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: Кнут-Моррис-Пратт

Студент гр. 9382	____________________	Герасев Г.А.
Преподаватель	____________________	Фирсов М.А.

Санкт-Петербург
2021

Цель работы.

Изучить алгоритм Кнута-Морриса-Пратта и создать программу, которая находит все вхождения подстроки в тексте. Также реализовать программу, которая определяет, является ли строка циклическим сдвигом другой строки.

Задание 1

Реализуйте алгоритм КМП и с его помощью для заданных шаблона *P* ($|P| \le 15000$) и текста *T* ($|T| \le 5000000$) 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

Sample Input:

ab

abab

Sample Output:

0,2

Задание 2

Заданы две строки *A* ($|A| \le 5000000$) и *B* ($|B| \le 5000000$).

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

Sample Input:

defabc

abcdef

Sample Output:

3

Описание алгоритма

Первоначально определяются значения префикс-функции для шаблона, который необходимо найти в тексте. Значение префикс-функции означает длину наибольшего совпадения префикса и суффикса в подстроке шаблона, которая рассматривается.

Всего значений будет то же, что и длинна шаблона – по одному на каждый символ и, где первое значение – 0, так как размер подстроки равен единице, соответственно максимальный размер префикса и суффикса этой подстроки равен единице.

В первой программе необходимо найти все вхождения шаблона в тексте. Рассматриваются символы текста до тех пор, пока не будет рассмотрен конечный символ. Также рассматриваются символы строки-шаблона. Если символ текста и символ шаблона равен, то рассматриваются следующие символы. Если этот символ был последним символом строки-шаблона, то было найдено вхождение шаблона в тексте, индекс вхождения записывается в результат. Индекс строки-шаблона в этом случае становится значением префикс-функции под предыдущим значением индекса строки-шаблона.

Если рассматриваемые символы не равны, и рассматриваемый символ строки-шаблона был начальным, то сдвигается индекс символа, который рассматривается в тексте на единицу. Если же символ был не начальным, то индекс символа в строке-шаблоне становится равен значению префикс-функции предыдущего индекса.

Во второй программе просто соединяется требуемая строка с самой собой, после чего в ней ищется изначальная строка предыдущим алгоритмом. Если удалось найти вхождение, то строка является циклическим сдвигом.

Оценка сложности по памяти

В обоих программах необходимо хранить две строки – шаблон и текст, а также нужно хранить вектор с значениями префикс-функции. Сложность составляет O(N + M), где N – длина первой строки, M – длина второй строки.

Оценка сложности по времени

Значение префикс-функции вычисляется за O(M) сравнений, где M – длина строки-шаблона, так как необходимо пройтись по всей строке, чтобы определить префикс-функцию.

Поиск строки-шаблона в тексте с помощью алгоритма КМП будет занимать O(N), где N – длина текста, так как весь текст будет пройден ровно один раз. Каждый символ текста будет рассмотрен ровно один раз за счет использования префикс-функции.

Итоговая оценка – O(M + N).

Оценка для второй программы будет такой же, ведь она использует за основу первую.

Тестирование

Результаты тестирования программы можно посмотреть в приложениях Б и В.

Выводы.

Был изучен алгоритм Кнута-Морриса-Пратта и создана программа, которая находит все вхождения подстроки в тексте. Также была реализована программа, которая определяет, является ли строка циклическим сдвигом другой строки.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.cpp

```
#include "string"
#include "vector"
#include "iostream"

#define FIRSTMODE false
#define COLOR BLUE



#define RESET   "\033[0m"
#define BLACK   "\033[30m"      /* Black */
#define RED     "\033[31m"      /* Red */
#define GREEN   "\033[32m"      /* Green */
#define YELLOW  "\033[33m"      /* Yellow */
#define BLUE    "\033[34m"      /* Blue */

void highlightedLetterPrint(int highlighted, std::string output)
    {
        for (int i=0; i<output.length(); i++)
        {
            if (i == highlighted) {
                std::cout << COLOR << output[i] << RESET;
            } else {
                std::cout << output[i];
            }
        }
    }



void printStringsHandelingState(std::string & first, std::string & second, int firstIndex, int secondIndex)
{
    highlightedLetterPrint(firstIndex, first);
    std::cout << " | ";
    highlightedLetterPrint(secondIndex, second);
    std::cout << '\n';
}


std::vector<int> getPrefixFunction(std::string & pattern) {
    auto prefixFunction = std::vector<int>(pattern.length());

    int i = 1;
    int j = 0; // Indexes


    std::cout << "Creating prefix function" << std::endl;

    while (i != pattern.length()) {

        std::cout << "i = " << std::to_string(i) << "; j = " <<
std::to_string(j) << "\n";

        if (pattern[i] == pattern[j]) {
            // If symbols are equal -- prefix function of i is
function[j+1]
            printStringsHandelingState(pattern, pattern, j, i);
            std::cout << "Symbols are equal, so f[" << i << "] = " <<
j + 1 << ", i and j are insreased by 1\n\n";
            prefixFunction[i] = j + 1;
            i++;std::cin >> pattern;
            // In this case f[i] = 0
            printStringsHandelingState(pattern, pattern, j, i);
            std::cout << "Symbols are not equal and j == 0 => f["
                << i << "] = 0" << ", i insreased by 1\n\n";

            prefixFunction[i] = 0;
            i++;
        } else {
            // In this case we change j on prefixFunction[j - 1]
            printStringsHandelingState(pattern, pattern, j, i);
            std::cout << "Symbols are not equal and j != 0 => j = " <<
prefixFunction[j - 1] << "\n\n";
            j = prefixFunction[j - 1];
        }
    }
    return prefixFunction;
}


std::vector<int> getSubstringIndexes(std::string & text, std::string &
pattern) {
    auto prefixFunction = getPrefixFunction(pattern);

    int textIndex = 0;
    int substringIndex = 0;
    auto result = std::vector<int>();

    std::cout << "[Looking for pattern in the text]" << '\n';

    // While there is any text left
    while (textIndex != text.length()) {

        std::cout << "textIndex = " << textIndex << "; substringIndex
= " << substringIndex << "\n";

        if (text[textIndex] == pattern[substringIndex]) {
            auto info = "";

            textIndex += 1;
            substringIndex += 1;

            if (substringIndex == pattern.length()) {
                // если индекс конечный для подстроки, то мы нашли
заданную подстроку в тексте
                // сдвигаемся назад в подстроке
                std::cout << "Symbols are equal and there is no
symbols in the substring left\n";
                printStringsHandelingState(text, pattern, textIndex -
1, substringIndex - 1);
                result.push_back(textIndex - substringIndex);
                substringIndex = prefixFunction[substringIndex - 1];
            } else {
                std::cout << "Symbols are equal and there are symbols
in the substring left, handeling them...\n";
                printStringsHandelingState(text, pattern, textIndex -
1, substringIndex - 1);
            }

        } else { // If symbols are not equal
            if (substringIndex == 0) {
                std::cout <<"Symbols are not equal and this is a first
letter of substring\n";
                printStringsHandelingState(text, pattern, textIndex,
substringIndex);
                textIndex += 1;
            }
            // If it's not a first symbol -- changing substringIndex
to prefixFunction[substringIndex - 1]
            else {
                std::cout << "Symbols are not equal and this is not a
first letter of substring => it's now equal
prefixFunction[substringIndex - 1]\n";
                printStringsHandelingState(text, pattern, textIndex,
substringIndex);
                substringIndex = prefixFunction[substringIndex - 1];
            }
        }
    }

    std::cout << "There is no symbols left in the text\n";

    return result;
```

```
}


void firstTask(std::string & text, std::string & pattern) {
    auto result = getSubstringIndexes(text, pattern);

    if (!result.empty()) {
        for (int i = 0; i < result.size() - 1; ++i) std::cout <<
result[i] << ',';
        std::cout << result[result.size() - 1];
    } else {
        std::cout << -1;
    }
}


void secondTask(std::string & text, std::string & pattern) {

    if (text.length() != pattern.length()) {
        std::cout << -1;
        return;
    }

    text = text + text;
    auto result = getSubstringIndexes(text, pattern);
    if (!result.empty()) {
        std::cout << result[0];
    } else {
        std::cout << -1;
    }
}


int main() {
    std::string text;
    std::string pattern;
    std::cin >> pattern;
    std::cin >> text;

    if (FIRSTMODE) {
        firstTask(text, pattern);
    } else {
        secondTask(text, pattern);
    }
    std::cout << '\n';
    return 0;
}
```

ПРИЛОЖЕНИЕ Б

ТЕСТИРОВАНИЕ ПРОГРАММЫ 1

Входные данные	Выходные данные
ababab ab	Важно – в выводе программы используется подсветка цветом рассматриваемых символов, которая в отчете не сохранилась. Creating prefix function i = 1; j = 0 ab | ab Symbols are not equal and j == 0 => f[1] = 0, i insreased by 1 [Looking for pattern in the text] textIndex = 0; substringIndex = 0 Symbols are equal and there are symbols in the substring left, handeling them... ababab | ab textIndex = 1; substringIndex = 1 Symbols are equal and there is no symbols in the substring left ababab | ab textIndex = 2; substringIndex = 0 Symbols are equal and there are symbols in the substring left, handeling them... ababab | ab textIndex = 3; substringIndex = 1 Symbols are equal and there is no symbols in the substring left ababab | ab textIndex = 4; substringIndex = 0 Symbols are equal and there are symbols in the substring left, handeling them... ababab | ab textIndex = 5; substringIndex = 1 Symbols are equal and there is no symbols in the substring left ababab | ab

	There is no symbols left in the text 0,2,4
oabcdfe abcdfeo	Creating prefix function i = 1; j = 0 abcdfeo | abcdfeo Symbols are not equal and j == 0 => f[1] = 0, i insreased by 1 i = 2; j = 0 abcdfeo | abcdfeo Symbols are not equal and j == 0 => f[2] = 0, i insreased by 1 i = 3; j = 0 abcdfeo | abcdfeo Symbols are not equal and j == 0 => f[3] = 0, i insreased by 1 i = 4; j = 0 abcdfeo | abcdfeo Symbols are not equal and j == 0 => f[4] = 0, i insreased by 1 i = 5; j = 0 abcdfeo | abcdfeo Symbols are not equal and j == 0 => f[5] = 0, i insreased by 1 i = 6; j = 0 abcdfeo | abcdfeo Symbols are not equal and j == 0 => f[6] = 0, i insreased by 1 [Looking for pattern in the text] textIndex = 0; substringIndex = 0 Symbols are not equal and this is a first letter of substring oabcdfe | abcdfeo textIndex = 1; substringIndex = 0 Symbols are equal and there are symbols in the substring left,

	handeling them... oabcdfe | abcdfeo textIndex = 2; substringIndex = 1 Symbols are equal and there are symbols in the substring left, handeling them... oabcdfe | abcdfeo textIndex = 3; substringIndex = 2 Symbols are equal and there are symbols in the substring left, handeling them... oabcdfe | abcdfeo textIndex = 4; substringIndex = 3 Symbols are equal and there are symbols in the substring left, handeling them... oabcdfe | abcdfeo textIndex = 5; substringIndex = 4 Symbols are equal and there are symbols in the substring left, handeling them... oabcdfe | abcdfeo textIndex = 6; substringIndex = 5 Symbols are equal and there are symbols in the substring left, handeling them... oabcdfe | abcdfeo There is no symbols left in the text -1
qwerty qwe	Creating prefix function i = 1; j = 0 qwe | qwe Symbols are not equal and j == 0 => f[1] = 0, i insreased by 1 i = 2; j = 0 qwe | qwe Symbols are not equal and j == 0 => f[2] = 0, i insreased by 1

	[Looking for pattern in the text] textIndex = 0; substringIndex = 0 Symbols are equal and there are symbols in the substring left, handeling them... qwerty | qwe textIndex = 1; substringIndex = 1 Symbols are equal and there are symbols in the substring left, handeling them... qwerty | qwe textIndex = 2; substringIndex = 2 Symbols are equal and there is no symbols in the substring left qwerty | qwe textIndex = 3; substringIndex = 0 Symbols are not equal and this is a first letter of substring qwerty | qwe textIndex = 4; substringIndex = 0 Symbols are not equal and this is a first letter of substring qwerty | qwe textIndex = 5; substringIndex = 0 Symbols are not equal and this is a first letter of substring qwerty | qwe There is no symbols left in the text 0
qwe qwert	Creating prefix function i = 1; j = 0 qwert | qwert Symbols are not equal and j == 0 => f[1] = 0, i insreased by 1 i = 2; j = 0 qwert | qwert Symbols are not equal and j == 0 => f[2] = 0, i insreased by 1 i = 3; j = 0

	qwert | qwert Symbols are not equal and j == 0 => f[3] = 0, i insreased by 1 i = 4; j = 0 qwert | qwert Symbols are not equal and j == 0 => f[4] = 0, i insreased by 1 [Looking for pattern in the text] textIndex = 0; substringIndex = 0 Symbols are equal and there are symbols in the substring left, handeling them... qwe | qwert textIndex = 1; substringIndex = 1 Symbols are equal and there are symbols in the substring left, handeling them... qwe | qwert textIndex = 2; substringIndex = 2 Symbols are equal and there are symbols in the substring left, handeling them... qwe | qwert There is no symbols left in the text -1
sdnvksefljjk pnkvsnvsklv	Creating prefix function i = 1; j = 0 pnkvsnvsklv | pnkvsnvsklv Symbols are not equal and j == 0 => f[1] = 0, i insreased by 1 i = 2; j = 0 pnkvsnvsklv | pnkvsnvsklv Symbols are not equal and j == 0 => f[2] = 0, i insreased by 1 i = 3; j = 0 pnkvsnvsklv | pnkvsnvsklv

	Symbols are not equal and j == 0 => f[3] = 0, i insreased by 1 i = 4; j = 0 pnkvsnvsklv | pnkvsnvsklv Symbols are not equal and j == 0 => f[4] = 0, i insreased by 1 i = 5; j = 0 pnkvsnvsklv | pnkvsnvsklv Symbols are not equal and j == 0 => f[5] = 0, i insreased by 1 i = 6; j = 0 pnkvsnvsklv | pnkvsnvsklv Symbols are not equal and j == 0 => f[6] = 0, i insreased by 1 i = 7; j = 0 pnkvsnvsklv | pnkvsnvsklv Symbols are not equal and j == 0 => f[7] = 0, i insreased by 1 i = 8; j = 0 pnkvsnvsklv | pnkvsnvsklv Symbols are not equal and j == 0 => f[8] = 0, i insreased by 1 i = 9; j = 0 pnkvsnvsklv | pnkvsnvsklv Symbols are not equal and j == 0 => f[9] = 0, i insreased by 1 i = 10; j = 0 pnkvsnvsklv | pnkvsnvsklv Symbols are not equal and j == 0 => f[10] = 0, i insreased by 1 [Looking for pattern in the text] textIndex = 0; substringIndex = 0 Symbols are not equal and this is a first letter of substring

<table>
<tr><td></td><td>sdnvksefljjk | pnkvsnvsklv
textIndex = 1; substringIndex = 0
Symbols are not equal and this is a first letter of substring
sdnvksefljjk | pnkvsnvsklv
textIndex = 2; substringIndex = 0
Symbols are not equal and this is a first letter of substring
sdnvksefljjk | pnkvsnvsklv
textIndex = 3; substringIndex = 0
Symbols are not equal and this is a first letter of substring
sdnvksefljjk | pnkvsnvsklv
textIndex = 4; substringIndex = 0
Symbols are not equal and this is a first letter of substring
sdnvksefljjk | pnkvsnvsklv
textIndex = 5; substringIndex = 0
Symbols are not equal and this is a first letter of substring
sdnvksefljjk | pnkvsnvsklv
textIndex = 6; substringIndex = 0
Symbols are not equal and this is a first letter of substring
sdnvksefljjk | pnkvsnvsklv
textIndex = 7; substringIndex = 0
Symbols are not equal and this is a first letter of substring
sdnvksefljjk | pnkvsnvsklv
textIndex = 8; substringIndex = 0
Symbols are not equal and this is a first letter of substring
sdnvksefljjk | pnkvsnvsklv
textIndex = 9; substringIndex = 0
Symbols are not equal and this is a first letter of substring
sdnvksefljjk | pnkvsnvsklv
textIndex = 10; substringIndex = 0
Symbols are not equal and this is a first letter of substring
sdnvksefljjk | pnkvsnvsklv
textIndex = 11; substringIndex = 0
Symbols are not equal and this is a first letter of substring</td></tr>
</table>

	sdnvksefljjk | pnkvsnvsklv There is no symbols left in the text -1

ПРИЛОЖЕНИЕ В

ТЕСТИРОВАНИЕ ПРОГРАММЫ 2

Входные данные	Выходные данные
abcdef defabc	Creating prefix function i = 1; j = 0 defabc | defabc Symbols are not equal and j == 0 => f[1] = 0, i increased by 1 i = 2; j = 0 defabc | defabc Symbols are not equal and j == 0 => f[2] = 0, i increased by 1 i = 3; j = 0 defabc | defabc Symbols are not equal and j == 0 => f[3] = 0, i increased by 1 i = 4; j = 0 defabc | defabc Symbols are not equal and j == 0 => f[4] = 0, i increased by 1 i = 5; j = 0 defabc | defabc Symbols are not equal and j == 0 => f[5] = 0, i increased by 1 [Looking for pattern in the text] textIndex = 0; substringIndex = 0 Symbols are not equal and this is a first letter of substring abcdefabcdef | defabc textIndex = 1; substringIndex = 0 Symbols are not equal and this is a first letter of substring

	abcdefabcdef | defabc textIndex = 2; substringIndex = 0 Symbols are not equal and this is a first letter of substring abcdefabcdef | defabc textIndex = 3; substringIndex = 0 Symbols are equal and there are symbols in the substring left, handeling them... abcdefabcdef | defabc textIndex = 4; substringIndex = 1 Symbols are equal and there are symbols in the substring left, handeling them... abcdefabcdef | defabc textIndex = 5; substringIndex = 2 Symbols are equal and there are symbols in the substring left, handeling them... abcdefabcdef | defabc textIndex = 6; substringIndex = 3 Symbols are equal and there are symbols in the substring left, handeling them... abcdefabcdef | defabc textIndex = 7; substringIndex = 4 Symbols are equal and there are symbols in the substring left, handeling them... abcdefabcdef | defabc textIndex = 8; substringIndex = 5 Symbols are equal and there is no symbols in the substring left abcdefabcdef | defabc textIndex = 9; substringIndex = 0 Symbols are equal and there are symbols in the substring left, handeling them... abcdefabcdef | defabcCreating prefix function i = 1; j = 0 defabc | defabc

	Symbols are not equal and j == 0 => f[1] = 0, i increased by 1 i = 2; j = 0 defabc | defabc Symbols are not equal and j == 0 => f[2] = 0, i increased by 1 i = 3; j = 0 defabc | defabc Symbols are not equal and j == 0 => f[3] = 0, i increased by 1 i = 4; j = 0 defabc | defabc Symbols are not equal and j == 0 => f[4] = 0, i increased by 1 i = 5; j = 0 defabc | defabc Symbols are not equal and j == 0 => f[5] = 0, i increased by 1 [Looking for pattern in the text] textIndex = 0; substringIndex = 0 Symbols are not equal and this is a first letter of substring abcdefabcdef | defabc textIndex = 1; substringIndex = 0 Symbols are not equal and this is a first letter of substring abcdefabcdef | defabc textIndex = 2; substringIndex = 0 Symbols are not equal and this is a first letter of substring abcdefabcdef | defabc textIndex = 3; substringIndex = 0 Symbols are equal and there are symbols in the substring left, handeling them... abcdefabcdef | defabc textIndex = 4; substringIndex = 1

	Symbols are equal and there are symbols in the substring left, handeling them... abcdefabcdef | defabc textIndex = 5; substringIndex = 2 Symbols are equal and there are symbols in the substring left, handeling them... abcdefabcdef | defabc textIndex = 6; substringIndex = 3 Symbols are equal and there are symbols in the substring left, handeling them... abcdefabcdef | defabc textIndex = 7; substringIndex = 4 Symbols are equal and there are symbols in the substring left, handeling them... abcdefabcdef | defabc textIndex = 8; substringIndex = 5 Symbols are equal and there is no symbols in the substring left abcdefabcdef | defabc textIndex = 9; substringIndex = 0 Symbols are equal and there are symbols in the substring left, handeling them... abcdefabcdef | defabc textIndex = 10; substringIndex = 1 Symbols are equal and there are symbols in the substring left, handeling them... abcdefabcdef | defabc textIndex = 11; substringIndex = 2 Symbols are equal and there are symbols in the substring left, handeling them... abcdefabcdef | defabc There is no symbols left in the text 3 textIndex = 10; substringIndex = 1

	Symbols are equal and there are symbols in the substring left, handeling them... abcdefabcdef | defabc textIndex = 11; substringIndex = 2 Symbols are equal and there are symbols in the substring left, Creating prefix function i = 1; j = 0 defabc | defabc Symbols are not equal and j == 0 => f[1] = 0, i increased by 1 i = 2; j = 0 defabc | defabc Symbols are not equal and j == 0 => f[2] = 0, i increased by 1 i = 3; j = 0 defabc | defabc Symbols are not equal and j == 0 => f[3] = 0, i increased by 1 i = 4; j = 0 defabc | defabc Symbols are not equal and j == 0 => f[4] = 0, i increased by 1 i = 5; j = 0 defabc | defabc Symbols are not equal and j == 0 => f[5] = 0, i increased by 1 [Looking for pattern in the text] textIndex = 0; substringIndex = 0 Symbols are not equal and this is a first letter of substring abcdefabcdef | defabc textIndex = 1; substringIndex = 0 Symbols are not equal and this is a first letter of substring abcdefabcdef | defabc

	textIndex = 2; substringIndex = 0 Symbols are not equal and this is a first letter of substring abcdefabcdef | defabc textIndex = 3; substringIndex = 0 Symbols are equal and there are symbols in the substring left, handeling them... abcdefabcdef | defabc textIndex = 4; substringIndex = 1 Symbols are equal and there are symbols in the substring left, handeling them... abcdefabcdef | defabc textIndex = 5; substringIndex = 2 Symbols are equal and there are symbols in the substring left, handeling them... abcdefabcdef | defabc textIndex = 6; substringIndex = 3 Symbols are equal and there are symbols in the substring left, handeling them... abcdefabcdef | defabc textIndex = 7; substringIndex = 4 Symbols are equal and there are symbols in the substring left, handeling them... abcdefabcdef | defabc textIndex = 8; substringIndex = 5 Symbols are equal and there is no symbols in the substring left abcdefabcdef | defabc textIndex = 9; substringIndex = 0 Symbols are equal and there are symbols in the substring left, handeling them... abcdefabcdef | defabc textIndex = 10; substringIndex = 1 Symbols are equal and there are symbols in the substring left, handeling them...

	abcdefabcdef | defabc textIndex = 11; substringIndex = 2 Symbols are equal and there are symbols in the substring left, handeling them... abcdefabcdef | defabc There is no symbols left in the text 3handeling them... abcdefabcdef | defabc There is no symbols left in the text 3
defabc abc	-1
aaaaaaaa bbbbbbbb	-1
abcabcabc bcabcabca	1
abcdfeo oabcdfe	6