**Федеральное агентство связи**

**Ордена Трудового Красного Знамени**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский технический университет связи и информатики»**

Кафедра Математической кибернетики и информационных технологий

**Отчет по лабораторной работе**

по предмету «СИАОД»

на тему:

«Методы поиска»

Выполнил: студент группы

Митрохин Ярослав Игоревич

Руководитель:

Кутейников Иван Александрович

Москва 2020

*Цель работы:* реализовать заданный метод поиска подстроки в строке в соответствии с индивидуальным заданием. Оценить время работы каждого алгоритма поиска и сравнить его со временем работы стандартной функции поиска, используемой в выбранном языке программирования.

*Ход работы:*

*Код программы:*

import java.io.BufferedReader;  
import java.io.IOException;  
import java.io.InputStreamReader;  
import java.util.HashMap;  
public class lab3 {  
 public static void main(String[] args) throws IOException {  
 //Hellow my name is Kyr who are you?  
 BufferedReader reader = new BufferedReader(new InputStreamReader(System.*in*));  
 System.*out*.println("Enter input text: ");  
 String src = reader.readLine();  
 src = src.toLowerCase();  
 System.*out*.println("Enter word: ");  
 String tmp = reader.readLine();  
 tmp = tmp.toLowerCase();  
 double time = System.*nanoTime*();  
 *myrBol*(src,tmp);  
 System.*out*.println("Time: " + (System.*nanoTime*()-time)/1000000);  
 time = 0;  
 System.*out*.println("Boyer Mura: " + *myrBol*(src,tmp));  
 System.*out*.println("\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*-\*");  
 double time1 = System.*nanoTime*();  
 src.indexOf(tmp);  
 System.*out*.println("Time: " + (System.*nanoTime*()-time1)/1000000);  
 time1 = 0;  
 System.*out*.println("Standart: " + src.indexOf(tmp));}  
 public static int myrBol(String source,String foundWord)  
 {  
 int sourceLen = source.length();  
 int tempLen = foundWord.length();  
 if (tempLen > sourceLen) return -1;  
 HashMap<Character,Integer> setTable = new HashMap<Character, Integer>();  
 for (int i = 0; i <= 255;i++)  
 setTable.put((char) i, tempLen);  
 for (int i = 0; i < tempLen - 1;i++)  
 setTable.put(foundWord.charAt(i),tempLen - i - 1);//таблица смещений для каждого символа  
 int i = tempLen - 1;  
 int j = i;  
 int k = i;  
 while (j >= 0 && i <= sourceLen - 1) {  
 j = tempLen - 1;  
 k = i;  
 while (j >= 0 && source.charAt(k) == foundWord.charAt(j)) {  
 k--;  
 j--;}  
 i += setTable.get(source.charAt(i));}  
 if (k >= sourceLen - tempLen)  
 return -1;  
 else  
 return k+1;}}

*Результат программы:*

*Индивидуальный метод поиска:* **упрощенный Бойера-Мура\**

*Стандартный метод поиска:* **indexof()**

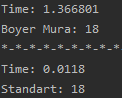
Входная строка:



Подстрока:



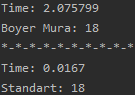
Результаты программы:



Подстрока(Независим от регистра):



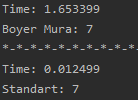
Результаты программы:



Подстрока:



Результат программы:



*Выводы:* реализовал заданный метод поиска подстроки в строке в соответствии с индивидуальным заданием. Оценил время работы каждого алгоритма поиска и сравнил его со временем работы стандартной функции поиска, используемой в выбранном языке программирования. С учетом результатов программы можно сделать вывод, что метод поиска Бойера-Мура(1.653) работает намного медленнее, чем стандартный метод поиска(0.012). Программа реализована таким образом, что подстрока не чувствительна к регистру.