[报告]　K

[source]　http://202.114.18.202:8080/judge/contest/view.action?cid=6236#problem/K

[Description]

有一个串，是由 $A+prefix+B+middle+C+suffix$ 组成的，而 $prefix+middle+suffix$ 为原串，原串为一个奇回文串，求最长原串是多少。注意以上每个部分均可能为空串。

[方法]

这个题好早就想出咋搞了，就是因为下标各种写错，想法各种脑残导致一直没有 A（连样例都过不去），今天总算没脑残了，给过了。

思路就是因为 suffix 是在最末尾，所以将原串反向，求以反串为模式串的 KMP，令 val[i]表示以第 i 为后缀的字符串和反串前缀的最长匹配数。令 pair<int,int> ans[N]，为第 i 位之前和反串的最长匹配数和该匹配数的结尾点。用 manacher 求一遍回文串，用 P[i]表示以 i 为中心点的最长奇回文子串的延一方向的伸展长度。最后 O(n)的扫一遍，令 L=i-P[i-1];R=i+P[i-1];找最大的

sum=(P[i-1]-1)*2+1+min(ans[L].second,n-R+1)*2;

```
#include<iostream>
#include<cstring>
#include<cstdio>
#include<algorithm>
#define N 111110
using namespace std;
char s[N],rs[N];
```

```
int P[N],next[N],val[N];
pair<int,int> ans[N],tt[3];
void manacher(int len)
{
    int m=0,id=0;
    P[0]=1;
    for(int i=1;i<len;i++)
    {
        if(i<=m)
        {
            int u=2*id-i;
            if(i+P[u]-1<m)
            {
                P[i]=P[u];
            }
            else
            {
                int v=m-i;
                while(i+v<len&&i>=v&&s[i+v]==s[i-v])
                {
                    v++;
                }
```

```
                P[i]=v;
                id=i;m=P[i]+i-1;
            }
        }
        else
        {
            int v=1;
            while(i+v<len&&i>=v&&s[i+v]==s[i-v])
                {
                    v++;
                }
                P[i]=v;
                id=i;m=P[i]+i-1;
        }

    }
}
void get_next(char *s,int n)
{
    next[0]=-1;
    int u=-1;
    for(int i=1;i<n;i++)
```

```
    {
        while(u>=0&&s[u+1]!=s[i])
        {
            u=next[u];
        }
        if(s[u+1]==s[i])
        {
            u++;
        }
        next[i]=u;
    }
}
void KMP(char *s,char *str)
{
    int n=strlen(s);
    int m=strlen(str);
    get_next(str,m);
    int u=-1;
    for(int i=0;i<n;i++)
    {
        while(u>-1&&str[u+1]!=s[i])
            u=next[u];
```

```
            if(str[u+1]==s[i])
            {
                u++;
            }
            val[i]=u+1;
        }
}
void debug(int a[],int n)
{
    for(int i=0;i<n;i++)
    {
        printf("%d ",a[i]);
    }
    printf("\n");
}
int main(void)
{
    scanf("%s",s);
    int n=strlen(s);
    manacher(n);
    for(int i=0;i<n;i++)
    {
```

```
    rs[i]=s[n-i-1];
}
rs[n]='\0';
get_next(rs,n);
KMP(s,rs);
ans[0]=make_pair(0,0);
for(int i=1;i<=n;i++)
{
    ans[i]=ans[i-1];
    if(val[i-1]>ans[i].second)
    {
        ans[i]=make_pair(i,val[i-1]);
    }
}
int mans=0;
int flag=0;
for(int i=1;i<=n;i++)
{
    int sum=0,tmp;
    int L,R;
    if(i-P[i-1]>=0)
    {
```

```
            sum=(P[i-1]-1)*2+1;
            L=i-P[i-1];
            R=i+P[i-1];
            tmp=min(ans[L].second,n-R+1);
            sum+=tmp*2;
        }
        if(mans<sum)
        {
            mans=sum;
            tt[0]=make_pair(ans[L].first-tmp+1,tmp);
            tt[1]=make_pair(L+1,(P[i-1]-1)*2+1);
            tt[2]=make_pair(n-tmp+1,tmp);
        }
    }
    if(tt[0].second==0)
    {
        printf("1\n");
    }
    else
    {
        printf("3\n");
    }
```

```
    for(int i=0;i<3;i++)
    {
        if(tt[i].second!=0)
        printf("%d %d\n",tt[i].first,tt[i].second);
    }
    return 0;
}
```