Pastebin Link: <https://paste.ubuntu.com/p/8K3DF7tPtx/>

*/\**  
 *Simply call maxPointCover()*  
 *Input: p[] contains the co-ordinates*  
 *of n points.*  
 *Output: Finds the maximum number of points*  
 *covered by a circle of radius r.*  
 *Complexity: O(n^2 log n)*  
*\*/*  
#define SIZE 2005  
#define EPS 1e-9  
#define XX first  
#define YY second  
**typedef** **long** **double** ld;  
**typedef** pair < ld, **bool** > plb;  
**struct** point{  
 **int** x, y;  
 point(){}  
 point(**int** xx, **int** yy){  
 x=xx;  
 y=yy;  
 }  
}p[SIZE];  
**int** n, r;  
complex < ld > cpy[SIZE];  
ld dis[SIZE][SIZE];  
  
**void** makePointsComplex(){  
 **for**(**int** i=0; i<n; i++){  
 cpy[i]=complex < ld > (p[i].x, p[i].y);  
 }  
}  
  
**bool** cmp(plb a, plb b){  
 **if**(abs(a.XX-b.XX)<EPS) **return** a.YY>b.YY;  
 **return** a.XX<b.XX;  
}  
  
plb angles[2\*SIZE];  
  
**int** maxPointCoverWithI(**int** i, ld r, **int** n){  
 **int** anglesz=0;  
 ld b, a, alpha, beta;  
 **for**(**int** j=0; j<n; j++){  
 **if**(i!=j && dis[i][j]<=(2.0\*r+EPS)){  
 b=acos(dis[i][j]/(2.0\*r));  
 a=arg(cpy[j]-cpy[i]);  
 alpha=a-b;  
 beta=a+b;  
 angles[anglesz++]=plb(alpha, **true**);  
 angles[anglesz++]=plb(beta, **false**);  
 }  
 }  
 sort(angles, angles+anglesz, cmp);  
 **int** cnt=1, res=1;  
 **for**(**int** i=0; i<anglesz; i++){  
 **if**(angles[i].YY) cnt++;  
 **else** cnt--;  
 res=max(cnt, res);  
 }  
 **return** res;  
}  
  
**int** maxPointCover(){  
 makePointsComplex();  
 **for**(**int** i=0; i<(n-1); i++){  
 **for**(**int** j=i+1; j<n; j++){  
 dis[i][j]=dis[j][i]=abs(cpy[i]-cpy[j]);  
 }  
 }  
 **int** ans=0;  
 **for**(**int** i=0; i<n; i++){  
 ans=max(ans, maxPointCoverWithI(i, r, n));  
 }  
 **return** ans;  
}