RMQ题杂

RMQ题杂
[NOI2010] 超级钢琴
BJOI 2020]封印：SAM+RMQ
「一本通 4.2 例 3」与众不同

[NOI2010] 超级钢琴

有 n个音符，编号为 1 至 n 。第 i 个音符的美妙度为 A_i 。

我们要找到 k 段超级和弦组成的乐曲，每段连续的音符的个数 x 满足 $L \leq x \leq R$ ，求乐曲美妙度的最大值。

```
#include<bits/stdc++.h>
using namespace std;
typedef long long ll;
const int N=5e5+7,lg=20;
inline int Max(int x,int y){return x>y?x:y;}
inline int Min(int x,int y){return x<y?x:y;}
ll sum[N],st[N][lg];

namespace RMQ{
    void init(int n){
        for(int i=1;i<=n;++i) st[i][0]=i;
        for(int j=1;(1<<j)<=n;++j){
            for(int i=1;i+(1<<j)-1<=n;++i){
                int x=st[i][j-1],y=st[i+(1<<(j-1))][j-1];
                st[i][j]=(sum[x]>sum[y])?x:y;
            }
        }
    }
    int query(int l,int r){
        int k=log2(r-l+1);
        int x=st[l][k],y=st[r-(1<<k)+1][k];
        return sum[x]>sum[y]?x:y;
    }
}

struct element{
    int o,l,r,t;
    element(){}
    element(int o,int l,int r):o(o),l(l),r(r),t(RMQ::query(l,r)){}
    friend bool operator <(const element&a,const element &b){
        return sum[a.t]-sum[a.o-1]<sum[b.t]-sum[b.o-1];
    }
};

std::priority_queue<element>Q;
```

```
signed main(){
    ios::sync_with_stdio(0);
    cin.tie(0);cout.tie(0);
    int n,k,L,R;
    cin>>n>>k>>L>>R;
    for(int i=1;i<=n;++i) cin>>sum[i],sum[i]+=sum[i-1];
    RMQ::init(n);
    for(int i=1;i<=n;++i) if(i+L-1<=n) Q.emplace(element(i,i+L-1,min(i+R-1,n)));
    ll ans=0;
    while(k--){
        int o=Q.top().o,l=Q.top().l,r=Q.top().r,t=Q.top().t;
        Q.pop();
        ans+=sum[t]-sum[o-1];
        if(l!=t) Q.emplace(element(o,l,t-1));
        if(r!=t) Q.emplace(element(o,t+1,r));
    }
    cout<<ans<<"\n";
    return 0;
}
```

BJOI 2020]封印：SAM+RMQ

给定只含小写字母a,b的两个字符串s,t，q次询问，每次询问$s[l...r]$和t的最长公共子串的长度。

```
#include<bits/stdc++.h>
using namespace std;

const int N=2e5+5;
char s[N],t[N];
int n,m,q,pre[N],pw[20],st[20][N];
int rt,las,node_cnt,ch[N<<1][2],fa[N<<1],len[N<<1];

inline void extend(int c){  //SAM添加字符
    int p=las,np=++node_cnt,q,nq; len[las=np]=len[p]+1;
    for(;p&&ch[p][c]==0;p=fa[p]) ch[p][c]=np;
    if(!p){ fa[np]=1; return; }
    if(len[p]+1==len[q=ch[p][c]]){ fa[np]=q; return; }
    len[nq=++node_cnt]=len[p]+1; fa[nq]=fa[q];
    for(int i=0;i<=1;++i) ch[nq][i]=ch[q][i];
    for(;p&&ch[p][c]==q;p=fa[p]) ch[p][c]=nq;
    fa[np]=fa[q]=nq;
}

inline int query(int l,int r){ //st表查询[l,r]区间最大值
    if(l>r) return 0; int k=log2(r-l+1);
    return max(st[k][l],st[k][r-pw[k]+1]);
}

signed main(){
    scanf("%s%s",s+1,t+1);
    scanf("%d",&q);
    n=strlen(s+1);m=strlen(t+1);
    rt=las=++node_cnt;
    for(int i=1;i<=m;++i) extend(t[i]-'a'); //对模式串建SAM
    for(int i=1,u=rt,leng=0;i<=n;++i){  //统计每个位置出发算出最大长度
```

```
        int c=s[i]-'a';
        while(u&&!ch[u][c]) u=fa[u],leng=len[u];
        if(ch[u][c]) u=ch[u][c],++leng; else u=rt;
        pre[i-leng+1]=i; st[0][i]=leng;
    }
    for(int i=2;i<=n;++i) pre[i]=max(pre[i],pre[i-1]); //前i位能够匹配的最大长度
    pw[0]=1; for(int j=1;j<=17;++j) pw[j]=pw[j-1]<<1;
    for(int j=1;j<=17;++j) for(int i=1;i<=n;++i)  //st表转移
        st[j][i]=max(st[j-1][i],st[j-1][i+pw[j-1]]);
    for(int i=1,l,r,p;i<=q;++i){   //静态查询
        scanf("%d%d",&l,&r);
        p=min(max(l-1,pre[l]),r);
        printf("%d\n",max(p-l+1,query(p+1,r)));
    }
    return 0;
}
```

「一本通 4.2 例 3」与众不同

给长度为n的序列a和m次询问，每次问$[L,R]$区间内最长的完美序列长度，定义完美序列为一段连续的序列满足序列中的数互不相同。

```
#include<bits/stdc++.h>
#define inf 0x3f3f3f3f
#define int long long
using namespace std;
const int N=2e5+7;
const int mod=1e9+7;

int n,m,a[N],st[N];
int lg[N],f[N][25];
map<int,int>pr;

void init(){  //初始化对数
    lg[1]=0;
    for(int i=2;i<N;i++) lg[i]=lg[i/2]+1;
}

void GM(){
    for(int j=1;j<=22;j++){
        for(int i=1;i+(1<<j)-1<=n;i++){
            f[i][j]=max(f[i][j-1],f[i+(1<<(j-1))][j-1]); //记录i到i+2^j的最大f[j]
        }
    }
}

int Getmax(int x,int y){
    int s=lg[y-x+1],ans=max(f[x][s],f[y-(1<<s)+1][s]);
    return ans;
}

int Solve(int l,int r){ //找到st[x]>=L的位置，前面的答案取x-L,后面的答案RMQ
    if(st[l]==l) return l;
    if(st[r]<=l) return r+1;
    int L=l,R=r;
```

```
    while(l<r){
        int mid=(l+r)>>1;
        if(st[mid]>=L) r=mid;
        else l=mid+1;
    }
    return l;
}

signed main(){
    ios::sync_with_stdio(0);
    cin.tie(0);cout.tie(0);
    cin>>n>>m;
    for(int i=1;i<=n;i++) cin>>a[i];
    for(int i=1;i<=n;i++){
        st[i]=max(st[i-1],pr[a[i]]+1); //记录以i结尾的最长完美序列起点
        pr[a[i]]=i;  //记录上一个a[i]的位置
        f[i][0]=i-st[i]+1; //以i结尾的最长完美序列长度
    }
    init();
    GM();
    for(int i=1,l,r;i<=m;i++){
        cin>>l>>r;
        l++;r++;
        int L=l,R=r,ans=0;
        l=Solve(L,R);
        if(l>L) ans=l-L;
        if(l<=R) ans=max(ans,Getmax(l,R)); //这一部分保证了st[i]>=l
        cout<<ans<<"\n";
    }
    return 0;
}
```