```
#include <iostream>
#include<vector>
#include<algorithm>
using namespace std;

void getLPSArray(char pat[], int M, int lps[]) {
    int len = 0;
    int i = 1;
    lps[0] = 0; //lps[0] is always 0
    //the loop calculates lps[i] for i=1 to M-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++;
            }
        }
    }
}

void KMPSearch(char pat[], char txt[]) {
    int const M = 4;
    int N = 11;
    int lps[M];
    //int lps[] = new int[M];
    int j = 0; //index for path[]
    getLPSArray(pat, M, lps);
    int i = 0; //index for txt[]
    while (i < N) {
        if (pat[j] == txt[i]) {
            j++;
            i++;
        }
        if (j == M) {
            cout << "\nIncidencia encontrada en el indice: " << (i - j);
            j = lps[j - 1];
        }//mismatch after j matches
```

```
        else if (i < N && pat[j] != txt[i]) {
            if (j != 0) {
                j = lps[j - 1];
            }
            else
                i++;
        }
    }
}



int main()
{

    char cadena[]={ 'A','A','A','B','A','A','C','A','B','A','A'};
    char pat[]={ 'A','B','A','A'};
    KMPSearch(pat,cadena);
    return 0;
}
```