题意

二维平面内，给定n个点的坐标， 求一个半径最小的圆，使得所有点都在这个圆的圆上及圆内，求这个圆的圆心坐标和半径

分析

对于给定的点集A，记 Mincircle(A) 为点集A的最小外接圆，显然，对于所有的点集情况A，MinCircle (A) 都是存在且唯一的。

特别地，当A为空集时，Mincircle(A)为空集；当A = {a} 时 ， Mincircle （A) 圆心坐标为a，半径为0；

Mincircle(A) 可以由A边界上最多三个点确定（当点集A中点的个数大于1时，有可能两个点确定了Mincircle(A))。

也就是说，存在一个点集 B , |B| ≤ 3 且 B包含于A ， 有Mincircle （B） = Mincircle(A。 所以，如果a不属于B，则Mincircle (A - {a}) = Mincircle(A)。

如果Mincircle(A - {a}) 不等于Mincircle(A) ,则a 属于B 。

因此，可以从一个空集 R开始，不断把题目给定的点集加入R，同时维护R的外接圆最小。

最终求得最小外接圆

代码

```
#include<bits/stdc++.h>
using namespace std;
typedef  long long LL;
typedef  unsigned long long ULL;
typedef  long double LDB;
const int maxn = 1000005;
const double eps = 1e-6;
struct Point{
    double x,y;
```

```
10      Point operator - (const Point &a) const{
11          Point p1;
12          p1.x = x-a.x;
13          p1.y = y-a.y;
14          return p1;
15      }
16  };
17  struct Circle{
18      double r;
19      Point centre;
20  };
21  struct Tri{
22      Point t[3];
23  };
24  Circle c;
25  Point a[maxn];
26  double dis(Point p1,Point p2)
27  {
28      Point p3;
29      p3.x = p2.x - p1.x;
30      p3.y = p2.y - p1.y;
31      return sqrt(p3.x * p3.x + p3.y * p3.y);
32  }
33  double triArea(Tri t)
34  {
35      Point p1,p2;
36      p1 = t.t[1] - t.t[0];
37      p2 = t.t[2] - t.t[0];
38      return fabs(p1.x * p2.y - p1.y * p2.x)/2;
39  }
40  Circle Wjy(Tri t)
41  {
42      Circle tmp;
43      double a,b,c,c1,c2;
44      double xa,ya,xb,yb,xc,yc;
45      a = dis(t.t[0],t.t[1]);
46      b = dis(t.t[1],t.t[2]);
47      c = dis(t.t[2],t.t[0]);
48      tmp.r = a*b*c / triArea(t)/4;
49      xa = t.t[0].x;ya = t.t[0].y;
50      xb = t.t[1].x;yb = t.t[1].y;
51      xc = t.t[2].x;yc = t.t[2].y;
52      c1 = (xa * xa + ya*ya - xb*xb -yb*yb)/2;
53      c2 = (xa * xa + ya*ya - xc*xc -yc*yc)/2;
54      tmp.centre.x = (c1 * (ya - yc) - c2 * (ya - yb)) / ((xa - xb) * (ya - yc)
    - (xa - xc) * ( ya - yb ));
```

```
    tmp.centre.y = (c1 * (xa - xc) - c2 * (xa - xb)) / ((ya - yb) * (xa - xc) - (ya - yc) * ( xa - xb ));
    return tmp;
}
Circle Mincircle2(int tce,Tri ce)
{
    Circle tmp;
    if(tce == 0) tmp.r = -2;
    else if(tce == 1)
    {
        tmp.centre = ce.t[0];
        tmp.r = 0;
    }
    else if(tce == 2)
    {
        tmp.r = dis(ce.t[0],ce.t[1])/2;
        tmp.centre.x = (ce.t[0].x + ce.t[1].x)/2;
        tmp.centre.y = (ce.t[0].y + ce.t[1].y)/2;
    }
    else if(tce == 3) tmp = Wjy(ce);
    return tmp;
}
void Mincircle(int t,int tce,Tri ce)
{
    Point tmp;
    c = Mincircle2(tce,ce);
    if(tce == 3) return ;
    for(int i=1;i<=t;i++)
    {
        if(dis(a[i],c.centre) > c.r)
        {
            ce.t[tce] = a[i];
            Mincircle (i-1, tce+1 , ce);
            tmp = a[i];
            for(int j=i;j>=2;j--)
            {
                a[j] = a[j-1];
            }
            a[1] = tmp;
        }
    }
}
void run ( int n)
{
    Tri ce;
    Mincircle(n,0,ce);
```

```
100     printf("%.2f %.2f %.2f\n",c.centre.x ,c.centre.y , c.r);
101 }
102 int main()
103 {
104     int n;
105     cin>>n;
106     for(int i=1;i<=n;i++)
107     {
108         scanf("%lf%lf",&a[i].x , & a[i].y);
109     }
110     run(n);
111 }
```