Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации

Ордена Трудового Красного Знамени федеральное государственное

бюджетное учреждение высшего образования

«Московский технический университет связи и информатики»

Кафедра Математической кибернетики и

информационных технологий

Лабораторная работа №3

на тему: «Методы поиска подстроки в строке»

Выполнила:

Студентка группы БФИ1902

Кареева В.В.

Вариант 5

Проверил:

Москва, 2021 г.

Оглавление

[1. Цель работы 2](#_Toc72409356)

[2. Задание на лабораторную работу 2](#_Toc72409357)

[3. Листинг программы 2](#_Toc72409358)

[4. Результат работы программы 7](#_Toc72409359)

[Список использованных источников 8](#_Toc72409360)

# **1. Цель работы**

Цель работы: рассмотреть и изучить основные методы поиска подстроки в строке.

# **2. Задание на лабораторную работу**

**Задание 1**

Реализовать методы поиска подстроки в строке. Добавить возможность ввода строки и подстроки с клавиатуры. Предусмотреть возможность существования пробела. Реализовать возможность выбора опции чувствительности или нечувствительности к регистру. Оценить время работы каждого алгоритма поиска и сравнить его со временем работы стандартной функции поиска, используемой в выбранном языке программирования. Алгоритмы: 1.Кнута-Морриса-Пратта 2.Упрощенный Бойера-Мура

**Задание 2 «Пятнашки»**

# **3. Листинг программы**

package com.company;  
import java.util.\*;  
import java.lang.String;  
  
public class Main {  
  
 public static void main(String[] args) {  
 Scanner scanstr = new Scanner(System.in);  
 System.out.println("Введите строку: ");  
 String str = scanstr.nextLine();  
 System.out.println("Введите подстроку: ");  
 String podstr = scanstr.nextLine();  
  
 int res = B\_Mura(str, podstr);  
 int[] res\_1 = pfl(str, podstr);  
 System.out.println("Строка входит с символа: " + res);  
 System.out.println("Массив вхождений: " + Arrays.toString(res\_1));  
  
 int[][] blocks = new int[][]{{1, 2, 3, 0}, {5, 6, 7, 8}, {9, 10, 11, 12}, {13, 14, 15, 4}};  
 Board initial = new Board(blocks);  
 Solver solver = new Solver(initial);  
 System.out.println("Minimum number of moves = " + solver.moves());  
 for (Board board : solver.solution())  
 System.out.println(board);  
  
 }  
  
 public static int B\_Mura(String str, String podstr){  
 int strLen = str.length();  
 int podstrLen = podstr.length();  
 if (podstrLen > strLen) {  
 return -1;  
 }  
 HashMap<Character, Integer> offsetTable = new HashMap<Character, Integer>();  
 for (int i = 0; i <= 255; i++) {  
 offsetTable.put((char) i, podstrLen);  
 }  
 for (int i = 0; i < podstrLen - 1; i++) {  
 offsetTable.put(podstr.charAt(i), podstrLen - i - 1);  
 }  
 int i = podstrLen - 1;  
 int j = i;  
 int k = i;  
 while (j >= 0 && i <= strLen - 1) {  
 j = podstrLen - 1;  
 k = i;  
 while (j >= 0 && str.charAt(k) == podstr.charAt(j)) {  
 k--;  
 j--;  
 }  
 i += offsetTable.get(str.charAt(i));  
 }  
 if (k >= strLen - podstrLen) {  
 return -1;  
 } else {  
 return k + 1;  
 }  
 }  
*/\*\*  
 public static int[] kmp(String str, String podstr) {  
 int pl = str.length();  
 int tl = podstr.length();  
 int[] answer = new int[];  
 int[] p = prefixFunction(str + "#" + podstr);  
 int count = 0;  
 for (int i = 0; i <= tl - 1; i++) {  
 if (p[pl + i + 1] == pl)  
 answer[count++] = i;  
 }  
 return answer;  
 }  
  
 public static int[] prefixFunction(String s) {  
 int[] p = new int[];  
 int k;  
 p[0] = 0;  
 for (int i = 1; i <= s.length() - 1; i++) {  
 k = p[i - 1];  
 while (k > 0 && s.charAt(i) != s.charAt(k))  
 k = p[k - 1];  
 if (s.charAt(i) == s.charAt(k)  
 k++;  
 p[i] = k;  
 }  
 return p;  
 }  
\*\*/* public static int[] pfl(String str, String podstr)  
 {  
 String t = podstr + "#" + str;  
 char[] text = t.toCharArray();  
 int[] pfl = new int[text.length];  
 pfl[0] = 0;  
  
 for (int i = 1; i < text.length; ++i)  
 {  
 int k = pfl[i - 1];  
 while (text[k] != text[i] && k > 0)  
 {  
 k = pfl[k - 1];  
 }  
 if (text[k] == text[i])  
 {  
 pfl[i] = k + 1;  
 }  
 else  
 {  
 pfl[i] = 0;  
 }  
 }  
  
 System.out.println("Строка входит с символов:");  
 for (int i = podstr.length() + 1 ; i <= pfl.length - 1; i++){  
 if (pfl[i] == podstr.length()) {  
 int vh = i - 2\*podstr.length() ;  
 System.out.print(vh + " ");  
 }  
 }  
 System.out.println();  
  
 return pfl;  
 }  
  
}

# **4. Вывод:** мы рассмотрели и изучили основные методы поиска подстроки в строке.

# **Список использованных источников**

1) ГОСТ 7.32-2017 Система стандартов по информации, библиотечному и издательскому делу. Отчёт о научно-исследовательской работе. Структура и правила оформления

2) ГОСТ 7.1-2003 Библиографическая запись. Библиографическое описание. Общие требования и правила составления