2020牛客暑期多校训练营（第八场）

比赛网址

训练结果

- 时间:2020-08-03 12:00~17:00
- rank:45/685
- 完成情况：4/4/11

题解

E.Enigmatic Partition

题意

n的一个神秘分割定义如下：

$n=a_1+a_2+\ldots+a_m$满足

1□a_i递增

2□$a_m-a_1=2$

3□$a_{i+1}-a_i \leq 1$

设$f(n)$为n的神秘分割数量，求$\sum_{i=l}^{r}f(i)$

题解

依照题意，分割是由三段相邻的数$x-1,x,x+1$组成的

我们枚举长度m和中间的数x□那么经过讨论，能构造的数范围是$[mx-m+3,mx+m-3]$□而且其中能构造的数量分别从两侧到中间以$1,1,2,2,..$的规律递增，那么只需奇偶分别开两个差分数组进行二阶差分即可统计答案

枚举m□再枚举x

复杂度$O(n \log n)$

```
#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 = 300005,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;
}
LL f[maxn],n = 100000;
LL d1[maxn],d2[maxn];
LL D1[maxn],D2[maxn];
void add1(int l,int r){
    if (l > r) return;
    d1[l] += 1; d1[r + 1] -= r - l + 2; d1[r + 2] += r - l + 1;
}
void add2(int l,int r){
    if (l > r) return;
    d2[l] += 1; d2[r + 1] -= r - l + 2; d2[r + 2] += r - l + 1;
}
void Add1(int l,int r){
    if (l > r) return;
    D1[r] += 1; D1[l - 1] -= r - l + 2; D1[l - 2] += r - l + 1;
}
void Add2(int l,int r){
    if (l > r) return;
    D2[r] += 1; D2[l - 1] -= r - l + 2; D2[l - 2] += r - l + 1;
}
void solve(){
    for (int i = 1; i < maxn; i++) d1[i] += d1[i - 1],d2[i] += d2[i - 1];
    for (int i = 1; i < maxn; i++) d1[i] += d1[i - 1],d2[i] += d2[i - 1];
    for (int i = maxn - 2; i; i--) D1[i] += D1[i + 1],D2[i] += D2[i + 1];
    for (int i = maxn - 2; i; i--) D1[i] += D1[i + 1],D2[i] += D2[i + 1];
    for (int i = 1; 2 * i - 1 <= n; i++) f[2 * i - 1] += d1[i] + D1[i];
    for (int i = 1; 2 * i <= n; i++) f[2 * i] += d2[i] + D2[i];
    //REP(i,100) printf("%lld ",f[i]); puts("");
    for (int i = 1; i <= n; i++) f[i] += f[i - 1];
```

```
}
void work(){
    for (int m = 3; m <= n; m++){
        int l = m + 3,mid = 2 * m,r = m * 3 - 3;
        while (l <= n){
            add1((l + 2) / 2,(mid + 1) / 2);
            add2((l + 1) / 2,mid / 2);
            Add1((mid + 3) / 2,(r + 1) / 2);
            Add2((mid + 2) / 2,r / 2);
            l += m; mid += m; r += m;
        }
    }
    solve();
}
int main(){
    work();
    int T = read(),l,r;
    for (int i = 1; i <= T; i++){
        l = read(); r = read();
        printf("Case #%d: %lld\n",i,f[r] - f[l - 1]);
    }
    return 0;
}
```

G.Game SET

题意

给你n个四元组，每个四元组的不同位置分别有三个不同属性，现在问你能不能找出三个四元组，使得他们的四个位置中对于每一个位置都有：三者属性完全不同或者完全相同。

题解

坚信能很快搜到答案□$O(n^3*T)$暴力直接莽

```
#include<bits/stdc++.h>
using namespace std;
#define mem(a,b) memset(a,b,sizeof(a))
typedef long long LL;
typedef pair<int,int> PII;
#define X first
#define Y second
inline int read()
{
    int x=0,f=1;char c=getchar();
    while(!isdigit(c)){if(c=='-')f=-1;c=getchar();}
```

```
    while(isdigit(c)){x=x*10+c-'0';c=getchar();}
    return x*f;
}
const int maxn=300;
int T,n,has[maxn][4][3],List[5];
string a[maxn];
bool ok,vis[maxn];
bool judge(int A,int B,int C)
{
    for(int i=1;i<4;i++)
    {
        int cnt[4];
        mem(cnt,0);
        for(int j=0;j<3;j++)
        {
            if(has[A][i][j]==1)cnt[j]++;
            if(has[B][i][j]==1)cnt[j]++;
            if(has[C][i][j]==1)cnt[j]++;
        }
        if(cnt[0]>=1 && cnt[1]>=1 && cnt[2]>=1)continue;
        else if(cnt[0]>=3 || cnt[1]>=3 || cnt[2]>=3)continue;
        else return 0;
    }
    return 1;
}
void dfs1(int step)
{
    if(ok)return;
    if(step==4)
    {
        if(judge(List[1],List[2],List[3]))ok=1;
        return;
    }
    for(int i=1;i<=n && !ok;i++)
    {
        if(has[i][0][step-1] && !vis[i])
        {
            List[step]=i;
            vis[i]=1;
            dfs1(step+1);
            vis[i]=0;
        }
    }
}
void dfs2(int step,int color)
{
    if(ok)return;
    if(step==4)
    {
```

```
        if(judge(List[1],List[2],List[3]))ok=1;
        return;
    }
    for(int i=step;i<=n && !ok;i++)
    {
        if(has[i][0][color] && !vis[i])
        {
            List[step]=i;
            vis[i]=1;
            dfs2(step+1,color);
            vis[i]=0;
        }
    }
}
int main()
{
    T=read();
    for(int Case=1;Case<=T;Case++)
    {
        ok=0;
        n=read();
        for(int i=1;i<=n;i++)
            for(int j=0;j<4;j++)
                for(int k=0;k<3;k++)has[i][j][k]=0;
        for(int i=1;i<=n;i++)
        {
            cin>>a[i];
            int cnt=-1;
            for(int j=0;j<a[i].length();j++)
            {
                if(a[i][j]=='[')
                {
                    ++cnt;
                    if(cnt==0 &&
a[i][j+1]=='*')has[i][0][0]=has[i][0][1]=has[i][0][2]=1;
                    else if(cnt==0 && a[i][j+2]=='h')has[i][0][2]=1;
                    else if(cnt==0 && a[i][j+2]=='w')has[i][0][1]=1;
                    else if(cnt==0)has[i][0][0]=1;
                    else if(cnt==1 &&
a[i][j+1]=='*')has[i][1][0]=has[i][1][1]=has[i][1][2]=1;
                    else if(cnt==1 && a[i][j+1]=='d')has[i][1][0]=1;
                    else if(cnt==1 && a[i][j+1]=='s')has[i][1][1]=1;
                    else if(cnt==1 && a[i][j+1]=='o')has[i][1][2]=1;
                    else if(cnt==2 &&
a[i][j+1]=='*')has[i][2][0]=has[i][2][1]=has[i][2][2]=1;
                    else if(cnt==2 && a[i][j+2]=='o')has[i][2][0]=1;
                    else if(cnt==2 && a[i][j+2]=='t')has[i][2][1]=1;
                    else if(cnt==2)has[i][2][2]=1;
                    else if(cnt==3 &&
a[i][j+1]=='*')has[i][3][0]=has[i][3][1]=has[i][3][2]=1;
                    else if(cnt==3 && a[i][j+1]=='r')has[i][3][0]=1;
```

```
                    else if(cnt==3 && a[i][j+1]=='g')has[i][3][1]=1;
                    else if(cnt==3 && a[i][j+1]=='p')has[i][3][2]=1;
                }
            }
        }
        dfs1(1);
        if(ok){printf("Case #%d: %d %d
%d\n",Case,List[1],List[2],List[3]);continue;}
        for(int col=0;col<3 && !ok;col++)
        {
            dfs2(1,col);
            if(ok){printf("Case #%d: %d %d
%d\n",Case,List[1],List[2],List[3]);break;}
        }
        if(ok)continue;
        printf("Case #%d: -1\n",Case);
    }
    return 0;
}
```

I.Interesting Computer Game

题意

给了 n 对数,每对数可以选择其中的一个,最后问你最多能选多少个数

题解

当成一个图,每条边选择其中的一个点,问最多能选多少个点,我们发现度数为零的点不可能选,度数为一的点一定选,如果所有点度数都大于一,这些点能被选. 我们就可以维护每个点的度数,每次选一个度数最小的点,若等于一,就删掉这条边并维护度数,若大于一,就说明剩下的点都可选.

```
#include<iostream>
#include<cstdio>
#include<algorithm>
#include<set>
#include<cstring>
#define ll long long
using namespace std;
int read()
{
    int k=0,f=1;char c=getchar();
    for(;!isdigit(c);c=getchar()) if(c=='-') f=-1;
    for(;isdigit(c);c=getchar()) k=k*10+c-'0';return k*f;
}
```

```
const int N=200055,inf=0x3f3f3f3f;
int T,n,m,tot,a[N],b[N],c[N],d[N],mn[N<<2];
set<int> s[N];
#define lson k<<1,l,mid
#define rson k<<1|1,mid+1,r
void pu(int k)
{
    if(d[mn[k<<1]]<d[mn[k<<1|1]])
        mn[k]=mn[k<<1];
    else mn[k]=mn[k<<1|1];
}
void build(int k,int l,int r)
{
    if(l==r) {mn[k]=l;return;}
    int mid=l+r>>1;
    build(lson);build(rson);
    pu(k);
}
void update(int k,int l,int r,int a)
{
    if(l==r) return;
    int mid=l+r>>1;
    if(a<=mid) update(lson,a);
    else update(rson,a);
    pu(k);
}
int main()
{
    T=read();
    for(int p=1;p<=T;p++)
    {
        n=read();tot=0;int res=0;
        for(int i=1;i<=n;i++)
        {
            a[i]=read(),c[++tot]=a[i],b[i]=read(),c[++tot]=b[i];
        }
        sort(c+1,c+1+tot);
        tot=unique(c+1,c+1+tot)-c-1;
        for(int i=1;i<=n;i++)
        {
            a[i]=lower_bound(c+1,c+1+tot,a[i])-c;
            b[i]=lower_bound(c+1,c+1+tot,b[i])-c;
            if(a[i]==b[i]) d[a[i]]=inf;
            else
s[a[i]].insert(b[i]),s[b[i]].insert(a[i]),d[a[i]]++,d[b[i]]++;
        }
        build(1,1,tot);
        set<int>::iterator it;
        while(1)
        {
            if(d[mn[1]]==0)
```

```
            {
                res--;
                int x=mn[1];
                d[x]=inf;
                update(1,1,tot,x);
            }
            else if(d[mn[1]]==1)
            {
                int x=mn[1];
                it=s[x].begin();
                d[x]=inf;d[*it]--;
                update(1,1,tot,x);
                update(1,1,tot,*it);
                s[*it].erase(s[*it].lower_bound(x));
                s[x].erase(it);
            }
            else break;
        }
        for(int i=1;i<=tot;i++)
            s[i].clear(),d[i]=0;
        printf("Case #%d: %d\n",p,res+tot);
    }
    return 0;
}
```

K.Kabaleo Lite

题意

有$1到n$的菜，每个菜有a_i的价值（有正有负），有b_i份，客人来会食用会选取一个位置，然后食用包含这个位置的前缀的菜，问你最多能有多少个客人来，并在第一问答案的基础上问价值和最大能多少。

题解

第一个答案根据题意只能是b_i之后就选取后面位置中前缀和最大的地方，然后让尽量多的人吃这个，能吃的数量就是这段区间的最小值，用线段树维护即可。

注意计算答案时要开_int128我吐了

```
#include<bits/stdc++.h>
using namespace std;
#define mem(a,b) memset(a,b,sizeof(a))
typedef long long LL;
typedef pair<int,int> PII;
#define X first
```

```
#define Y second
inline int read()
{
    int x=0,f=1;char c=getchar();
    while(!isdigit(c)){if(c=='-')f=-1;c=getchar();}
    while(isdigit(c)){x=x*10+c-'0';c=getchar();}
    return x*f;
}
const int maxn=100010,inf=1e9;
int T,n,Mino[maxn<<2],Minv[maxn<<2],tag[maxn<<2];
LL a[maxn],b[maxn],pre[maxn],Maxpos[maxn];
void maintain(int L,int R,int o)
{
    int mid=L+R>>1,lo=o<<1,ro=lo|1;
    Mino[o]=min(Mino[lo],Mino[ro]);
    if(Mino[lo]==Mino[ro])Minv[o]=max(Minv[lo],Minv[ro]);
    else if(Mino[lo]<Mino[ro])Minv[o]=Minv[lo];
    else Minv[o]=Minv[ro];
}
void build(int L,int R,int o)
{
    if(L==R)
    {
        Mino[o]=b[L];
        Minv[o]=L;
        tag[o]=0;
        return;
    }
    int mid=L+R>>1,lo=o<<1,ro=lo|1;
    build(L,mid,lo);
    build(mid+1,R,ro);
    maintain(L,R,o);
    tag[o]=0;
}
void pushdown(int o)
{
    int lo=o<<1,ro=lo|1;
    Mino[lo]-=tag[o];Mino[ro]-=tag[o];
    tag[lo]+=tag[o];tag[ro]+=tag[o];
    tag[o]=0;
}
PII query(int L,int R,int o,int ql,int qr)
{
    if(L==ql && qr==R)
    {
        return make_pair(Minv[o],Mino[o]);
    }
    pushdown(o);
    int mid=L+R>>1,lo=o<<1,ro=lo|1;
    PII res;
    if(qr<=mid)res=query(L,mid,lo,ql,qr);
```

```
    else if(ql>mid)res=query(mid+1,R,ro,ql,qr);
    else
    {
        PII ans1=query(L,mid,lo,ql,mid),ans2=query(mid+1,R,ro,mid+1,qr);
        if(ans1.Y<ans2.Y)res=ans1;
        else if(ans1.Y>ans2.Y)res=ans2;
        else res=make_pair(max(ans1.X,ans2.X),ans1.Y);
    }
    maintain(L,R,o);
    return res;
}
void update(int L,int R,int o,int ql,int qr,int plus)
{
    if(L==ql && R==qr)
    {
        Mino[o]=Mino[o]-plus;
        tag[o]+=plus;
        return;
    }
    pushdown(o);
    int mid=L+R>>1,lo=o<<1,ro=lo|1;
    if(qr<=mid)update(L,mid,lo,ql,qr,plus);
    else if(ql>mid)update(mid+1,R,ro,ql,qr,plus);
    else update(L,mid,lo,ql,mid,plus),update(mid+1,R,ro,mid+1,qr,plus);
    maintain(L,R,o);
    return;
}

void print(__int128 x)
{
    if (!x) return ;
    if (x < 0) putchar('-'),x = -x;
    print(x / 10);
    putchar(x % 10 + '0');
}

int main()
{
    T=read();
    for(int Case=1;Case<=T;Case++)
    {
        n=read();
        for(int i=1;i<=n;i++)a[i]=read(),pre[i]=pre[i-1]+a[i];
        for(int i=1;i<=n;i++)b[i]=read();
        build(1,n,1);
        printf("Case #%d: %lld ",Case,b[1]);
        __int128 ans=(__int128)b[1]*(__int128)a[1];

        Maxpos[1]=1;
```

```
        for(int i=2;i<=n;i++)
        {
            if(pre[i]>pre[Maxpos[i-1]])Maxpos[i]=i;
            else Maxpos[i]=Maxpos[i-1];
        }
        int r=Maxpos[n],cnt=0;
        while(r>=2 && cnt<=b[1])
        {
            PII tmp=query(1,n,1,2,r);
            if(tmp.Y<=0)
            {
                r=Maxpos[tmp.X-1];
                continue;
            }
            ans+=(__int128)tmp.Y*(__int128)(pre[r]-pre[1]);
            cnt+=tmp.Y;
            if(cnt>b[1]){
                ans-=(__int128)(cnt-b[1])*(__int128)(pre[r]-pre[1]);
                break;
            }
            update(1,n,1,2,r,tmp.Y);
            r=Maxpos[tmp.X-1];
        }
        print(ans);
        puts("");
    }
    return 0;
}
```

训练实况

开局wxg看**K**，想出**K**的做法后fyh写**K**

wxg和hxm讨论**I** wxg写**I** 过**I**

fyh狂wa**K** wxg帮忙调**K** hxm想**E**并开写

2:20 发现要开int128 fyh过**K**，fyh开写**G**

3:12 hxm过**E**

3:25fyh过**G**

训练总结

wxg: 只能说尽力了,罚时日常垫底

hxm:这把能做的都做了，但就是罚时还差一点

fyh:罚时怪我，缺一个各种功能都有的线段树板子。