2020牛客暑期多校训练营（第二场）

比赛情况

题号	A	B	C	D	E	F	G	H	I	J	K
状态	-	Ø	O	O	-	O	Ø	-	-	Ø	!

O 在比赛中通过 Ø 赛后通过 ! 尝试了但是失败了 - 没有尝试

比赛时间

2020-07-13 12:00-17:00

题解

B - Boundary

问过原点的圆周里过给定点数量的最大值。

$n\le 2000$想到选各点作为圆上一点P则圆心在OP的中垂线上，对于中垂线两两求交点看重合次数map查询$O(n^2\log n)$

```
#pragma GCC optimize(2)
#pragma GCC optimize(3,"Ofast","inline")
#include<bits/stdc++.h>
#define ll long long
using namespace std;
const int N=2e3+10;
const double eps=1e-5;
const double INF=(1LL<<30);
inline ll gcd(ll a,ll b){return b?gcd(b,a%b):a;}
inline ll lcm(ll a,ll b){return a*b/gcd(a,b);}
int dcmp(double x,double y)
{
    double t=fabs(x-y);
    if(t<-eps)return -1;
    return t>eps;
}
struct Point
{
    double x,y;
    Point(double x=0,double y=0):x(x),y(y){}
    bool operator < (const Point &a) const {return
dcmp(x,a.x)==0?(y<a.y):(x<a.x);}
    bool operator == (const Point &a) const {return
dcmp(x,a.x)==0&&dcmp(y,a.y)==0;}
}p[N];
```

```
struct Line
{
    double a,b,c;
    Line(double a=0,double b=0,double c=0):a(a),b(b),c(c){}
}l[N];
inline Point middle(Point p1,Point p2){return
Point((p1.x+p2.x)*0.5,(p1.y+p2.y)*0.5);}
Point ori=Point(0,0);
inline Point intersection(Line l1,Line l2)
{
    if(dcmp(l1.b*l2.a,l2.b*l1.a)==0)return ori;
    double y=(l2.c*l1.a-l1.c*l2.a)/(l1.b*l2.a-l2.b*l1.a);
    double x=(l1.c*l2.b-l2.c*l1.b)/(l1.b*l2.a-l2.b*l1.a);
    return Point(x,y);
}
map<Point,int>mp;
inline ll read()
{
    ll x=0,f=1;char c=getchar();
    while(c<'0'||c>'9'){if(c=='-')f=-1;c=getchar();}
    while(c>='0'&&c<='9'){x=x*10+c-'0';c=getchar();}
    return x*f;
}
int main()
{
    int n=read(),res=0;
    for(int i=1;i<=n;i++)
    {
        p[i].x=read(),p[i].y=read();
        Point mid=middle(p[i],ori);
        l[i].a=p[i].x,l[i].b=p[i].y;
        l[i].c=-l[i].a*mid.x-l[i].b*mid.y;
        mp.clear();
        for(int j=1;j<i;j++)
        {
            Point inter=intersection(l[i],l[j]);
            if(inter==ori)continue;
            res=max(res,++mp[inter]);
        }
    }
    printf("%d\n",res+1);
    return 0;
}
```

D - Duration

签到题，把两个时间都换算成到 $0:0:0$ 的秒数，相减取绝对值。

```
#include<bits/stdc++.h>
#define ALL(x) (x).begin(),(x).end()
#define ll long long
#define ull unsigned long long
#define pii_ pair<int,int>
#define mp_ make_pair
#define pb push_back
#define fi first
#define se second
#define rep(i,a,b) for(int i=(a);i<=(b);i++)
#define per(i,a,b) for(int i=(a);i>=(b);i--)
#define show1(a) cout<<#a<<" = "<<a<<endl
#define show2(a,b) cout<<#a<<" = "<<a<<"; "<<#b<<" = "<<b<<endl
using namespace std;
const ll INF = 1LL<<60;
const int inf = 1<<30;
const int maxn = 2e5+5;
inline void fastio() {ios::sync_with_stdio(false);cin.tie(0);cout.tie(0);}



int main()
{
    //fastio();
    int a,b,c,x,y,z;
    scanf("%d:%d:%d",&a,&b,&c);
    scanf("%d:%d:%d",&x,&y,&z);
    ll t = a*3600 + b*60 + c;
    ll p = x*3600 + y*60 +z;
    ll ans = abs(t-p);
    printf("%lld\n",ans);
    return 0;
}
```

F - Fake Maxpooling

$n\times m$矩阵$a[i][j]=\gcd(i,j)$求所有$k\times k$子矩阵最大值的和

先对某一维单调队列（对于每行k列最大值缩为一格），之后另一维也单调队列□□$O(nm)$□gcd这个暴力要多个$\log n$□有点慢，但卡过去了，其实是应该记忆化

```
#pragma GCC optimize(2)
#pragma GCC optimize(3,"Ofast","inline")
#include<bits/stdc++.h>
#define ll long long
using namespace std;
```

```
const int N=5e3+10;
int a[N][N],b[N][N];
int head,tail,q[N*10];
inline int gcd(int a,int b){return b?gcd(b,a%b):a;}
inline int lcm(int a,int b){return a*b/gcd(a,b);}
int main()
{
    int n,m,k;
    scanf("%d%d%d",&n,&m,&k);
    ll res=0;
    for(int i=1;i<=n;i++)
    {
        head=1,tail=0;
        for(int j=1;j<=m;j++)
        {
            a[i][j]=lcm(i,j);
            while(tail>=head&&j-q[head]>=k)head++;
            while(tail>=head&&a[i][j]>a[i][q[tail]])tail--;
            q[++tail]=j;
            if(j>=k)b[i][j-k+1]=a[i][q[head]];
        }
    }
    for(int j=1;j<=m-k+1;j++)
    {
        head=1,tail=0;
        for(int i=1;i<=n;i++)
        {
            while(tail>=head&&i-q[head]>=k)head++;
            while(tail>=head&&b[i][j]>b[q[tail]][j])tail--;
            q[++tail]=i;
            if(i>=k)res+=b[q[head]][j];
        }
    }
    printf("%lld\n",res);
    return 0;
}
```

G - Greater and Greater

给两个序列 A 和 B，求 A 的所有长度为 $|B|$ 的子串中，每一位都对应大于 B 的每一位的个数。

用 bitset 压位，先预处理出 S ，其中 $S[i][j]=1$ 表示 $A[i]>=B[j]$，并维护 A 的长度为 $|B|$ 的子串 a 的前缀与 B 的后缀的大小关系，考虑从后往前 dp，具体来说就是，$cur[i]=1$ 表示 a 的 $m-i$ 前缀每一位都对应大于等于 B 的 $m-i$ 后缀，从后往前滚动，每次向左移一位，把 $m-1$ 置为 1，再与 $S[i]$ 按位与，统计 $cur[0]=1$ 的个数就是答案。

```cpp
#include<bits/stdc++.h>
#define ALL(x) (x).begin(),(x).end()
#define ll long long
#define ull unsigned long long
#define pii_ pair<int,int>
#define mp_ make_pair
#define pb push_back
#define fi first
#define se second
#define rep(i,a,b) for(int i=(a);i<=(b);i++)
#define per(i,a,b) for(int i=(a);i>=(b);i--)
#define show1(a) cout<<#a<<" = "<<a<<endl
#define show2(a,b) cout<<#a<<" = "<<a<<"; "<<#b<<" = "<<b<<endl
using namespace std;
const ll INF = 1LL<<60;
const int inf = 1<<30;
const int maxn = 2e5+5;
inline void fastio() {ios::sync_with_stdio(false);cin.tie(0);cout.tie(0);}

int a[150005],b[40005],c[40005],id[40005];
int n,m;
bitset<40000> S[40001],cur;

int main()
{
    fastio();
    cin>>n>>m;
    rep(i,0,n-1) cin>>a[i];
    rep(i,0,m-1) {cin>>b[i];c[i] = b[i];id[i]=i;}
    sort(c,c+m);
    sort(id,id+m,[](int x,int y){return b[x]<b[y];});
    rep(i,0,m-1){
        if(i>0) S[i] |= S[i-1];
        S[i][id[i]] = 1;
    }
    //rep(i,0,m-1) {rep(j,0,m-1) cout<<S[i][j]<<" ";cout<<endl;}
    int ans = 0;
    rep(i,0,m-1){
        int flag = 1;
        rep(j,0,m-i-1){
            if(a[n-m+j]<b[j+i]) {flag=0;break;}
        }
        cur[i] = flag;
    }
    ans += cur[0];
    per(i,n-m-1,0){ //show1(a[i]);
        int pos = upper_bound(c,c+m,a[i]) - c - 1; //show1(pos);
        cur >>= 1;
        cur[m-1] = 1;
        if(pos==-1) cur &= S[m];
        else cur &= S[pos];
```

```
        ans += cur[0];
    }
    cout<<ans<<endl;
    return 0;
}
```

K - Keyboard Free

三个半径分别为r_1,r_2,r_3的同心圆上分别取一点构成三角形，求面积期望，保留一位小数。（懂了，保留一位小数就是增量乱搞qaq

暴力积分有点难，涉及到负面积之类的，所以掺杂一点乱搞成分比较好。第一维位置固定，第二维增量乱搞，第三维easy积分。

第一维取点G第二维在$[0,\pi]$间均匀枚举一定量的角度得E第三位积分算圆周到EG直线的距离即可得三角形面积。

```
#pragma GCC optimize(2)
#pragma GCC optimize(3,"Ofast","inline")
#include<bits/stdc++.h>
using namespace std;
const double pi=acos(-1.0);
double r1,r2,r3;
double sqr(double x){return x*x;}
double f(double j)
{
    double EG=sqrt(sqr(r1)+sqr(r2)-2*cos(j)*r1*r2);
    double AGE=acos((sqr(r1)+sqr(EG)-sqr(r2))/2/r1/EG);
    double alp=asin(r1*sin(AGE)/r3);
    return r3*(alp*sin(alp)+cos(alp))*EG/pi;
}
int main()
{
    int t;
    scanf("%d",&t);
    while(t--)
    {
        scanf("%lf%lf%lf",&r1,&r2,&r3);
        if(r1>r2)swap(r1,r2);
        if(r2>r3)swap(r2,r3);
        if(r1>r2)swap(r1,r2);
        double res=0,j=pi/10000.0;
        for(int i=0;i<5000;i++,j+=pi/5000.0)res+=f(j);
        printf("%.1lf\n",res/5000);
    }
```

```
    return 0;
}
```

比赛总结与反思