半平面交

给定n个形如 $ax+by+c\le0$ 的半平面，找到所有满足它们的点所组成的额点集，称为半平面交。

相交后的区域可能是直线、射线、线段或者点，甚至也有可能是空集。

求解算法 排序增量法

把半平面分成两部分，一部分是极角范围内的 $(-\frac{\pi}{2},\frac{\pi}{2})$ 另一部分是极角范围外的角度。

考虑 $(-\frac{\pi}{2},\frac{\pi}{2})$ 的半平面，将他们按照极角排序。极角相同的半平面根据常数项保留一个。

然后从排序后极角最小的两个半平面开始，求出他们的交点并将它们压入一个栈，每次按照极角从小到大的顺序增加一个半平面，算出他和栈顶半平面的交点。如果当前的交点在栈顶两个半平面交点的右边，则让它出栈。

例题

洛谷

大致题意

求 n 个多边形的交

把每个多边形表示成半平面交的形式，然后只需计算所有的半平面交。

代码

```
#include<bits/stdc++.h>
using namespace std;
const double eps=1e-13;
struct point
{
    double x,y;
    point operator -(point &s)
    {
        return (point){x-s.x,y-s.y};
    }
};
double operator *(point a,point b)
{
    return a.x*b.y-a.y*b.x;
}
struct line
{
    double d;
```

```
    point a,b;
}l[1005];
bool cmpd(line a,line b)
{
    return a.d<b.d;
}
bool bian(point Q,point P1,point P2)
{
    return fabs((Q-P1)*(P2-P1)<eps&&min(P1.x,P2.x)-eps<=Q.x&&Q.x-
eps<=max(P1.x,P2.x)&&min(P1.y,P2.y)-eps<=Q.y&&Q.y-eps<=max(P1.y,P2.y));
}
point crosp(line a,line b)
{
    double s1=(b.a-a.a)*(a.b-a.a),s2=(a.b-a.a)*(b.b-a.a);
    return (point){(b.a.x*s2+s1*b.b.x)/(s1+s2),(b.a.y*s2+s1*b.b.y)/(s1+s2)};
}
int n,m,sta[2005];
int main()
{
    cin>>n;
    for (int i=1;i<=n;i++)
    {
        int mm;
        cin>>mm;
        for (int j=1;j<=mm;j++)
        {
            m++,cin>>l[m].a.x>>l[m].a.y;
            l[(j==1)?m+mm-1:m-1].b=l[m].a;
        }
    }
    n=m;
    for (int i=1;i<=n;i++)
        l[i].d=atan2(l[i].b.y-l[i].a.y,l[i].b.x-l[i].a.x);
    sort(&l[1],&l[n+1],cmpd);
    for (int i=1;i<=n;i++)
    {
        for (;sta[0]>=1;sta[0]--)
        {
            if (fabs(l[i].d-l[sta[sta[0]]].d)<eps)
            {
                if ((l[sta[sta[0]]].b-l[sta[sta[0]]].a)*(l[i].a-
l[sta[sta[0]]].a)<eps) break;
            }
            else break;
        }
        for (;sta[0]>=2;sta[0]--)
        {
            if ((l[i].b-l[i].a)*(crosp(l[sta[sta[0]]],l[sta[sta[0]-1]])-
l[i].a)>eps) break;
        }
```

```
        if (fabs(l[i].d-l[sta[sta[0]]].d)>=eps) sta[++sta[0]]=i;
    }
    int L=1,R=sta[0];
    while (L<R)
    {
        if ((l[sta[L]].b-l[sta[L]].a)*(crosp(l[sta[R]],l[sta[R-1]])-
l[sta[L]].a)<eps) R--;
        else
        {
            if ((l[sta[R]].b-l[sta[R]].a)*(crosp(l[sta[L]],l[sta[L+1]])-
l[sta[R]].a)<eps) L++;
            else break;
        }
    }
    if (R-L<=1)
    {
        printf("0.000\n");
        return 0;
    }
    double ans=0;
    sta[R+1]=sta[L],sta[R+2]=sta[L+1];
    for (int i=L;i<=R;i++)
        ans+=crosp(l[sta[i]],l[sta[i+1]])*crosp(l[sta[i+1]],l[sta[i+2]])/2;
    printf("%.3lf\n",ans);
    return 0;
}
```