```cpp
//KMP Algorithm for pattern match in a string
//Time complexity = O(m+n) ---------- m=length of pattern, n=length of text

#include<bits/stdc++.h>
using namespace std;
void KMP(string ,string); //function for KMP that takes two parameters of String type first
void computeLPSarray(string,int [],int);
int main()
{
    string text = "ABABDABACDABABCABAB";
    string pattern = "ABABCABAB";
    KMP(text,pattern);
    return 0;

}
void computeLPSarray(string pattern,int lps[],int m)
{
    int i=1,len=0;
    lps[0]=0; //Initially lps[0] = 0 always
    while(i<m)
    {
        if(pattern[len]==pattern[i])
        {
            len++;
            lps[i]=len;
            i++;
        }

        else
        {
            if(len>0)
            {
                len=lps[len-1];
            }
            else
            {
                lps[i]=0; //Because of not matched and len reached to 0 so lps[i] will be zero
                i++; //increase i by 1 to match next
            }
        }
    }
}
void KMP(string text,string pattern)
{
    int n=text.length();
    int m=pattern.length();
    int i=0,j=0;
    int lps[m]; //LPS array to stores lps of pattern
    computeLPSarray(pattern,lps,m);//Compute the LPS array
    while(i<n)
    {
        if(text[i]==pattern[j])
        {
            i++;
            j++;
        }
        if(j==m)
        {
            cout<<"Pattern matched at indexed start from "<<i-j<<endl;
            j=lps[j-1];
        }
        else if(i<n && pattern[j]!=text[i])
        {
            if(j!=0)
            {
                j=lps[j-1];
            }
            else
            {
                i++;
            }
        }
    }
}
```