Experiment No: 10

Student Name : Tejinderpal Singh
UID: 23BCS80343
Branch: BE-CSE
Section/Group: KGR-3(A)
Semester: 5th
Subject Name: DAA

Subject Code: 23CSH-301
Date of Performance:3th Nov, 2025

Aim: Develop a program and analyze complexity to find all occurrences of a pattern P in a given stringS.

Objective: Analyze to find all occurrences of a pattern P in a given string S.

Procedure/Algorithm/Pseudocode:

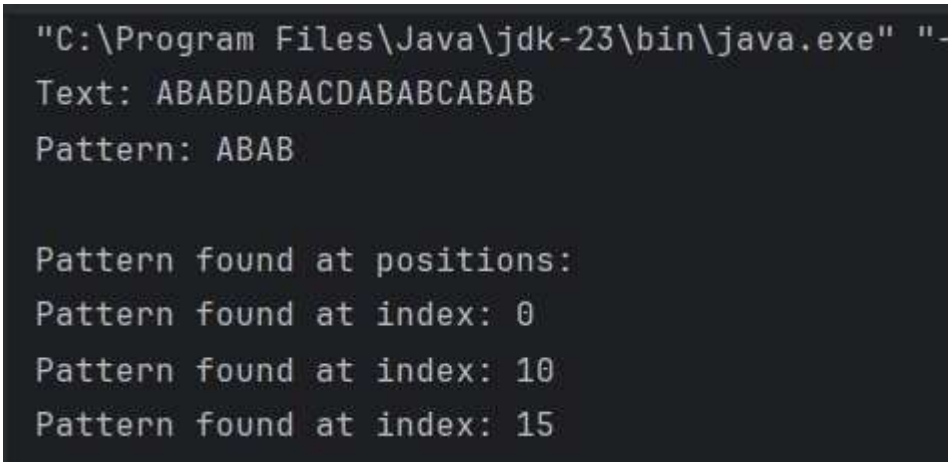
Algorithm: Naive Pattern Matching

1. Input:
   A text string S of length n and a pattern string P of length m.
2. Initialize:
   Start scanning S from index i = 0 to i = n - m.
3. For each position i:
   - Compare P[0...m-1] with the substring S[i...i+m-1].
   - If all characters match, record position i as a match.
4. Increment i and repeat the process until the end of the text.
5. Output:
   All starting indices where the pattern occurs in the string.

Code:

```
package DP;

public class PatternMatching {

    public static void findPattern(String text, String pattern) {
        int n = text.length();
        int m = pattern.length();
        boolean found = false;

        System.out.println("Text: " + text);
        System.out.println("Pattern: " + pattern);
        System.out.println("\nPattern found at positions:");

        for (int i = 0; i <= n - m; i++) {
            int j;

            for (j = 0; j < m; j++) {
                if (text.charAt(i + j) != pattern.charAt(j))
                    break;
            }

            if (j == m) {
                System.out.println("Pattern found at index: " + i);
                found = true;
            }
        }

        if (!found) {
            System.out.println("No occurrences found.");
        }
    }

    public static void main(String[] args) {
        String text = "ABABDABACDABABCABAB";
        String pattern = "ABAB";
        findPattern(text, pattern);
    }
}
```

Output :



Time Complexity :

Time Complexity

O((V + E) log V)

Using PriorityQueue (Min-Heap)

Learning Outcomes:

1. Understood the concept of pattern matching in strings.
2. Learned how to compare substrings and detect occurrences of a pattern.
3. Implemented the Naive String Matching algorithm in Java.
4. Analyzed time and space complexity for text search problems.
5. Built a foundation to explore efficient algorithms like KMP and Rabin-Karp.