//KMP

//0-base

//fail[i]表示0~i-1的最长border

void getfail()

{

fail[0]=-1;

int k=-1,j=0;

while (j<=m-1)

{

if (k==-1 || b[j]==b[k])

fail[++j]=++k;

else

k=fail[k];

}

}

//统计共有多少个和模式串匹配的子串

int kmp\_count()

{

int ans=0;

while (i<=n-1)

{

while (i<=n-1 && j<=m-1)

{

if (j==-1 || a[i]==b[j])

{

i++;

j++;

}

else

j=fail[j];

}

if (j>=m)

{

ans++;

j=fail[j];

}

}

return ans;

}

//返回第一个与模式串匹配的位置

int kmp\_search()

{

i=0;j=0;

while (i<=n-1 && j<=m-1)

{

if (j==-1 || a[i]==b[j])

{

i++;

j++;

}

else

j=fail[j];

}

return j>=m?i+1-m:-1;

}

//Manacher

//ss原串；s加过"#"的串

//RL[i]：pos=i的最长半径

cur=1;

len=strlen(ss+1);

for (i=1;i<=len;i++)

{

s[cur++]='#';

s[cur++]=ss[i];

}

s[cur]='#';

len=cur;

maxright=pos=0;

int ans=0;

for (i=1;i<=len;i++)

{

if (i<maxright)

//RL[2\*pos-i]是i关于pos的对称位置的半径

RL[i]=min(RL[2\*pos-i],maxright-i);

else

RL[i]=1;

while (i-RL[i]>=1 && i+RL[i]<=len && s[i-RL[i]]==s[i+RL[i]]) RL[i]++;

ans=max(ans,RL[i]-1);

if (i+RL[i]>maxright)

{

maxright=i+RL[i];

pos=i;

}

}

printf("%d\n",ans);