East West University
Department of Computer Science and Engineering

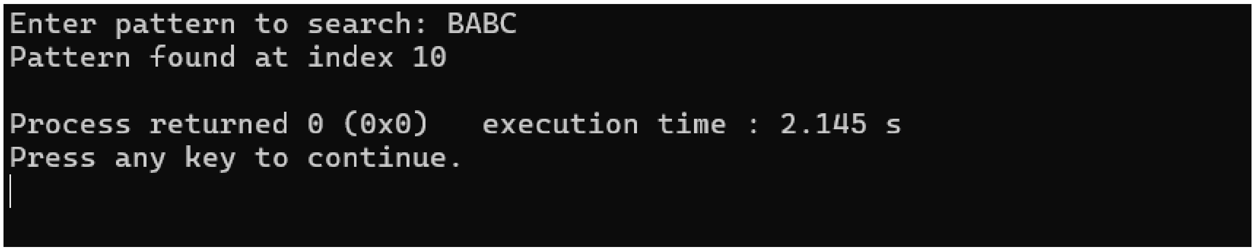
Course: CSE 246(Algorithms)
Section - 02

Submitted To :	**Submitted By :**
Dr. Md. Tauhid Bin Iqbal Assistant Professor Department of CSE East West University	**Name** : Hafsa Ferdousi **Student ID** : 2023-3-60-321 **Department** : CSE **Date** : 23 / 11 / 2025

Lab Report:07

1. Naive Pattern Matching

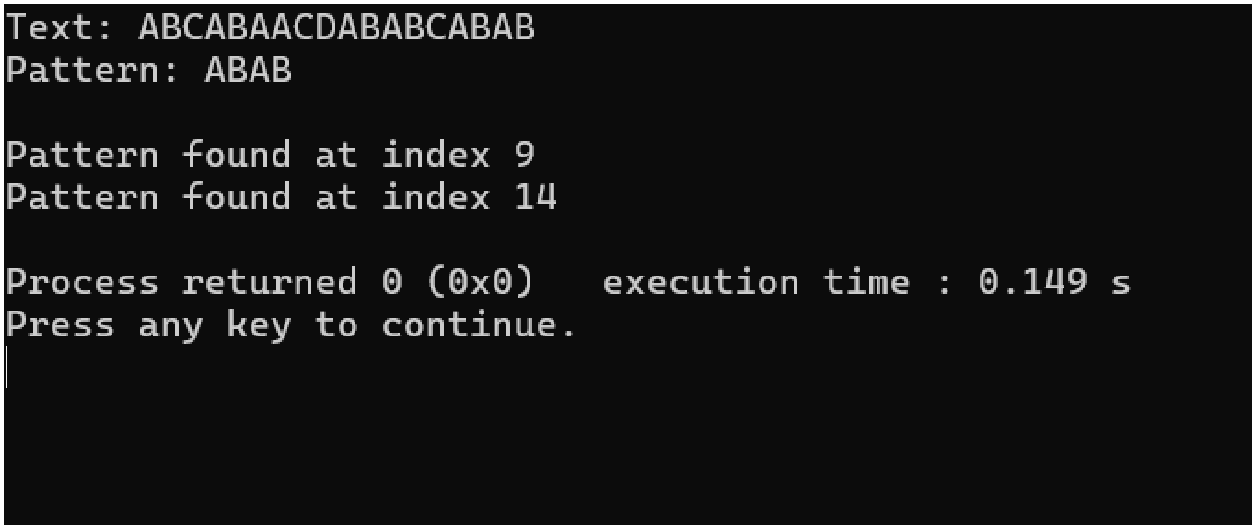
```
#include <bits/stdc++.h>
using namespace std;

void NaivePatternSearch(string text, string pattern) {
    int n = text.length();
    int m = pattern.length();

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

        int j = 0;
        while (j < m && text[i + j] == pattern[j]) {
            j++;
        }

        if (j == m) {
            cout << "Pattern found at index " << i << endl;
        }
    }
}

int main() {

    string text = "ABCABAACDABABCABAB";
    string pattern;

    cout << "Enter pattern to search: ";
    cin >> pattern;

    NaivePatternSearch(text, pattern);

    return 0;
}
```



```
Enter pattern to search: BABC
Pattern found at index 10

Process returned 0 (0x0)   execution time : 2.145 s
Press any key to continue.
```

2. Knuth-Morris-Pratt (KMP) Algorithm

```
#include <bits/stdc++.h>
using namespace std;

void computeLPSArray(string pattern, vector<int> &lps) {
    int m = pattern.length();
    int length = 0;
    lps[0] = 0;

    int i = 1;
    while (i < m) {
        if (pattern[i] == pattern[length]) {
            length++;
            lps[i] = length;
            i++;
        }
        else {
            if (length != 0) {
                length = lps[length - 1];
            }
            else {
                lps[i] = 0;
                i++;
            }
        }
    }
}


void KMPAlgorithm(string text, string pattern) {
    int n = text.length();
    int m = pattern.length();

    vector<int> lps(m);
    computeLPSArray(pattern, lps);

    int i = 0;
    int j = 0;

    while (i < n) {
        if (text[i] == pattern[j]) {
            i++;
            j++;
        }

        if (j == m) {
            cout << "Pattern found at index " << (i - j) << endl;
            j = lps[j - 1];
        }
        else if (i < n && text[i] != pattern[j]) {
            if (j != 0)
                j = lps[j - 1];
            else
                i++;
        }
    }
}

int main() {

    string text = "ABCABAACDABABCABAB";
    string pattern = "ABAB";

    cout << "Text: " << text << endl;
    cout << "Pattern: " << pattern << endl << endl;

    KMPAlgorithm(text, pattern);

    return 0;
}
```



```
Text: ABCABAACDABABCABAB
Pattern: ABAB

Pattern found at index 9
Pattern found at index 14

Process returned 0 (0x0)   execution time : 0.149 s
Press any key to continue.
```