Experiment: 3.3

Student Name: Aniket Kumar
UID: 20BCS5306
Branch: BE-CSE
Section/Group: WM_703-B
Semester: 5th
Subject Code: 20CSP-312
Subject Name: DAA Lab
Date of Performance: 07/11/2022

1. Aim:

Code and analyze to find all occurrences of a pattern P in a given string S.

2. Task:

To find all occurrences of a pattern P in a given string S.

3. Software Used:

1. Visual Studio Code
2. MinGW
3. C++ compiler

4. Code:

```
#include <bits/stdc++.h> using namespace std;
void computeLPSArray(char* pat, int M, int* lps);
void KMPSearch(char* pat, char* txt)
{
        int M = strlen(pat); int N =
        strlen(txt); int lps[M];
        computeLPSArray(pat, M,
        lps);
        int i = 0; int j = 0; while
        ((N - i) >= (M - j)) { if
        (pat[j] == txt[i]) { j++;
        i++; }
                if (j == M) { printf("Found pattern at index %d
                        ", i - j); j = lps[j - 1];
                }
```

```
                else if (i < N && pat[j] != txt[i])
                        { if (j != 0) j = lps[j - 1];
                        else i = i + 1;
                }
        } } void computeLPSArray(char* pat, int
M, int* lps)
{
        int len = 0; lps[0] = 0; int i =
        1; while (i < M) { if (pat[i] ==
        pat[len]) { len++;
                        lps[i] = len;
                i++; }
                else
                {
                        if (len != 0) { len =
                                lps[len - 1];
                        }
                        else
                        {
                                lps[i] = 0;
                                i++;
                        }
                }
        }
}

int main()
{ char S[] = "ABABDABACDABABCABAB"; char
        P[] = "ABABCABAB"; KMPSearch(P, S);
        return 0;
}
```

5. Output:

```
Found pattern at index 10
```

6. Time Complexity:-

The time complexity of this algorithm will be O(n) and if we use the Naive algorithm for solving the same problem then in the worst case it will take O(m(n-m+1)) depending upon the strength of the Hash Function.

Learning outcomes:

1. Learned about Dynamic programming
2. Learned about optimization techniques
3. Learned about the knapsack problem
4. Learned about different ways of solving knapsack problem