**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: Алгоритм Кнута-Морриса-Пратта**

|  |  |  |
| --- | --- | --- |
| Студент гр. 8304 |  | Сани Заяд. |
| Преподаватель |  | Размочаева Н.В. |

Санкт-Петербург

2020

## Цель работы.

Реализовать алгоритм Кнута-Морриса-Пратта, найти индексы вхождения подстроки в строку, а также разработать алгоритм проверки двух строк на циклический сдвиг.

**Задание.**

Реализуйте алгоритм КМП и с его помощью для заданных шаблона P и текста T найдите все вхождения P в T.

Вход:

Первая строка – P

Вторая строка – T

Выход:

Индексы начал вхождений P в T, разделенных запятой, если P не входит в T, то вывести -1.

**Вариант 2.**

Оптимизация по памяти: программа должна требовать O(m) памяти, где m - длина образца. Это возможно, если не учитывать память, в которой хранится строка поиска.

## Описание алгоритма.

Для оптимизации алгоритма по памяти будем считывать текст посимвольно. Алгоритм использует префикс-функцию, которая считает, сколько символов совпало у префикса строки Pattern и у строки Text, заканчивающимся в i-ой позиции.

Сложность алгоритма O(m + n), где m – длина образца, n – длина строки в которой мы ищем.

## Описание основных структур данных и функций.

void getPatternPrefix(const std::string& pattern, std::vector<int>& prefix);

- функция, возвращающая префикс строки pattern.

void KMP();

- функция, находящая все вхождения подстроки pattern в строку text и выводящая индексы всех вхождений. Если вхождения не найдены, то выводится -1.

**Тестирование.**

Таблица 1 – Результат работы.

|  |  |
| --- | --- |
| **Ввод** | **Вывод** |
| ab  abab | 0,2 |
| ab  anybody | -1 |
| dangote  findmoneydangoteyeyey | 10 |
| ddd  aureowubdjdnasd | -1 |
| lt  ltltltltltltltltltltlsdfjtltlt | 0,2,4,6,8,10,12,14,16,18,26,28 |

# Вывод.

В ходе выполнения данной работы был реализован алгоритм Кнута-Морриса-Пратта, алгоритм проверки двух строк на циклический сдвиг, а также функция вычисления префикса строки.

**ПРИЛОЖЕНИЕ А.  
ИСХОДНЫЙ КОД**

#include <iostream>

#include <string>

#include <vector>

**void** getPatternPrefix(**const** std::string& pattern, std::vector<**int**>& prefix) {

**for** (**int** i = 1; i < pattern.size(); ++i) {

**int** index = prefix[i - 1];

**while** (index > 1 && pattern[index] != pattern[i])

index = prefix[index - 1];

**if** (pattern[index] == pattern[i])

index += 1;

prefix[i] = index;

}

}

**void** printPrefix(**const** std::vector<**int**>& prefix){

std::cout<<"Prefix: ";

**for** (**int** number : prefix){

std::cout << number << " ";

}

std::cout<<"\n";

}

**void** printResult(**const** std::vector<**int**>& ans){

std::cout<<"Answers : ";

**if**(!ans.empty()){

std::string separator;

**for** (**int** number : ans){

std::cout << separator <<number;

separator = ", ";

}

}**else**{

std::cout<<"-1";

}

std::cout<<"\n";

}

**void** KMP() {

std::cout << "Enter pattern: " << std::endl;

std::string pattern;

std::getline(std::cin, pattern);

std::vector<**int**> prefix(pattern.size());

getPatternPrefix(pattern,prefix);

printPrefix(prefix);

**int** index = 0;

**int** counter = 0;

std::vector<**int**> ans;

std::cout << "Enter text: " << std::endl;

**char** character;

**while** (std::cin.get(character) && character != '\n') {

counter++;

**while** (index > 0 && character != pattern[index])

index = prefix[index - 1];

**if** (character == pattern[index])

index++;

**if** (index == pattern.size()) {

ans.push\_back(counter - index + 1);

}

}

printResult(ans);

}

**int** main() {

KMP();

std::cout<<"\nTime Complexity O(m + n) ,where \nm – lenght of text, \nn – lenght of pattern\n";

**return** 0;

}