

ANALYSIS & DESIGN OF ALGORITHM

PRACTICAL-2

190031968

Kinthali praneeth

IN-LAB:

1)

```
def search(pat, txt):
    M = len(pat)
    N = len(txt)

    # A loop to slide pat[] one by one */
    for i in range(N - M + 1):
        j = 0

        # For current index i, check
        # for pattern match */
        while(j < M):
            if (txt[i + j] != pat[j]):
                break
            j += 1

        if (j == M):
            print("Pattern found at index ", i)

# Driver Code
if __name__ == '__main__':
    txt = "AABAACAADAABAAABAA"
    pat = "AABA"
    search(pat, txt)
```

PYTH 3.6 (Python 3.6)    

```
1 def search(pat, txt):
2     M = len(pat)
3     N = len(txt)
4
5     # A loop to slide pat[] one by one */
6     for i in range(N - M + 1):
7         j = 0
8
9         # For current index i, check
10        # for pattern match */
11        while(j < M):
12            if (txt[i + j] != pat[j]):
13                break
14            j += 1
15
16        if (j == M):
17            print("Pattern found at index ", i)
18
19 # Driver Code
20 if __name__ == '__main__':
21     txt = "AABAACAADAABAAABAA"
22     pat = "AABA"
23     search(pat, txt)
24
```

Status Successfully executed **Date** 2021-08-04 16:05:43 **Time** 0.03 sec **Mem** 17.968 kB



Output

```
Pattern found at index 0
Pattern found at index 9
Pattern found at index 13
```

2)

Python program for KMP Algorithm

```
def KMPSearch(pat, txt):
```

```
    M = len(pat)
```

```
    N = len(txt)
```

```
    lps = [0]*M
```

```
    j = 0 # index for pat[]
```

```
    computeLPSArray(pat, M, lps)
```

```
    i = 0
```

```
    while i < N:
```

```
        if pat[j] == txt[i]:
```

```
            i += 1
```

```

    j += 1

if j == M:
    print ("Found pattern at index " + str(i-j))
    j = lps[j-1]

elif i < N and pat[j] != txt[i]:

    if j != 0:
        j = lps[j-1]
    else:
        i += 1
def computeLPSArray(pat, M, lps):
    len = 0 # length of the previous longest prefix suffix

    lps[0] # lps[0] is always 0
    i = 1
    while i < M:
        if pat[i] == pat[len]:
            len += 1
            lps[i] = len
            i += 1
        else:

            if len != 0:
                len = lps[len-1]

            # Also, note that we do not increment i here
        else:
            lps[i] = 0
            i += 1

txt = "b c m a l m n x y z"
pat = " m a l"
KMPSearch(pat, txt)

```

PYTH 3.6 (Python 3.6)



```
1 # Python program for KMP Algorithm
2 def KMPSearch(pat, txt):
3     M = len(pat)
4     N = len(txt)
5
6     lps = [0]*M
7     j = 0 # index for pat[]
8
9     computeLPSArray(pat, M, lps)
10
11     i = 0
12     while i < N:
13         if pat[j] == txt[i]:
14             i += 1
15             j += 1
16
17         if j == M:
18             print ("Found pattern at index " + str(i-j))
19             j = lps[j-1]
20
21         elif i < N and pat[j] != txt[i]:
22
23             if j != 0:
24                 j = lps[j-1]
25             else:
26                 i += 1
27
28 def computeLPSArray(pat, M, lps):
```

PYTH 3.6 (Python 3.6)



```
24     j = lps[j-1]
25 else:
26     i += 1
27
28 def computeLPSArray(pat, M, lps):
29     len = 0 # length of the previous longest prefix suffix
30
31     lps[0] # lps[0] is always 0
32     i = 1
33
34     while i < M:
35         if pat[i] == pat[len]:
36             len += 1
37             lps[i] = len
38             i += 1
39         else:
40
41             if len != 0:
42                 len = lps[len-1]
43
44             # Also, note that we do not increment i here
45         else:
46             lps[i] = 0
47             i += 1
48
49     txt = "b c m a l m n x y z"
50     pat = "m a l"
51     KMPSearch(pat, txt)
```

Status Successfully executed Date 2021-08-04 16:09:28 Time 0.02 sec Mem 17.968 kB



Output

```
Found pattern at index 3
```