

**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. 7304

Сергеев И.Д.

Преподаватель

Филатов Ар.Ю.

Санкт-Петербург

2019

## Цель работы:

Исследовать алгоритм Кнута-Морриса-Пратта, его реализацию на языке c++ и его практическое применение.

## Задача:

---

1) Реализуйте алгоритм КМП и с его помощью для заданных шаблона  $PP$  ( $|P| \leq 15000$ ) и текста  $TT$  ( $|T| \leq 5000000$ ) найдите все вхождения  $PP$  в  $TT$ .

Вход:

Первая строка -  $PP$

Вторая строка -  $TT$

Выход:

индексы начал вхождений  $PP$  в  $TT$ , разделенных запятой, если  $PP$  не входит в  $TT$ , то вывести  $-1$ .

2) Заданы две строки  $A$  ( $|A| \leq 5000000$ ) и  $B$  ( $|B| \leq 5000000$ ).

Определить, является ли  $A$  циклическим сдвигом  $B$  (это значит, что  $A$  и  $B$  имеют одинаковую длину и  $A$  состоит из суффикса  $B$ , склеенного с префиксом  $B$ ). Например,  $defabc$  является циклическим сдвигом  $abcdef$ .

Вход:

Первая строка -  $A$

Вторая строка -  $B$

Выход:

Если  $A$  является циклическим сдвигом  $B$ , индекс начала строки  $B$  в  $A$ , иначе вывести  $-1$ . Если возможно несколько сдвигов вывести первый индекс.

## Основные теоретические положения:

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

## Ход работы:

1. Реализована префикс-функция

Данная функция необходима для нахождения образа, который входит в текст. Если есть совпадение символов в строке, увеличиваем индекс. Продолжаем пока строка не кончилась.

```
vector<int> prefix_func(string str) {  
    vector<int> entries(str.length());  
    entries[0] = 0;  
    for (int i = 1; i < str.length(); i++) {  
        int j = entries[i - 1];  
        while ((j > 0) && (str[i] != str[j]))  
            j = entries[j - 1];  
        if (str[i] == str[j])  
            ++j;  
        entries[i] = j;  
    }  
    return entries;  
}
```

## 2. Реализован алгоритм КМП

Функция реализует алгоритм КМП. Сравниваем символы строки и подстроки, если количество удачных сравнений равно размеру подстроки, то запоминаем индекс начала повторения. Для того, чтобы определить является ли строка циклическим сдвигом другой можно использовать тот же алгоритм КМП, но с небольшими поправками. Отличие заключается в том, что при достижении конца строки, мы не будем выходить из функции, а обнулим индекс. Выход из функции будет осуществляться в том случае, если циклический сдвиг будет найден или после прохода строки не будет найдено ни одного вхождения второй строки.

```
vector<int> KMP(string str, string substr) {  
    vector<int> p;  
    vector<int> result;  
    p = prefix_func(substr);  
    unsigned long k = 0, j = 0;  
    while (k < str.length()){  
        if (str[k] == substr[j]){  
            k++;  
            j++;  
        }  
    }  
}
```

```

        if (j == substr.size()){
            result.push_back(k-substr.size());
        }
    }
else if (j == 0){
    k++;
}
else{
    j = p[j-1];
}



}

return result;
}



```

## Результат:

1)

 stdin	<a href="#">copy</a>
ab abab	
 stdout	<a href="#">copy</a>
0,2	

2)

 stdin	<a href="#">copy</a>
defabc abcdef	
 stdout	<a href="#">copy</a>
3	

## **Вывод:**

Таким образом, в ходе данной лабораторной работы , была разработана программа, которая реализует алгоритм КМП. Программа вычисляет индексы вхождений подстроки в строку, а также проверяет, не является ли одна строка циклическим сдвигом второй.

В ходе данной работы столкнулся с проблемой построения префиксной функции и ее дальнейшим использованием в программе. Также было непросто реализовать выход из программы при реализации циклического сдвига.