - j;

}

if (j == sample.length())

j = prefix[j - 1];

if (i < text.length() && sample[j] != text[i]) {

if (j != 0) {

j = prefix[j - 1];

}

else {

i++;

}

}

}

return -1;

}

void console\_input() {

std::cout << "Please, Enter the text\n";

std::string text;

std::string sample;

std::string out\_file\_name = "out.txt";

size\_t result;

std::cin >> text;

std::cout << "Please, Enter the sample\n";

std::cin >> sample;

result = KMPSearch(text, sample);

std::cout << result << std::endl;

std::ofstream out\_file;

out\_file.open(out\_file\_name);

if (!out\_file.is\_open()) {

std::cout << "Error! Output file isn't open" << std::endl;

}

out\_file << result << std::endl;

}

void file\_input(char\*& argv) {

std::ifstream file;

std::string testfile = argv;

file.open(testfile);

std::string out\_file\_name = "out.txt";

size\_t result;

if (!file.is\_open()) {

std::cout << "Error! File isn't open" << std::endl;

return;

}

std::string text;

std::string sample;

while (!file.eof()) {

file >> text;

file >> sample;

}

result = KMPSearch(text, sample);

std::cout << result << std::endl;

std::ofstream out\_file;

out\_file.open(out\_file\_name);

if (!out\_file.is\_open()) {

std::cout << "Error! Output file isn't open" << std::endl;

}

out\_file << result << std::endl;

}

int main(size\_t argc, char\*\* argv)

{

if (argc == 1)

console\_input();

else if (argc == 2)

file\_input(argv[1]);

}