**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

Клюшкин Дмитрий Алексеевич

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:*

public class lab3 {  
 static long *timer*=0;  
  
 public static void main(String[] args) {  
 String str = "mas Dima pole cudes";  
 String sub = "dima";  
 Boolean registr = true;  
  
 *timeSet*();  
 int search1=*KMP*(str, sub, registr);  
 *timeSet*();*timeSet*();  
 int search2=str.indexOf(sub);  
 *timeSet*();  
 }  
  
 public static int KMP(String string, String subString, Boolean Registr){  
  
 if (!Registr){  
 string=string.toLowerCase();  
 subString=subString.toLowerCase();  
 }  
  
 int[] pref=*PrF*(subString);  
 int index=0;  
 int indexPref=0;  
 char[] str=string.toCharArray();  
 char[] sub=subString.toCharArray();  
  
  
 while(index<(string.length()-pref.length) && indexPref<sub.length){  
 if (str[index]==sub[indexPref]){  
 index++;  
 indexPref++;  
 }  
 else {  
 if (indexPref == 0)  
 index++;  
 else  
 indexPref = pref[indexPref - 1];  
 }  
 }  
  
 if (indexPref>=sub.length) return index-indexPref;  
 return -1;  
 }  
  
 public static int[] PrF(String s){  
 int size = s.length();  
 int[] mas = new int[size];  
 int sum=0;  
  
 for(int i=2; i<=size;i++, sum=0){  
 for (int j=1;j<i;j++){  
 String str1=s.substring(0,j);  
 String str2=s.substring(i-j,i);  
 if (s.substring(0,j).equals(s.substring(i-j,i)))  
 sum+=1;  
 }  
 mas[i-1]=sum;  
 }  
  
 return mas;  
 }  
  
 public static void timeSet(){  
 if (*timer*==0) *timer*=System.*nanoTime*();  
 else {  
 double val = (double)(System.*nanoTime*()-*timer*);  
 System.*out*.println(val/1000000);  
 *timer*=0;  
 }  
 }  
}

*Результат программы:*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Символов | | 15 | 50 | 100 | 15 | 50 | 100 |
| Длина искомой строки | | 3 | 3 | 3 | 10 | 10 | 10 |
| время | Кнута-Морриса-Пратта | 0.042 | 0.049 | 0.061 | 0.077 | 0.086 | 0.122 |
| IndexOf | 0.011 | 0.006 | 0.016 | 0.006 | 0.009 | 0.01 |

Вывод: был реализован метод поиска **Кнута-Морриса-Пратта.** При сравнении со стандартным поиском **IndeOf**, можно сделать вывод, что стандартный поиск подстроки более оптимизирован.