计算几何初步-凸包Graham扫描法

概要知识

向量

> 在数学中，向量（也称为欧几里得向量、几何向量、矢量），指具有大小（magnitude）和方向的量。它可以形象化地表示为带箭头的线段。箭头所指：代表向量的方向；线段长度：代表向量的大小。与向量对应的只有大小，没有方向的量叫做数量（物理学中称标量）。

叉积

> 向量积，数学中又称外积、叉积，物理中称矢积、叉乘，是一种在向量空间中向量的二元运算。与点积不同，它的运算结果是一个向量而不是一个标量。并且两个向量的叉积与这两个向量和垂直。其应用也十分广泛，通常应用于物理学光学和计算机图形学中。

定义：向量积可以被定义为：$a \times b = ab * sin\theta$

模长：（在这里θ表示两向量之间的夹角（共起点的前提下）（$0^o \leq \theta \leq 180^o$），它位于这两个矢量所定义的平面上。$|\vec{a} \times \vec{b}| = |\vec{a}| \cdot |\vec{b}| \cdot sin\theta$

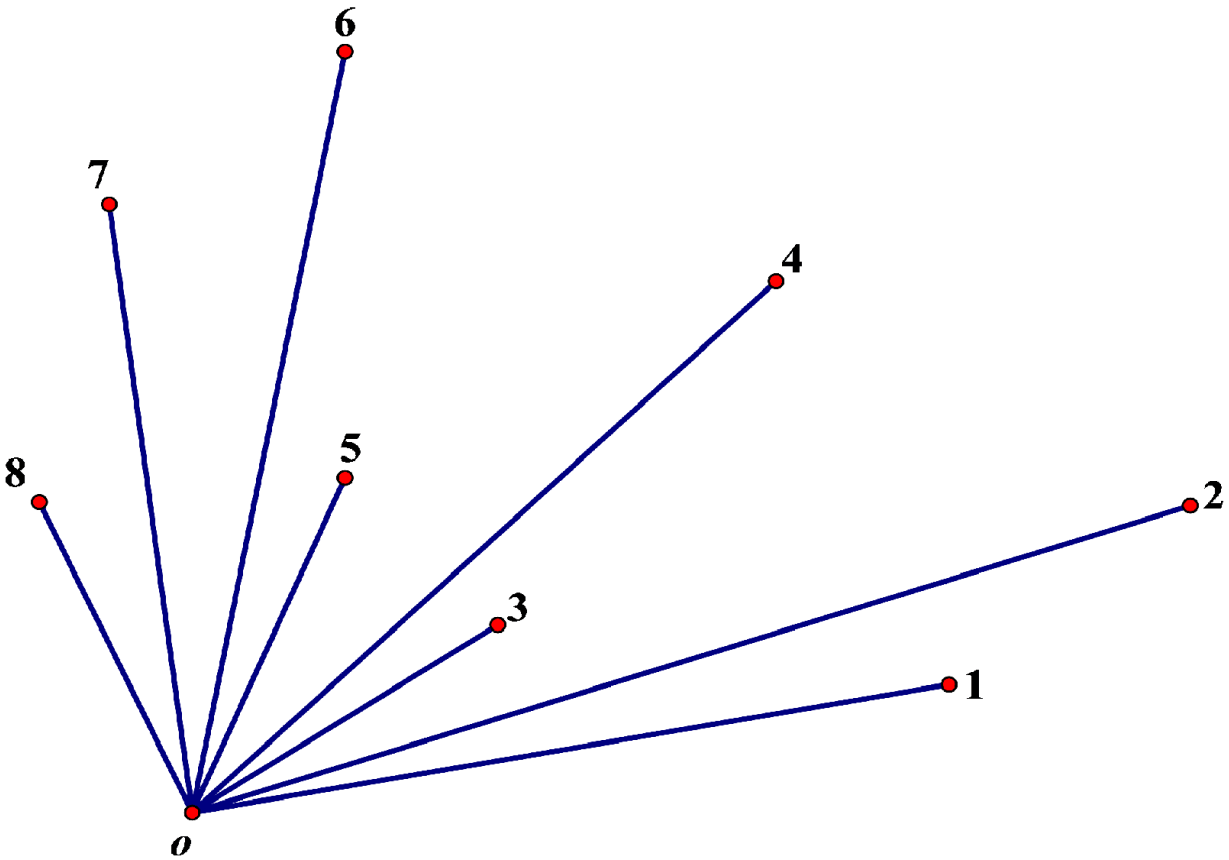
方向：a向量与b向量的向量积的方向与这两个向量所在的平面垂直，且遵守右手定则。也可以这样定义（等效）：向量积$|c| = |a||b|sin < a, b >$，即c的长度在数值上等于以a，b，夹角为θ组成的平行四边形面积。

*运算结果c是一个伪向量。这是因为c在不同的坐标系中可能不同。

算法流程

1.首先找出在平面直角坐标系中y坐标最小的点（一样就按x坐标排）。

2.将所有点按你找到的点（之后称其为o）o与它们所构成的极角排序。如图：



3.我们已知凸包上两个点o和1，我们将它们放入栈中，然后从3开始枚举。

4.将当前点与栈顶两个点比较，判断当前点是否在栈顶两点的左侧。（设栈顶为Top）

这可以用叉积判断，即判断$Top-1$与当前点构成的向量和$Top-1$与Top构成的向量的向量积是否为正。

5.若为负，则不停弹出栈顶，当然保证$Top>1$。

6.枚举完成，栈内的点即为凸包上的点。

例题传送门

```
#include <cmath>
#include <cstdio>
#include <algorithm>
using namespace std;

const int Maxn=10005;
struct Node{double x,y;}A[Maxn],St[Maxn];
int N,Cnt;
double Multi(Node x,Node y,Node z){
    double dx,dy,fx,fy;
    dx=x.x-z.x,dy=x.y-z.y;
    fx=y.x-z.x,fy=y.y-z.y;
    return dx*fy-dy*fx;
}
double Euild(Node x,Node y){
    return sqrt((x.x-y.x)*(x.x-y.x)+(x.y-y.y)*(x.y-y.y));
```

```
}
int Cmp(Node x,Node y){
    double Dis=Multi(x,y,A[1]);
    if(Dis<0)return 0;
    if(!Dis && Euild(x,A[1])>Euild(y,A[1]))return 0;
    return 1;
}

int main()
{
    scanf("%d",&N);
    for(int i=1;i<=N;i++){
        scanf("%lf%lf",&A[i].x,&A[i].y);
        if(A[i].y<A[1].y || A[i].y==A[1].y&&A[i].x<A[1].x){
            Node t=A[1];A[1]=A[i],A[i]=t;
        }
    }
    sort(A+2,A+N+1,Cmp);
    St[++Cnt]=A[Cnt],St[++Cnt]=A[Cnt];
    for(int i=3;i<=N;i++){
        while(Cnt>1&&Multi(St[Cnt],A[i],St[Cnt-1])<=0)Cnt--;
        St[++Cnt]=A[i];
    }
    double Ans=0;
    for(int i=1;i<Cnt;i++)
        Ans+=Euild(St[i],St[i+1]);
    Ans+=Euild(St[1],St[Cnt]);
    printf("%.2lf",Ans);
    return 0;
}
```