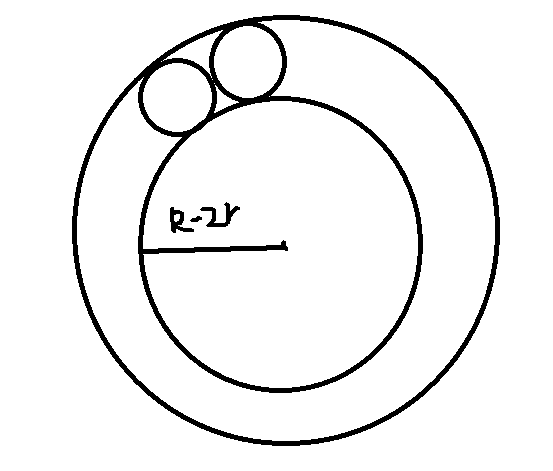
题意：

给出n幢建筑，每个都以一个点表示，给出点坐标。

有一个以原点为圆心，以R为半径的圆，记为圆O，是原始安全范围；

then，安全范围变为在原来那个圆内任意位置的以r为半径的圆（不会超出原来的圆），记为圆P；

求缩圈后，仍在安全范围内的概率最大的，所有的点。



R-2r之内的点，无论以它为圆心怎么画，画的圆都不会超出原始的大圆

而当落在R-2r之外的话，就会有些以它为圆心的圆面积超出原始的大圆，而题中要求新圆必须包含在原始大圆之内，所以以它为圆心的圆只有部分圆满足要求，其概率自然就低了

对于一幢建筑（或者说一个点），以其为圆心，做一个以半径为2r的圆Q；

圆Q在圆O内的面积，除以圆O的面积，得到的商即为这个点对应的概率。

显然这个概率，有关于：“点和原点的距离”；

当R>2\*r 时，

在以原点为圆心，以R-2\*r为半径的圆的范围内的点，安全概率最高且全部相同；

如果不存在这样的点，则越靠近原点越安全；

当R<=2\*r时，

在以原点为圆心，以2\*r-R为半径的圆的范围内的点，安全概率最高且全部相同；

如果不存在这样的点，则越靠近原点越安全；

#include<iostream>

#include<cstdio>

#include<cmath>

#define INF 1e9+7

using namespace std;

int a[110];

struct node

{

int id;

double radius;

}point[110];

int main()

{

//freopen("input.txt","r",stdin);

int n,R,r,x,y;

int T;

cin>>T;

while(T--)

{

cin>>n>>R>>r;

int cnt(0);

double minn=INF;

double \_radius=abs(R-2\*r);//这里用double是因为后面要与圆的半径sqrt(x\*x+y\*y)比较

for(int i=1;i<=n;i++)

{

cin>>x>>y;

point[i].id=i;

point[i].radius=sqrt(x\*x+y\*y);

if(point[i].radius<=\_radius)

a[++cnt]=i;

if(point[i].radius<minn)

minn=point[i].radius;

}

if(cnt>0)

{

cout<<cnt<<"\n";

for(int i=1;i<cnt;i++)

cout<<a[i]<<' ';

cout<<a[cnt]<<"\n";

}

else

{

for(int i=1;i<=n;i++)

if(point[i].radius==minn)

a[++cnt]=i;

cout<<cnt<<"\n";

for(int i=1;i<cnt;i++)

cout<<a[i]<<' ';

cout<<a[cnt]<<"\n";

}

}

return 0;

}