2020牛客暑期多校训练营（第九场）

比赛网址

训练结果

- 时间:2020-08-17 13:00~18:00
- rank:61/951
- 完成情况：5/6/11

I.Just Skip The Problem

题意

给了一个字符串，问你有多少子串，自身是回文串而且一半也是回文串

题解

由回文自动机的性质知道一个串的本质不同的回文串最多有 n 个，我们可以用回文自动机统计不同回文串的个数并标记位置，之后枚举每个串并用哈希判断他的一半是不是回文串就行

```
#include<iostream>
#include<cstdio>
#include<algorithm>
#include<cstring>
#define ll long long
using namespace std;
const int N=300055;
char s[N];
int fail[N],len[N],last,n,tot,ch[N][26],cnt[N],ps[N],ans[N];
ll hs[N],pw[N],hs2[N];
int newnode(int x)
{
    len[++tot]=x;return tot;
}

void init()
{
    tot=-1;last=0;
    newnode(0);newnode(-1);
    fail[0]=1;s[0]=-1;
}

int getfail(int pos,int x)
```

```
{
    while(s[pos-len[x]-1]!=s[pos]) x=fail[x];
    return x;
}

void add(int x,int c)
{
    int cur=getfail(x,last);
    if(!ch[cur][c])
    {
        int now=newnode(len[cur]+2);
        fail[now]=ch[getfail(x,fail[cur])][c];
        ch[cur][c]=now;
    }
    cnt[last=ch[cur][c]]++;ps[last]=x;
}
ll query(int l,int r,int opt)
{
    int len=r-l+1;
    if(opt==1)
    {
        return hs[r]-hs[l-1]*pw[len];
    }
    else
    {
        return hs2[l]-hs2[r+1]*pw[len];
    }
}
int chk(int l,int r)
{
    int mid=l+r>>1,len=r-l+1;
    if(len&1)
    {
        return query(l,mid,1)==query(mid,r,2);
    }
    else return query(l,mid,1)==query(mid+1,r,2);
}
int main()
{
    pw[0]=1;
    for(int i=1;i<N;i++)
        pw[i]=pw[i-1]*31;
    while(scanf("%s",s+1)!=EOF)
    {
        n=strlen(s+1);init();
        for(int i=1;i<=n;i++)
        {
            add(i,s[i]-'a');//if(i<=3) cout<<last<<endl;
            hs[i]=hs[i-1]*31+(s[i]-'a'+1);
        }
```

```
        hs2[n+1]=0;
        for(int i=n;i;i--)
            hs2[i]=hs2[i+1]*31+(s[i]-'a'+1);
        for(int i=tot;i>1;i--)
        {
//            cout<<len[i]<<" "<<cnt[i]<<" "<<ps[i]<<endl;
            cnt[fail[i]]+=cnt[i];
            if(chk(ps[i]-len[i]+1,ps[i]-len[i]+(len[i]+1)/2))
                ans[len[i]]+=cnt[i];
        }
        printf("%d",ans[1]);ans[1]=0;
        for(int i=0;i<=tot;i++)
            len[i]=fail[i]=cnt[i]=ps[i]=0,memset(ch[i],0,sizeof(ch[i]));
        for(int i=2;i<=n;i++)
            printf(" %d",ans[i]),ans[i]=0;
        puts("");
    }
    return 0;
}
```

J.Just Skip The Problem

题意

求 n 的阶乘，水

K.Keen On Everything But Triangle

题意

给出一排长度不同的木棒，每次询问一个区间，求这个区间木棒能构成周长最长的三角形

题解

对于一部分木棒，一定是优先选最大的三个尝试组合

如果三个木棒不合法，那么最小的和最大的差别一定减半，所以可以暴力$O(logn)$枚举

那么枚举最大的是第几大，用线段树查询即可

```
#include<algorithm>
#include<iostream>
#include<cstdlib>
#include<cstring>
#include<cstdio>
```

```
#include<vector>
#include<queue>
#include<cmath>
#include<map>
#include<set>
#define LL long long int
#define REP(i,n) for (int i = 1; i <= (n); i++)
#define Redge(u) for (int k = h[u],to; k; k = ed[k].nxt)
#define cls(s,v) memset(s,v,sizeof(s))
#define mp(a,b) make_pair<int,int>(a,b)
#define cp pair<int,int>
using namespace std;
const int maxn = 100005,maxm = 10000005,INF = 0x3f3f3f3f;
inline int read(){
    int out = 0,flag = 1; char c = getchar();
    while (c < 48 || c > 57){if (c == '-') flag = 0; c = getchar();}
    while (c >= 48 && c <= 57){out = (out << 1) + (out << 3) + c - 48; c =
getchar();}
    return flag ? out : -out;
}
int rt[maxn],sum[maxm],ls[maxm],rs[maxm],pos,siz,n,Q;
LL a[maxn],b[maxn],tot;
void add(int& u,int v,int l,int r){
    u = ++siz; ls[u] = ls[v]; rs[u] = rs[v]; sum[u] = sum[v];
    if (l == r){sum[u]++;return;}
    int mid = (l + r) >> 1;
    if (mid >= pos) add(ls[u],ls[v],l,mid);
    else add(rs[u],rs[v],mid + 1,r);
    sum[u] = sum[ls[u]] + sum[rs[u]];
}
LL ans,L,R;
int kth(int u,int v,int l,int r,int k){
    if (l == r) return l;
    int mid = (l + r) >> 1;
    if (sum[rs[u]] - sum[rs[v]] >= k) return kth(rs[u],rs[v],mid + 1,r,k);
    return kth(ls[u],ls[v],l,mid,k - sum[rs[u]] + sum[rs[v]]);
}
void work(){
    while (Q--){
        L = read() - 1; R = read(); ans = -1;
        LL K = 1,t,tt,ttt;
        while (K + 2 <= R - L){
            t = b[kth(rt[R],rt[L],1,tot,K)];
            tt = b[kth(rt[R],rt[L],1,tot,K + 1)];
            ttt = b[kth(rt[R],rt[L],1,tot,K + 2)];
            if (t < tt + ttt) {ans = t + tt + ttt; break;}
            K++;
        }
        printf("%lld\n",ans);
    }
```

```
}
int main(){
    while (scanf("%d%d",&n,&Q)==2){
        siz = 0;
        REP(i,n) b[i] = a[i] = read();
        sort(b + 1,b + 1 + n); tot = 1;
        for (int i = 2; i <= n; i++) if (b[i] != b[tot]) b[++tot] = b[i];
        REP(i,n) a[i] = lower_bound(b + 1,b + 1 + tot,a[i]) - b;
        REP(i,n) pos = a[i],add(rt[i],rt[i - 1],1,tot);
        work();
    }
    return 0;
}
```

L.Longest Subarray

题意

一个序列每个位置有一种颜色，找到一个最长的子区间，使得每种颜色要么出现过超过K次，要么出现过0次

题解

从后往前枚举位置作为子区间的开端，每种颜色有两种状态：

1、数量不足，不能被包含，那么这些位置都被禁了，要选一个最大的子区间，那么右端点一定在最左边被禁的地方

2、当前数量足够K个，记录一下如果要包含这种颜色，最少要包含到哪个位置

那么每次只需要在线段树上找当前区间最大值有没有超过设定的右端点即可

```
#include<algorithm>
#include<iostream>
#include<cstdlib>
#include<cstring>
#include<cstdio>
#include<vector>
#include<queue>
#include<cmath>
#include<map>
#include<set>
#define LL long long int
#define REP(i,n) for (int i = 1; i <= (n); i++)
#define Redge(u) for (int k = h[u],to; k; k = ed[k].nxt)
#define cls(s,v) memset(s,v,sizeof(s))
#define mp(a,b) make_pair<int,int>(a,b)
#define cp pair<int,int>
```

```
#define ls (u << 1)
#define rs (u << 1 | 1)
using namespace std;
const int maxn = 100005,maxm = 100005,INF = 0x3f3f3f3f;
inline int read(){
    int out = 0,flag = 1; char c = getchar();
    while (c < 48 || c > 57){if (c == '-') flag = 0; c = getchar();}
    while (c >= 48 && c <= 57){out = (out << 1) + (out << 3) + c - 48; c =
getchar();}
    return flag ? out : -out;
}
int n,C,K,ans,A[maxn];
vector<int> pos[maxn];
int sc[maxn],top;
int mx[4 * maxn],L,R;
void modify(int u,int l,int r,int v){
    if (l == r){mx[u] = v; return;}
    int mid = (l + r) >> 1;
    if (mid >= L) modify(ls,l,mid,v);
    else modify(rs,mid + 1,r,v);
    mx[u] = max(mx[ls],mx[rs]);
}
int query(int u,int l,int r){
    if (l >= L && r <= R) return mx[u];
    int mid = (l + r) >> 1;
    if (mid >= R) return query(ls,l,mid);
    else if (mid < L) return query(rs,mid + 1,r);
    return max(query(ls,l,mid),query(rs,mid + 1,r));
}
void build(int u,int l,int r){
    if (l == r){mx[u] = 0; return;}
    int mid = (l + r) >> 1;
    build(ls,l,mid);
    build(rs,mid + 1,r);
    mx[u] = 0;
}
void work(){
    build(1,1,n); ans = 0; top = 0;
    for (int i = n; i; i--){
        int u = A[i];
        pos[u].push_back(i);
        sc[++top] = u;
        while (pos[sc[top]].size() >= K) top--;
        if (pos[u].size() >= K) {
            if (pos[u].size() > K){
                L = pos[u][pos[u].size() - 2];
                modify(1,1,n,0);
            }
            L = i;
            modify(1,1,n,pos[u][pos[u].size() - K]);
```

```
        }
        L = i; R = top ? pos[sc[top]][pos[sc[top]].size() - 1] - 1 : n;
        if (L <= R){
            if (query(1,1,n) <= R) ans = max(ans,R - L + 1);
        }
    }
    REP(i,C) pos[i].clear();
    printf("%d\n",ans);
}
int main(){
    while (~scanf("%d%d%d",&n,&C,&K)){
        REP(i,n) A[i] = read();
        work();
    }
    return 0;
}
```

训练实况

训练总结

wxg:

hxm:比较遗憾B最后差一点没写出

fyh: