**Федеральное агентство связи**

**Ордена Трудового Красного Знамени**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

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

Кафедра Информатики

****

**Отчет по лабораторной работе № 3**

по дисциплине «СиАОД»

на тему:

«**Методы поиска подстроки в строке**»

Выполнила: студентка группы БВТ1802

Лаврухина Елена Павловна

Руководитель:

Кутейников Иван Алексеевич

Москва 2020

# **Цель работы**

Реализовать заданный метод поиска подстроки в строке в соответствии с индивидуальным заданием. Для всех вариантов добавить реализацию добавления строк, ввода подстроки и поиска подстроки. Предусмотреть возможность существования пробела. Ввести опцию чувствительности/нечувствительности к регистру. Оценить время работы каждого алгоритма поиска и сравнить его со временем работы стандартной функции поиска, используемой в выбранном языке программирования.

Выполнение

Код программы

**import** java.io.BufferedReader;  
**import** java.io.IOException;  
**import** java.io.InputStreamReader;  
**import** java.util.HashMap;

**public class** lab3 {  
 **public static double** *timer*=0.0;  
 **public static final** String ***ANSI\_RED*** = **"\u001B[31m"**;  
 **public static final** String ***ANSI\_RESET*** = **"\u001B[0m"**;  
 **public static final** String ***BLACK\_BOLD*** = **"\033[1;30m"**;  
 **public static void** main(String[] args) **throws** IOException {  
 *// ввод строки* System.***out***.print(**"Введите строку: "**);  
 BufferedReader read = **new** BufferedReader(**new** InputStreamReader(System.***in***));  
 **char**[] n = read.readLine().toCharArray();  
 String[] line = **new** String[n.**length**];  
 System.***out***.print(**"Количество элементов: "**);  
 System.***out***.println(n.**length**);  
 **for** (**int** i = 0; i <line.**length**; i++) {  
 line[i] = String.*valueOf*(n[i]);  
 }  
 String str = **""**;  
 **for** (**int** i=0; i<line.**length**; i++) {  
 str+=line[i];  
 }  
 *// создание шаблона* System.***out***.print(**"Введите подстроку: "**);  
 BufferedReader reader = **new** BufferedReader(**new** InputStreamReader(System.***in***));  
 **char**[] s = reader.readLine().toCharArray();  
 String[] forFindLine = **new** String[s.**length**];  
 **for** (**int** i = 0; i < s.**length**; i++) {  
 forFindLine[i] = String.*valueOf*(s[i]);  
 }  
 String podStr = **""**;  
 **for** (**int** i=0; i<forFindLine.**length**; i++) {  
 podStr+=forFindLine[i];  
 }  
 *// опция чувствительности к регистру* **boolean** sense;  
 System.***out***.println(**"Включить чувствительность к регистру? Д/Н"**);  
 **if** (reader.readLine().equals(**"Д"**)) sense = **true**;  
 **else** sense = **false**;  
 String[] newline=**new** String[line.**length**];  
 String[] word = **new** String[forFindLine.**length**];  
 **if** (!sense) {  
 newline = *ToDown*(line);  
 *printStrln*(newline);  
 word = *ToDown*(forFindLine);  
 *printStrln*(word);  
 }  
 **else** {  
 newline=*Copy*(line);  
 word=*Copy*(forFindLine);  
 }  
 *// создание мапа для шаблона* HashMap<String, Integer> map = *StrsIndexes*(word);  
 System.***out***.println(**"\033[1;30mПоиск методом Бойера-Мура\u001B[0m"**);  
 System.***out***.println(map);  
 *// поиск подстроки  
 time*();  
 **int** index = *Finding*(newline, word, map);  
 System.***out***.print(**"Время: "**);  
 *time*();  
 **if** (index!=-1) {  
 line=*RedColor*(line, index, forFindLine.**length**);  
 *printStrln*(line);  
 System.***out***.print(**"Индекс подстроки: "**);  
 System.***out***.println(index);}  
 *//стандартный поиск* System.***out***.println(**"\033[1;30mСтандартный поиск\u001B[0m"**);  
 System.***out***.print(**"Время: "**);  
 *time*();  
 index = str.indexOf(podStr);  
 *time*();  
 **if** (index!=-1) {  
 line=*RedColor*(line, index, forFindLine.**length**);  
 *printStrln*(line);  
 System.***out***.print(**"Индекс подстроки: "**);  
 System.***out***.println(index);}  
 **else** System.***out***.println(**"Подстрока не найдена"**);  
 }  
 **public static void** printStrln(String[] line) {  
 **for** (**int** i = 0; i < line.**length**; i++) {  
 System.***out***.print(line[i]);  
 }  
 System.***out***.println();  
 }  
 **public static** HashMap<String, Integer> StrsIndexes(String[] line) {  
 HashMap<String, Integer> map = **new** HashMap<>();  
 **for** (**int** i = 0; i < line.**length**; i++) {  
 **if** (!map.containsKey(line[i])) {  
 **if** (i != line.**length** - 1)  
 map.put(line[i], line.**length** - i - 1);  
 **else** map.put(line[i], line.**length**);  
 }

**else if** (i != line.**length** - 1) {  
 map.put(line[i], line.**length** - i - 1);  
 }  
 }  
 **return** map;  
 }  
 **public static int** Finding(String[] line, String[] forfindline, HashMap<String, Integer> map) {  
 **for** (**int** i = forfindline.**length**-1; i < line.**length**; i++) {  
 **for** (**int** k = 0; k <= forfindline.**length**-1; k++) {  
 **if** (!forfindline[forfindline.**length**-k-1].equals(line[i - k])) {  
 **if** (k!=0) i+=map.get(forfindline[forfindline.**length**-1])-1;  
 **else if** (map.containsKey(line[i-k])) i+=map.get(line[i-k])-1;  
 **else** i+=map.get(forfindline[forfindline.**length**-1])-1;  
 **break**;  
 }  
 **if** (k == forfindline.**length**-1 && forfindline[forfindline.**length**-k-1].equals(line[i - k])) {  
 System.***out***.println(**"Успешно"**);  
 **return** i-forfindline.**length**+1;  
 }  
 }  
 }  
 System.***out***.println(**"Подстрока не найдена"**);  
 **return** -1;  
 }  
 **public static** String[] Copy(String[] line) {  
 String[] strings = **new** String[line.**length**];  
 **for** (**int** i=0; i<line.**length**; i++) {  
 strings[i]=line[i];  
 }  
 **return** strings;  
 }  
 **public static** String[] ToDown(String[] line) {  
 String[] strings = **new** String[line.**length**];  
 **for** (**int** i=0; i<line.**length**; i++) {  
 strings[i]=line[i].toLowerCase();  
 }  
 **return** strings;  
 }  
 **public static** String[] RedColor(String[] line, **int** index, **int** length) {  
 String[] redStrings = **new** String[line.**length**];  
 redStrings=*Copy*(line);  
 String a;  
 **for** (**int** i=index; i<length+index; i++) {  
 a=redStrings[i];  
 redStrings[i]=***ANSI\_RED***+a+***ANSI\_RESET***;  
 }  
 **return** redStrings;  
 }  
 **public static void** time(){  
 **if** (*timer* == 0) *timer*=System.*nanoTime*();  
 **else** {  
 **double** val = (**double**)(System.*nanoTime*()-*timer*);  
 System.***out***.println(val/1000000);  
 *timer*=0;  
 }  
 }  
}

Скриншот работы программы

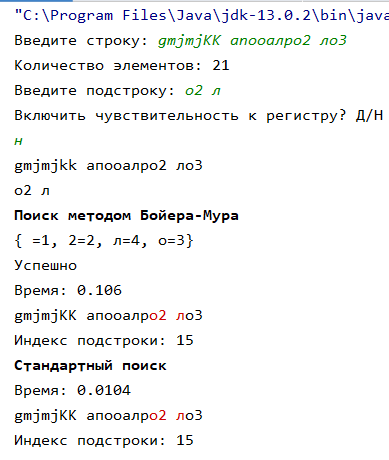


Рисунок 1 – Работа программы

Сравнение времени работы используемых функций поиска подстроки

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Количество элементов** | | 21 | 128 | 497 | 67 | 241 |
| **Индекс подстроки** | | 15 | 78 | 483 | 33 | 230 |
| **Время** | **Бойера-Мура** | 0,106 | 0,1033 | 0,3412 | 0,112 | 0,0844 |
| **Стандартный** | 0,0104 | 0,0101 | 0,0614 | 0,0156 | 0,0134 |

# **Вывод**

В ходе работы был изучен поиск подстроки в строке с помощью метода Бойера-Мура, и с помощью стандартного поиска. Исходя из полученных результатов, можно сделать вывод о том, что поиск методом Бойера-Мура медленнее стандартного. Чем дальше находится подстрока, тем дольше осуществляется поиск. При стандартном поиске время несильно отличается, и из данных результатов максимальная разница 0.0513 мс, а при поиске методом Бойера-Мура разница составляет 0,2568 мс.