# Experiment:3.2

**Student Name: Virat Samdarshi** **UID: 22BCS12648**
**Branch: B.E CSE** **Section/Group:IOT-627-B**
**Semester:5th** **DateofPerformance:23-10-24**
**Subject Name:DAAlab** **SubjectCode:22CSH-311**

1. **Aim:** Develop a program and analyze complexity to find all occurrences of a pattern P in a given string S.

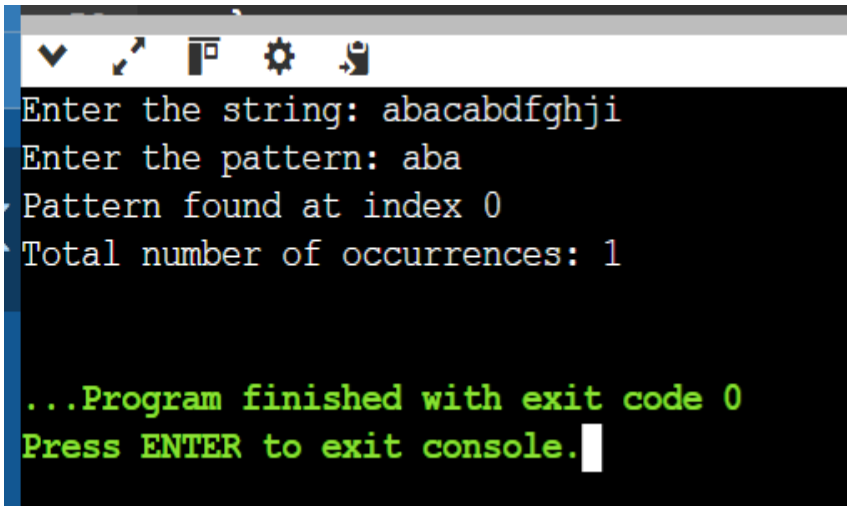
2. **Objective:** The goal of this program is to find all occurrences of a given pattern P in a larger text string S. This involves searching for every position in S where the substring P matches exactly.

**3.Implementation/Code:**

```
 #include <iostream>
 #include <vector>
using namespace std;
void computeLPSArray(string P, int M, vector<int> &lps) {
  int length = 0;
  lps[0] = 0;
  int i = 1;
  while (i < M) {
    if (P[i] == P[length]) {
      length++;
      lps[i] = length;
      i++;
    } else {
      if (length != 0) {
        length = lps[length - 1];
      } else {
        lps[i] = 0;
        i++;
      }
    }
  }
}

void KMPSearch(string P, string S) {
  int M = P.size();
```

```
    int N = S.size();
    vector<int> lps(M);
    computeLPSArray(P, M, lps);

    int i = 0;
    int j = 0;
    int occurrences = 0;
    while (i < N) {
        if (P[j] == S[i]) {
            i++;
            j++;
        }

        if (j == M) {
            cout << "Pattern found at index " << i - j << endl;
            occurrences++;
            j = lps[j - 1];
        } else if (i < N && P[j] != S[i]) {
            if (j != 0)
                j = lps[j - 1];
            else
                i++;
        }
    }

    cout << "Total number of occurrences: " << occurrences << endl;
}

int main() {
    string S, P;

    cout << "Enter the string: ";
    getline(cin, S);

    cout << "Enter the pattern: ";
    getline(cin, P);

    KMPSearch(P, S);

    return 0;
}
```

**4.Output:**




**5.Time Complexity:** O(n+m)