

# DESIGN ANALYSIS AND ALGORITHMS

## LAB ASSIGNMENT-9

NAME: PRATHAPANI SATWIKA

REG.NO.: 20BCD7160

**Write a program that uses Branch and bound algorithm to solve the String Matching problem**

**CODE :**

```
package Lab9;
import java.util.*;
public class StringMatchingBranchandBound{
    static int NO_OF_CHARS = 256;
    static int max (int a, int b) { return (a > b)? a: b; }
    static void badCharHeuristic( char []str, int size,int badchar[])
    {
        for (int i = 0; i < NO_OF_CHARS; i++)
            badchar[i] = -1;
        for (int i = 0; i < size; i++)
            badchar[(int) str[i]] = i;
    }
    static void search( char txt[], char pat[])
    {
        int m = pat.length;
        int n = txt.length;
        int badchar[] = new int[NO_OF_CHARS];
        badCharHeuristic(pat, m, badchar);
        int s = 0;
        while(s <= (n - m))
        {
            int j = m-1;
            while(j >= 0 && pat[j] == txt[s+j])
                j--;
            if (j < 0)
            {
                System.out.println("Patterns occur at shift = " + s);
                s += (s+m < n)? m-badchar[txt[s+m]] : 1;
            }
            else
                s += max(1, j - badchar[txt[s+j]]);
        }
    }
    public static void main(String []args) {
        char txt[] = "ABAAABCD".toCharArray();
```

```

        char pat[] = "ABC".toCharArray();
        search(txt, pat);
    }
}

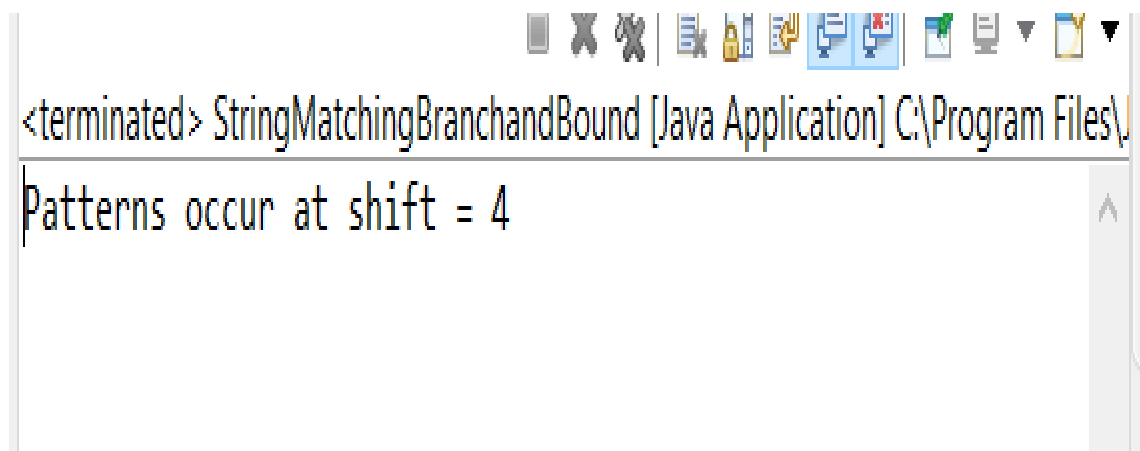
```

```

1 package Lab9;
2 import java.util.*;
3 public class StringMatchingBranchandBound{
4     static int NO_OF_CHARS = 256;
5     static int max (int a, int b) { return (a > b)? a: b; }
6     static void badCharHeuristic( char []str, int size,int badchar[])
7     {
8         for (int i = 0; i < NO_OF_CHARS; i++)
9             badchar[i] = -1;
10        for (int i = 0; i < size; i++)
11            badchar[(int) str[i]] = i;
12    }
13    static void search( char txt[], char pat[])
14    {
15        int m = pat.length;
16        int n = txt.length;
17        int badchar[] = new int[NO_OF_CHARS];
18        badCharHeuristic(pat, m, badchar);
19        int s = 0;
20        while(s <= (n - m))
21        {
22            int j = m-1;
23            while(j >= 0 && pat[j] == txt[s+j])
24                j--;
25            if (j < 0)
26            {
27                System.out.println("Patterns occur at shift = " + s);
28                s += (s+m < n)? m-badchar[txt[s+m]] : 1;
29            }
30        }
31        else
32            s += max(1, j - badchar[txt[s+j]]);
33    }
34 }
35 public static void main(String []args) {
36     char txt[] = "ABAAABCD".toCharArray();
37     char pat[] = "ABC".toCharArray();
38     search(txt, pat);
39 }
40 }

```

**OUTPUT :**



```
<terminated> StringMatchingBranchandBound (Java Application) C:\Program Files\  
Patterns occur at shift = 4
```