|  |
| --- |
|  |

**Kmp:**

void lps\_ar(char \*pat,int M,int \*lps)

{ ll len=0; lps[0]=0; ll 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++;}}}

void KMPsearch(char \*txt,char \*pat)

{

ll N=strlen(txt);

ll M=strlen(pat);

ll lps[M];

lps\_ar(pat,M,lps);

ll i=0;

ll j=0;

total=0;

while(i<N)

{

//cout<<"kmp"<<endl;

if(pat[j]==txt[i]) i++,j++;

if(j==M)

{

//to print how many times match

total++;

//to print matched starting index

cout<<"found matched pattern at index: "<<i-j<<endl;

j=lps[j-1];

}

else if(i<N && pat[j]!=txt[i])

{

if(j!=0) j=lps[j-1];

else i++;

}

}

}

void solve()

{

char txt[1000009],pat[1000009];

cin>>txt>>pat;

KMPsearch(txt,pat);

cout<<"Case "<<cas++<<": "<<total<<endl;

}

**///BRACKET SEQUENCE:**

**void bracket(string s)**

**{**

**ll ct=0;**

stack<ll>q;

for(ll i=0;i<s.size();i++)

{

if(q.size()==0) q.push(s[i]);

else

{

if(s[i]==')')

{

if(q.top()=='(')

{

ct+=2;

q.pop();

}

}

else q.push(s[i]);

}

}

// numbner of bracket sequence

cout<<ct<<endl;

}