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: Префикс-функция и алгоритм КМП.

Студент гр. 7304 _____ Субботин А.С.

Преподаватель _____ Филатов А.Ю.

Санкт-Петербург

2019

Цель работы

Изучить и реализовать на языке программирования C++ алгоритм Кнута-Морриса-Пратта, осуществляющий поиск подстроки в строке.

Формулировка задачи

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

Ход работы

В работе используется тип данных uint_least32_t из библиотеки cstdint. Такой выбор обусловлен ограниченностью доступной памяти и логичным стремлением её сэкономить.

Алгоритм КМП:

Функция Prefix() получает на вход конкатенацию строк вида S+$+T, где $ выступает в качестве разделителя, и создаёт вектор значений префикс-функции. Из-за наличия разделителя значения префикс-функции не превосходят длины подстроки, а индекс тех значений, которые равны длине подстроки, впоследствии дадут возможность получить искомые значения. Функция KMP() сравнивает значения из вектора, который она получила от функции Prefix(), с длиной подстроки, и в случае, если находит – вычитает из индекса найденного элемента 2 длины подстроки, получает искомые значения (или выводит -1 в случае отсутствия подстроки в строке).

Циклический сдвиг:

Функция Cycle() также получает вектор от функции Prefix(), но для работы функции необходимо только последнее значение из него, для чего используется соответствующая функция. Если это значение равно длине подстроки (условно

строка A), то выводится значение 0 и производится выход из функции. Вычисляется префикс-функция конкатенации в обратном порядке, от неё так же нужно последнее значение. Если сумма двух полученных значений равна длине подстроки – выводится первое, если нет, то -1, ведь B не является циклическим сдвигом A.

Результаты работы программы

Алгоритм КМП:

Входные данные: ab, abab

Выходные данные: 0,2

Циклический сдвиг:

Входные данные: defabc, abcdef

Выходные данные: 3

Выводы

В ходе выполнения данной лабораторной работы был исследован и реализован жадный алгоритм, который оказался довольно прост по принципу и по реализации – на каждом шаге выбирается ребро с наименьшим весом, но цена за простоту – отсутствие гарантий того, что полученный путь будет иметь минимальный вес. Алгоритм A* незначительно сложнее, но за счёт полного перебора можно иметь уверенность в том, что путь будет иметь минимально возможный вес.

Приложение А. Код программы

```
#include <iostream>
#include <vector>
#include <string>
#include <cstdint>

using namespace std;

vector<uint_least32_t> Prefix(string combo){
    uint_least32_t size = combo.length();
    vector<uint_least32_t> result(size);
    for(uint_least32_t i = 1; i < size; i++){
        uint_least32_t j = result[i - 1];
        while(j > 0 && combo[i] != combo[j])
            j = result[j - 1];
        if(combo[i] == combo[j])
            j++;
        result[i] = j;
    }
    return result;
}

void KMP(string substr, string str){
    vector<uint_least32_t> prefix = Prefix(substr + "$" + str);
    uint_least32_t subsize = (uint_least32_t)substr.length();
    uint_least32_t size = (uint_least32_t)str.length() + subsize + 1;
    bool flag = false;
    for(uint_least32_t i = subsize + 1; i < size; i++){
        if(prefix[i] == subsize){
            if(flag)
                cout << ',';
            cout << i - 2 * subsize;
            flag = true;
        }
    }
    if(!flag)
        cout << -1;
}

void Cycle(string substr, string str){
    uint_least32_t index = Prefix(substr + "$" + str).back();
    if(index == (uint_least32_t)substr.length())
        cout << 0;
    else
        if(index + Prefix(str + "$" + substr).back() ==
(uint_least32_t)substr.length())
            cout << index;
        else
            cout << -1;
}

int main()
{
    string substr, str;
    getline(cin, substr);
    getline(cin, str);
    //KMP(substr, str);
    Cycle(substr, str);
    return 0;
}
```