

[比赛链接](#)

题解

G. Game of Swapping Numbers

题意

给定两个长度为 n 的序列 A, B 和 k 次操作，每次操作可以交换 a_i, a_j

要求在 k 次操作后最小化 $\sum_{i=1}^n |a_i - b_i|$

题解

首先考虑不存在 k 约束的情况。

可以将本题转化为从 a_i, b_i 中选定 n 个数在前面加上 $+$ 号，其余 n 个数在前面加上 $-$ 号。最后需要最大化所有带符号数的和。

显然贪心选取 a_i, b_i 这 $2n$ 个数中前 n 大加上 $+$ 号即可。然后为了保证合法性，需要使得 a_i, b_i 正好一正一负。

于是考虑任选一组 $(+a_i, +b_i), (-a_j, -b_j)$ 交换 a_i, a_j 直到不存在这种情况为止即可。

接下来考虑 k 有限的情况，显然贪心每次选取收益最大的 $(+a_i, +b_i), (-a_j, -b_j)$ 交换即可。

此时收益为 $2(\min(a_i, b_i) - \max(a_j, b_j))$ 于是考虑将 $\min(a_i, b_i)$ 序列从大到小排序 $\max(a_i, b_i)$ 从小到大排序。

然后贪心取前 k 个收益即可。注意 $(+a_i, +b_i)$ 在 $\min(a_i, b_i)$ 序列中一定排在 $(+a_i, -b_i), (-a_i, +b_i), (-a_i, -b_i)$ 的前面。

注意 $(-a_i, -b_i)$ 在 $\max(a_i, b_i)$ 序列中一定排在 $(+a_i, -b_i), (-a_i, +b_i), (+a_i, +b_i)$ 的前面。

于是一定会先让所有 $(+a_i, +b_i)$ 和 $(-a_j, -b_j)$ 配对，至于其他的配对方案计算出来的 $2(\min(a_i, b_i) - \max(a_j, b_j))$ 都是非正数，可以舍去。

注意到 k 可能大于需要交换的次数，但此时如果 $n > 2$ 一定可以找到两个同号的 a_i, a_j 做无意义的交换消耗 k 使得 k 正好等于需要交换的次数。

最后 $n=2$ 的情况没有选择只能强制交换，所以不一定可以得到最优解，需要特判。总时间复杂度 $O(n \log n)$

```
const int MAXN=5e5+5;
int a[MAXN],b[MAXN];
int main()
{
    int n=read_int(),k=read_int();
```

```
_for(i,0,n)a[i]=read_int();
_for(i,0,n)b[i]=read_int();
if(n==2){
    if(k&1)swap(a[0],a[1]);
    enter(abs(a[0]-b[0])+abs(a[1]-b[1]));
    return 0;
}
LL ans=0;
_for(i,0,n){
    if(a[i]>b[i])
        swap(a[i],b[i]);
    ans+=b[i]-a[i];
}
sort(a,a+n,greater<int>());
sort(b,b+n);
_for(i,0,min(n,k))
    ans+=max(0,a[i]-b[i])*2;
enter(ans);
return 0;
}
```

J. Journey among Railway Stations

题意

给定 n 个车站，车站 i 开放时间为 $[u_i, v_i]$ 从车站 i 到车站 $i+1$ 需要花费 c_i 时间。接下来 q 次操作：

1. 询问是否可以从 x 车站出发达到 y 车站，初始时为 0 时刻
2. 修改某个 c_i
3. 修改某个 $[u_i, v_i]$

注意只能在车站开放时间内进入车站，且车站 i 只能直接到达车站 $i+1$

题解

定义时间函数 $f(l, r, t)$ 表示起始时间为 t 时从车站 l 到车站 $r+1$ 需要花费的时间。不难发现

$$f(i, i, t) = \begin{cases} c_i, & 0 \leq t \leq u_i \\ c_i + (t - u_i), & u_i < t \leq v_i \end{cases}$$

同时对任意 $i \leq j \leq k$ 有 $f(i, k, t) = f(j+1, k, f(i, j, t))$ 利用数学归纳法可以证明 $f(i, j, t)$ 总是形如以下形式：

$$f(i, j, t) = \begin{cases} y, & 0 \leq t \leq x_1 \\ y + (t - x_1), & x_1 < t \leq x_2 \end{cases}$$

于是考虑线段树维护区间 $[l, r]$ 的时间函数 $f(l, r, t)$ 顺便维护区间合法性即可，时间复杂度 $O(n \log n)$

```

const int MAXN=1e6+5;
int u[MAXN],v[MAXN],c[MAXN];
struct Node{
    bool legal;
    LL x1,x2,y;
}s[MAXN<<2];
Node add(const Node &a,const Node &b){
    Node c;
    c.legal=a.legal&& b.legal;
    if(c.legal){
        if(a.y>b.x2)
            c.legal=false;
        else if(a.y>=b.x1){
            c.x1=a.x1;
            c.x2=min(a.x2,c.x1+b.x2-a.y);
            c.y=b.y+a.y-b.x1;
        }
        else{
            c.x1=min(a.x2,a.x1+b.x1-a.y);
            c.x2=min(a.x2,c.x1+b.x2-b.x1);
            c.y=b.y;
        }
    }
    return c;
}
int lef[MAXN<<2],rig[MAXN<<2];
void push_up(int k){
    s[k]=add(s[k<<1],s[k<<1|1]);
}
void build(int k,int pos){
    s[k].legal=true;
    s[k].x1=u[pos];
    s[k].x2=v[pos];
    s[k].y=u[pos]+c[pos];
}
void build(int k,int L,int R){
    lef[k]=L,rig[k]=R;
    int M=L+R>>1;
    if(L==R){
        build(k,M);
        return;
    }
    build(k<<1,L,M);
    build(k<<1|1,M+1,R);
    push_up(k);
}
void update(int k,int pos){
    if(lef[k]==rig[k]){
        build(k,pos);
        return;
    }
}

```

```
int mid=lef[k]+rig[k]>>1;
if(mid>=pos)
update(k<<1,pos);
else
update(k<<1|1,pos);
push_up(k);
}
Node query(int k,int L,int R){
if(L<=lef[k]&&rig[k]<=R)
return s[k];
int mid=lef[k]+rig[k]>>1;
if(mid>=R)
return query(k<<1,L,R);
else if(mid<L)
return query(k<<1|1,L,R);
else
return add(query(k<<1,L,R),query(k<<1|1,L,R));
}
int main()
{
int T=read_int();
while(T--){
int n=read_int();
_rep(i,1,n)u[i]=read_int();
_rep(i,1,n)v[i]=read_int();
_for(i,1,n)c[i]=read_int();
build(1,1,n);
int q=read_int();
while(q--){
int opt=read_int();
if(opt==0){
int l=read_int(),r=read_int();
Node ans=query(1,l,r);
if(ans.legal)
puts("Yes");
else
puts("No");
}
else if(opt==1){
int pos=read_int();
c[pos]=read_int();
update(1,pos);
}
else{
int pos=read_int();
u[pos]=read_int();
v[pos]=read_int();
update(1,pos);
}
}
}
```

```
}  
return 0;  
}
```

赛后总结

jxm 玄学场，打表、随机化yyds

From: <https://wiki.cvbbacm.com/> - CVBB ACM Team
Permanent link: https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:legal_string:%E7%BB%84%E9%98%9F%E8%AE%AD%E7%BB%83%E6%AF%94%E8%B5%9B%E8%AE%B0%E5%BD%95:contest3&rev=1626578448
Last update: 2021/07/18 11:20