

```

1 //http://codeforces.com/contest/280/submission/3275722
2
3 const double eps=1e-8;
4 const double pi=acos(-1.0);
5 const double inf=1e20;
6 const int maxp=1111;
7 int dblcmp(double d)
8 {
9     if (fabs(d)<eps)return 0;
10    return d>eps?1:-1;
11 }
12 inline double sqr(double x){return x*x;}
13 /*
14 point() - Empty constructor
15 point(double x, double y) - constructor
16 input() - double input
17 output() - .21f output
18 operator == - compares x and y
19 operator < - compares first by x, then by y
20 len() - gives length from origin
21 len2() - gives square of length from origin
22 distance(point p) - gives distance from p
23 add(point p) - returns new point after adding curreseponding x and y
24 sub(point p) - returns new point after subtracting curreseponding x
and y
25 mul(double b) - returns new point after multiplieing x and y by b
26 div(double b) - returns new point after divideing x and y by b
27 dot(point p) - dot product
28 det(point p) - cross product of 2d points
29 rad(point a, point b) - Probably radius of circumcircle of the triangle
30 trunc(double r) - return point that is truncated the distance from
center to r
31 rotleft() - returns 90 degree ccw rotated point
32 rotright() - returns 90 degree cw rotated point
33 rotate(point p, double angle) - returns point after rotateing the point centering
at p by angle radian ccw
34 */
35 struct point
36 {
37     double x,y;
38     point() {}
39     point(double _x,double _y){ x = _x; y = _y;}
40     void input() { scanf("%lf%lf",&x,&y); }
41     void output() { printf("%.2f %.2f\n",x,y); }
42     bool operator==(point a)const{
43         return dblcmp(a.x - x) == 0 && dblcmp(a.y - y) == 0;
44     }
45     bool operator<(point a)const{
46         return dblcmp(a.x - x) == 0 ? dblcmp(y - a.y) < 0 : x < a.x;
47     }
48     point operator-(point a)const{
49         return point(x-a.x, y-a.y);
50     }
51     double len() { return hypot(x, y); }
52     double len2() { return x * x + y * y; }
53     double distance(point p){return hypot(x - p.x, y - p.y); }
54     point add(point p) { return point(x + p.x, y + p.y); }
55     point sub(point p) { return point(x - p.x, y - p.y); }
56     point mul(double b) { return point(x * b, y * b); }
57     point div(double b) { return point(x / b, y / b); }
58     double dot(point p) { return x*p.x+y*p.y; }
59     double det(point p) { return x*p.y-y*p.x; }
60     double rad(point a,point b){
61         point p=*this;
62         return fabs(atan2( fabs(a.sub(p).det(b.sub(p))),a.sub(p).dot(b.sub(p)) ));
63     }

```

```

64     point trunc(double r){
65         double l=len();
66         if (!dblcmp(l))return *this;
67         r/=l;
68         return point(x*r,y*r);
69     }
70     point rotleft()      {   return point(-y,x);      }
71     point rotright()     {   return point(y,-x);      }
72     point rotate(point p,double angle){
73         point v=this->sub(p);
74         double c=cos(angle),s=sin(angle);
75         return point(p.x+v.x*c-v.y*s,p.y+v.x*s+v.y*c);
76     }
77 };
78
79 /*
80 Stores two points
81
82 line()                - Empty constructor
83 line(point a, point b) - line through a and b
84 operator ==           - checks if two points are same
85 line(point p, double angle) - one end p, another end at angle degree
86 line(double a, double b, double c) - line of equation ax + by + c = 0
87 input()              - inputs a and b
88 adjust()             - orders in such a way that a < b
89 length()             - distance of ab
90 angle()              - returns 0 <= angle < 180
91 relation()           - 0 if collinear
92                     1 if ccw
93                     2 if cw
94 pointonseg(point p)   - returns 1 if point is on segment
95 parallel(line v)      - returns 1 if they are parallel
96 segcrossseg(line v)   - returns 0 if does not intersect
97                     returns 1 if non-standard intersection
98                     returns 2 if intersects
99 segcrossseg_inside(line v) - returns 1 if intersects strictly inside
100                     returns 0 if not
101 linecrossseg(line v)  - v is line
102 linecrossline(line v) - 0 if parallel
103                     1 if coincides
104                     2 if intersects
105 crosspoint(line v)    - returns intersection point
106 dispointtoline(point p) - distance from point p to the line
107 dispointtoseg(point p) - distance from p to the segment
108 lineprog(point p)     - returns projected point p on ab line
109 symmetrypoint(point p) - returns reflection point of p over ab
110 */
111 struct line
112 {
113     point a,b;
114     line() { }
115     line(point _a,point _b){ a=_a; b=_b; }
116     bool operator==(line v){ return (a==v.a)&&(b==v.b); }
117     line(point p,double angle){
118         a=p;
119         if (dblcmp(angle-pi/2)==0){
120             b=a.add(point(0,1));
121         }else{
122             b=a.add(point(1,tan(angle)));
123         }
124     }
125     //ax+by+c=0
126     line(double _a,double _b,double _c){
127         if (dblcmp(_a)==0){
128             a=point(0,-_c/_b);
129             b=point(1,-_c/_b);

```

```

130     }else if (dblcmp(_b)==0){
131         a=point(-_c/_a,0);
132         b=point(-_c/_a,1);
133     }else{
134         a=point(0,-_c/_b);
135         b=point(1,(-_c-_a)/_b);
136     }
137 }
138 void input()          {   a.input(); b.input();          }
139 void adjust()         {   if(b<a)swap(a,b);              }
140 double length()       {   return a.distance(b);          }
141 double angle(){
142     double k=atan2(b.y-a.y,b.x-a.x);
143     if (dblcmp(k)<0)k+=pi;
144     if (dblcmp(k-pi)==0)k-=pi;
145     return k;
146 }
147 int relation(point p){
148     int c=dblcmp(p.sub(a).det(b.sub(a)));
149     if (c<0)return 1;
150     if (c>0)return 2;
151     return 3;
152 }
153 bool pointonseg(point p){
154     return dblcmp(p.sub(a).det(b.sub(a)))==0&&dblcmp(p.sub(a).dot(p.sub(b)))<=0
;
155 }
156 bool parallel(line v){
157     return dblcmp(b.sub(a).det(v.b.sub(v.a)))==0;
158 }
159 int segcrossseg(line v){
160     int d1=dblcmp(b.sub(a).det(v.a.sub(a)));
161     int d2=dblcmp(b.sub(a).det(v.b.sub(a)));
162     int d3=dblcmp(v.b.sub(v.a).det(a.sub(v.a)));
163     int d4=dblcmp(v.b.sub(v.a).det(b.sub(v.a)));
164     if ((d1^d2)==-2&&(d3^d4)==-2)return 2;
165     return (d1==0&&dblcmp(v.a.sub(a).dot(v.a.sub(b)))<=0 ||
166         d2==0&&dblcmp(v.b.sub(a).dot(v.b.sub(b)))<=0 ||
167         d3==0&&dblcmp(a.sub(v.a).dot(a.sub(v.b)))<=0 ||
168         d4==0&&dblcmp(b.sub(v.a).dot(b.sub(v.b)))<=0);
169 }
170 int segcrossseg_inside(line v){
171     if(v.pointonseg(a) || v.pointonseg(b) || pointonseg(v.a) || pointonseg(v.b
)) return 0;
172     int d1=dblcmp(b.sub(a).det(v.a.sub(a)));
173     int d2=dblcmp(b.sub(a).det(v.b.sub(a)));
174     int d3=dblcmp(v.b.sub(v.a).det(a.sub(v.a)));
175     int d4=dblcmp(v.b.sub(v.a).det(b.sub(v.a)));
176     if ((d1^d2)==-2&&(d3^d4)==-2)return 1;
177     return (d1==0&&dblcmp(v.a.sub(a).dot(v.a.sub(b)))<=0 ||
178         d2==0&&dblcmp(v.b.sub(a).dot(v.b.sub(b)))<=0 ||
179         d3==0&&dblcmp(a.sub(v.a).dot(a.sub(v.b)))<=0 ||
180         d4==0&&dblcmp(b.sub(v.a).dot(b.sub(v.b)))<=0);
181 }
182 int linecrossseg(line v){/*this seg v line
183     int d1=dblcmp(b.sub(a).det(v.a.sub(a)));
184     int d2=dblcmp(b.sub(a).det(v.b.sub(a)));
185     if ((d1^d2)==-2)return 2;
186     return (d1==0 || d2==0);
187 }
188 int linecrossline(line v){
189     if ((*this).parallel(v)){
190         return v.relation(a)==3;
191     }
192     return 2;
193 }

```

```

194     point crosspoint(line v){
195         double a1=v.b.sub(v.a).det(a.sub(v.a));
196         double a2=v.b.sub(v.a).det(b.sub(v.a));
197         return point((a.x*a2-b.x*a1)/(a2-a1),(a.y*a2-b.y*a1)/(a2-a1));
198     }
199     double dispointtoline(point p){
200         return fabs(p.sub(a).det(b.sub(a)))/length();
201     }
202     double dispointtoseg(point p){
203         if (dblcmp(p.sub(b).dot(a.sub(b)))<0||dblcmp(p.sub(a).dot(b.sub(a)))<0){
204             return min(p.distance(a),p.distance(b));
205         }
206         return dispointtoline(p);
207     }
208     point lineprog(point p){
209         return a.add(b.sub(a).mul(b.sub(a).dot(p.sub(a))/b.sub(a).len2()));
210     }
211     point symmetrypoint(point p){
212         point q=lineprog(p);
213         return point(2*q.x-p.x,2*q.y-p.y);
214     }
215 };
216
217 /*
218 a circle of point p and radius r
219
220 circle() -empty constructor
221 circle(point p,double r) -circle of point p and radius r
222 circle(point a,point b,point c) -circumcircle of triangle of abc
223 circle(point a,point b,point c,bool t)-incircle of triangle of abc, bool t is
nothing
224 input() -takes input of a circle
225 output() -outputs a circle
226 operator== -checks for equality
227 operator< -comparison opertaor
228 area() -area of the circle
229 circumference() -circumference of the circle
230 relation(point p) -0 outside
231 -1 on circumference
232 -2 inside circle
233 relationseg(line v) -0 outside
234 -1 on circumference
235 -2 inside circle
236 relationline(line v) -0 outside
237 -1 on circumference
238 -2 inside circle
239 getcircle(point a,point b,double r,circle&c1,circle&c2)
240 -returns two circle c1,c2 through points a,b
of radius r
241 -returns 0 for nor circle
242 getcircle(line u,point q,double r1,circle &c1,circle &c2)
243 -returns two circle c1,c2 which is tangent to
line u, goes through
244 point q and has radius r1
245 returns 0 for no circle ,1 if c1=c2 ,2 if
c1!=c2
246 getcircle(line u,line v,double r1,circle &c1,circle &c2,circle &c3,circle &c4)
247 -returns 4 circles which is tangent to line
u,v has radius r1.
248 getcircle(circle cx,circle cy,double r1,circle&c1,circle&c2)
249 -not sure
250 pointcrossline(line v,point &p1,point &p2)
251 -not sure
252 relationcircle(circle v) -1 for
253 -2
254 -3

```

```

255                                     4
256                                     5
257 pointcrosscircle(circle v,point &p1,point &p2)
258                                     -not sure what it does
259 tangentline(point q,line &u,line &v) -not sure what it does
260 areacircle(circle v)               -intersection area of circle v
261 areatriangle(point a,point b)      -intersection area of circle and triangle of
point a,b,p
262 */
263 struct circle
264 {
265     point p;
266     double r;
267     circle() { }
268     circle(point _p,double _r): p(_p),r(_r){ }
269     circle(double x,double y,double _r): p(point(x,y)),r(_r){ }
270     circle(point a,point b,point c){
271         p=line(a.add(b).div(2),a.add(b).div(2).add(b.sub(a).rotleft())).crosspoint(
line(c.add(b).div(2),c.add(b).div(2).add(b.sub(c).rotleft())));
272         r=p.distance(a);
273     }
274     circle(point a,point b,point c,bool t){
275         line u,v;
276         double m=atan2(b.y-a.y,b.x-a.x),n=atan2(c.y-a.y,c.x-a.x);
277         u.a=a;
278         u.b=u.a.add(point(cos((n+m)/2),sin((n+m)/2)));
279         v.a=b;
280         m=atan2(a.y-b.y,a.x-b.x),n=atan2(c.y-b.y,c.x-b.x);
281         v.b=v.a.add(point(cos((n+m)/2),sin((n+m)/2)));
282         p=u.crosspoint(v);
283         r=line(a,b).dispointtoseg(p);
284     }
285     void input() { p.input();scanf("%lf",&r); }
286     void output() { printf("%.2lf %.2lf %.2lf\n",p.x,p.y,r); }
287     bool operator==(circle v){
288         return ((p==v.p)&&dblcmp(r-v.r)==0);
289     }
290     bool operator<(circle v)const{
291         return ((p<v.p) || (p==v.p)&&dblcmp(r-v.r)<0);
292     }
293     double area() { return pi*sqr(r); }
294     double circumference(){ return 2*pi*r; }
295     int relation(point b){
296         double dst=b.distance(p);
297         if (dblcmp(dst-r)<0)return 2;
298         if (dblcmp(dst-r)==0)return 1;
299         return 0;
300     }
301     int relationseg(line v){
302         double dst=v.dispointtoseg(p);
303         if (dblcmp(dst-r)<0)return 2;
304         if (dblcmp(dst-r)==0)return 1;
305         return 0;
306     }
307     int relationline(line v){
308         double dst=v.dispointtoline(p);
309         if (dblcmp(dst-r)<0)return 2;
310         if (dblcmp(dst-r)==0)return 1;
311         return 0;
312     }
313     int getcircle(point a,point b,double r,circle&c1,circle&c2){
314         circle x(a,r),y(b,r);
315         int t=x.pointcrosscircle(y,c1.p,c2.p);
316         if (!t)return 0;
317         c1.r=c2.r=r;
318         return t;

```

```

319     }
320     int getcircle(line u,point q,double r1,circle &c1,circle &c2){
321         double dis=u.dispointtoline(q);
322         if (dblcmp(dis-r1*2)>0)return 0;
323         if (dblcmp(dis)==0){
324             c1.p=q.add(u.b.sub(u.a).rotleft().trunc(r1));
325             c2.p=q.add(u.b.sub(u.a).rotright().trunc(r1));
326             c1.r=c2.r=r1;
327             return 2;
328         }
329         line u1=line(u.a.add(u.b.sub(u.a).rotleft().trunc(r1)),u.b.add(u.b.sub(u.a)
330 ).rotleft().trunc(r1));
331         line u2=line(u.a.add(u.b.sub(u.a).rotright().trunc(r1)),u.b.add(u.b.sub(u.a)
332 ).rotright().trunc(r1));
333         circle cc=circle(q,r1);
334         point p1,p2;
335         if (!cc.pointcrossline(u1,p1,p2))cc.pointcrossline(u2,p1,p2);
336         c1=circle(p1,r1);
337         if (p1==p2) { c2=c1;return 1; }
338         c2=circle(p2,r1);
339         return 2;
340     }
341     int getcircle(line u,line v,double r1,circle &c1,circle &c2,circle &c3,circle &
342 c4){
343         if (u.parallel(v))return 0;
344         line u1=line(u.a.add(u.b.sub(u.a).rotleft().trunc(r1)),u.b.add(u.b.sub(u.a)
345 ).rotleft().trunc(r1));
346         line u2=line(u.a.add(u.b.sub(u.a).rotright().trunc(r1)),u.b.add(u.b.sub(u.a)
347 ).rotright().trunc(r1));
348         line v1=line(v.a.add(v.b.sub(v.a).rotleft().trunc(r1)),v.b.add(v.b.sub(v.a)
349 ).rotleft().trunc(r1));
350         line v2=line(v.a.add(v.b.sub(v.a).rotright().trunc(r1)),v.b.add(v.b.sub(v.a)
351 ).rotright().trunc(r1));
352         c1.r=c2.r=c3.r=c4.r=r1;
353         c1.p=u1.crosspoint(v1);
354         c2.p=u1.crosspoint(v2);
355         c3.p=u2.crosspoint(v1);
356         c4.p=u2.crosspoint(v2);
357         return 4;
358     }
359 }
360 int getcircle(circle cx,circle cy,double r1,circle&c1,circle&c2){
361     circle x(cx.p,r1+cx.r),y(cy.p,r1+cy.r);
362     int t=x.pointcrosscircle(y,c1.p,c2.p);
363     if (!t)return 0;
364     c1.r=c2.r=r1;
365     return t;
366 }
367 int pointcrossline(line v,point &p1,point &p2){
368     if (!(*this).relationline(v))return 0;
369     point a=v.lineprog(p);
370     double d=v.dispointtoline(p);
371     d=sqrt(r*r-d*d);
372     if (dblcmp(d)==0){ p1=a; p2=a; return 1; }
373     p1=a.sub(v.b.sub(v.a).trunc(d));
374     p2=a.add(v.b.sub(v.a).trunc(d));
375     return 2;
376 }
377 int relationcircle(circle v){
378     double d=p.distance(v.p);
379     if (dblcmp(d-r-v.r)>0)return 5;
380     if (dblcmp(d-r-v.r)==0)return 4;
381     double l=fabs(r-v.r);
382     if (dblcmp(d-r-v.r)<0&&dblcmp(d-l)>0)return 3;
383     if (dblcmp(d-l)==0)return 2;
384     if (dblcmp(d-l)<0)return 1;

```

```

378     }
379     int pointcrosscircle(circle v,point &p1,point &p2){
380         int rel=relationcircle(v);
381         if (rel==1||rel==5)return 0;
382         double d=p.distance(v.p);
383         double l=(d+(sqr(r)-sqr(v.r))/d)/2;
384         double h=sqrt(sqr(r)-sqr(l));
385         p1=p.add(v.p.sub(p).trunc(l).add(v.p.sub(p).rotleft().trunc(h)));
386         p2=p.add(v.p.sub(p).trunc(l).add(v.p.sub(p).rotright().trunc(h)));
387         if (rel==2||rel==4)return 1;
388         return 2;
389     }
390     int tangentline(point q,line &u,line &v){
391         int x=relation(q);
392         if (x==2)return 0;
393         if (x==1){
394             u=line(q,q.add(q.sub(p).rotleft()));
395             v=u; return 1;
396         }
397         double d=p.distance(q);
398         double l=sqr(r)/d;
399         double h=sqrt(sqr(r)-sqr(l));
400         u=line(q,p.add(q.sub(p).trunc(l).add(q.sub(p).rotleft().trunc(h))));
401         v=line(q,p.add(q.sub(p).trunc(l).add(q.sub(p).rotright().trunc(h))));
402         return 2;
403     }
404     double areacircle(circle v){
405         int rel=relationcircle(v);
406         if (rel>=4)return 0.0;
407         if (rel<=2)return min(area(),v.area());
408         double d=p.distance(v.p);
409         double hf=(r+v.r+d)/2.0;
410         double ss=2*sqrt(hf*(hf-r)*(hf-v.r)*(hf-d));
411         double al=acos((r*r+d*d-v.r*v.r)/(2.0*r*d));
412         al=al*r*r;
413         double a2=acos((v.r*v.r+d*d-r*r)/(2.0*v.r*d));
414         a2=a2*v.r*v.r;
415         return al+a2-ss;
416     }
417     double areatriangle(point a,point b){
418         if (dblcmp(p.sub(a).det(p.sub(b))==0))return 0.0;
419         point q[5];
420         int len=0;
421         q[len++]=a;
422         line l(a,b);
423         point p1,p2;
424         if (pointcrossline(l,q[1],q[2])==2){
425             if (dblcmp(a.sub(q[1]).dot(b.sub(q[1]))<0)q[len++]=q[1];
426             if (dblcmp(a.sub(q[2]).dot(b.sub(q[2]))<0)q[len++]=q[2];
427         }
428         q[len++]=b;
429         if (len==4&&(dblcmp(q[0].sub(q[1]).dot(q[2].sub(q[1]))>0))swap(q[1],q[2]));
430         double res=0;
431         int i;
432         for (i=0;i<len-1;i++){
433             if (relation(q[i])==0||relation(q[i+1])==0){
434                 double arg=p.rad(q[i],q[i+1]);
435                 res+=r*r*arg/2.0;
436             }
437             else res+=fabs(q[i].sub(p).det(q[i+1].sub(p))/2.0);
438         }
439         return res;
440     }
441 };
442
443 /*

```

```

444 n, p, line l for each side
445
446 input(n)                - inputs n size polygon
447 add(point p)            - adds a point at end of the list
448 getline()              - populates line array
449 cmp                     - comparison in convex_hull order
450 norm()                  - sorting in convex_hull order
451 getconvex(polygon &convex) - returns convex hull in convex (monotone chain)
452 isconvex()              - checks if convex
453 relationpoint(point q)  - returns 3 if q is a vertex
454                          2 if on a side
455                          1 if inside
456                          0 if outside
457 relationline(line u)    - returns 1 if there is some intersection
458                          0 if no intersection
459                          2 if intersect at corner
460 convexcut(line u,polygon &po) - left side of u in po
461 getcircumference()      - returns side length
462 getarea()               - returns area
463 getdir()                - returns 0 for cw, 1 for ccw
464 getbarycentre()         - returns barycenter / cg
465 areaintersection(polygon po) - not implemented
466 areaunion(polygon po)   - not implemented
467 areacircle(circle c)    - intersection area of circle and polygon
468 relationcircle(circle c) - returns 0 if outside circle
469                          1 if tangent
470                          2 if inside
471 mincircle()             - returns minimum enclosing circle
472 circlecover()           - i think there is mistake. it tries to find
minimum enclosing circle
473 pointinpolygon(point q) - -1 if not on polygon, non negative number.. side
index
474 inside_polygon(point q, int on_edge=1)
475     - returns on_edge if on edge, otherwise 0 for outside 1 for inside
476 isdiagonal(int a, int b) - checks if p[a], p[b] is diagonal or not. returns
0/1
477 */
478 struct polygon
479 {
480     int n;
481     point p[maxp];
482     line l[maxp];
483     void input(int _n){
484         n=_n;
485         for (int i=0;i<n;i++) p[i].input();
486     }
487     void add(point q) { p[n++]=q; }
488     void getline(){
489         for (int i=0;i<n;i++)
490             l[i]=line(p[i],p[(i+1)%n]);
491     }
492     struct cmp{
493         point p;
494         cmp(const point &p0){p=p0;}
495         bool operator()(const point &aa,const point &bb){
496             point a=aa,b=bb;
497             int d=dblcmp(a.sub(p).det(b.sub(p)));
498             if (d==0)
499                 return dblcmp(a.distance(p)-b.distance(p))<0;
500             return d>0;
501         }
502     };
503     void norm(){
504         point mi=p[0];
505         for (int i=1;i<n;i++)mi=min(mi,p[i]);
506         sort(p,p+n,cmp(mi));

```



```

507     }
508 void getconvex(polygon &convex){
509     int i;
510     sort(p,p+n);
511     convex.n=n;
512     for (i=0;i<min(n,2);i++) convex.p[i]=p[i];
513     if (n<=2)return;
514     int &top=convex.n;
515     top=1;
516     for (i=2;i<n;i++){
517         while (top&&convex.p[top].sub(p[i]).det(convex.p[top-1].sub(p[i]))<=0)
518             top--;
519         convex.p[++top]=p[i];
520     }
521     int temp=top;
522     convex.p[++top]=p[n-2];
523     for (i=n-3;i>=0;i--){
524         while (top!=temp&&convex.p[top].sub(p[i]).det(convex.p[top-1].sub(p[i
525     ]))<=0)
526         top--;
527         convex.p[++top]=p[i];
528     }
529 bool isconvex(){
530     bool s[3];
531     memset(s,0,sizeof(s));
532     int i,j,k;
533     for (i=0;i<n;i++){
534         j=(i+1)%n;
535         k=(j+1)%n;
536         s[dblcmp(p[j].sub(p[i]).det(p[k].sub(p[i])))+1]=1;
537         if (s[0]&&s[2])return 0;
538     }
539     return 1;
540 }
541 int relationpoint(point q){
542     int i,j;
543     for (i=0;i<n;i++){
544         if (p[i]==q)return 3;
545     }
546     getline();
547     for (i=0;i<n;i++){
548         if (l[i].pointonseg(q))return 2;
549     }
550     int cnt=0;
551     for (i=0;i<n;i++){
552         j=(i+1)%n;
553         int k=dblcmp(q.sub(p[j]).det(p[i].sub(p[j])));
554         int u=dblcmp(p[i].y-q.y);
555         int v=dblcmp(p[j].y-q.y);
556         if (k>0&&u<0&&v>=0)cnt++;
557         if (k<0&&v<0&&u>=0)cnt--;
558     }
559     return cnt!=0;
560 }
561 int relationline(line u){
562     int i,k=0;
563     getline();
564     for (i=0;i<n;i++){
565         if (l[i].segcrossseg(u)==2)return 1;
566         if (l[i].segcrossseg(u)==1)k=1;
567     }
568     if (!k)return 0;
569     vector<point>vp;
570     for (i=0;i<n;i++){
571         if (l[i].segcrossseg(u)){

```

```

572         if (l[i].parallel(u)){
573             vp.pb(u.a);
574             vp.pb(u.b);
575             vp.pb(l[i].a);
576             vp.pb(l[i].b);
577             continue;
578         }
579         vp.pb(l[i].crosspoint(u));
580     }
581 }
582 sort(vp.begin(),vp.end());
583 int sz=vp.size();
584 for (i=0;i<sz-1;i++){
585     point mid=vp[i].add(vp[i+1]).div(2);
586     if (relationpoint(mid)==1)return 1;
587 }
588 return 2;
589 }
590 void convexcut(line u,polygon &po){
591     int i;
592     int &top=po.n;
593     top=0;
594     for (i=0;i<n;i++){
595         int d1=dblcmp(p[i].sub(u.a).det(u.b.sub(u.a)));
596         int d2=dblcmp(p[(i+1)%n].sub(u.a).det(u.b.sub(u.a)));
597         if (d1>=0)po.p[top++]=p[i];
598         if (d1*d2<0)po.p[top++]=u.crosspoint(line(p[i],p[(i+1)%n]));
599     }
600 }
601 double getcircumference(){
602     double sum=0;
603     int i;
604     for (i=0;i<n;i++)
605         sum+=p[i].distance(p[(i+1)%n]);
606     return sum;
607 }
608 double getarea(){
609     double sum=0;
610     int i;
611     for (i=0;i<n;i++)
612         sum+=p[i].det(p[(i+1)%n]);
613     return fabs(sum)/2;
614 }
615 bool getdir(){
616     double sum=0;
617     int i;
618     for (i=0;i<n;i++)
619         sum+=p[i].det(p[(i+1)%n]);
620     if (dblcmp(sum)>0)return 1;
621     return 0;
622 }
623 point getbarycentre(){
624     point ret(0,0);
625     double area=0;
626     int i;
627     for (i=1;i<n-1;i++){
628         double tmp=p[i].sub(p[0]).det(p[i+1].sub(p[0]));
629         if (dblcmp(tmp)==0)continue;
630         area+=tmp;
631         ret.x+=(p[0].x+p[i].x+p[i+1].x)/3*tmp;
632         ret.y+=(p[0].y+p[i].y+p[i+1].y)/3*tmp;
633     }
634     if (dblcmp(area))ret=ret.div(area);
635     return ret;
636 }
637 double areaintersection(polygon po){

```

```

638 double areaunion(polygon po){
639     return getarea()+po.getarea()-areaintersection(po);
640 }
641 double areacircle(circle c){
642     int i,j,k,l,m;
643     double ans=0;
644     for (i=0;i<n;i++){
645         int j=(i+1)%n;
646         if (dblcmp(p[j].sub(c.p).det(p[i].sub(c.p)))>=0)
647             ans+=c.areastriangle(p[i],p[j]);
648         else ans-=c.areastriangle(p[i],p[j]);
649     }
650     return fabs(ans);
651 }
652 int relationcircle(circle c){
653     getline();
654     int i,x=2;
655     if (relationpoint(c.p)!=1)return 0;
656     for (i=0;i<n;i++){
657         if (c.relationseg(l[i])==2)return 0;
658         if (c.relationseg(l[i])==1)x=1;
659     }
660     return x;
661 }
662 void find(int st,point tri[],circle &c){
663     if (!st) c=circle(point(0,0),-2);
664     if (st==1) c=circle(tri[0],0);
665     if (st==2) c=circle(tri[0].add(tri[1]).div(2),tri[0].distance(tri[1])/2.0);
666     if (st==3) c=circle(tri[0],tri[1],tri[2]);
667 }
668 void solve(int cur,int st,point tri[],circle &c){
669     find(st,tri,c);
670     if (st==3)return;
671     int i;
672     for (i=0;i<cur;i++){
673         if (dblcmp(p[i].distance(c.p)-c.r)>0){
674             tri[st]=p[i];
675             solve(i,st+1,tri,c);
676         }
677     }
678 }
679 circle mincircle(){
680     random_shuffle(p,p+n);
681     point tri[4];
682     circle c;
683     solve(n,0,tri,c);
684     return c;
685 }
686 int circlecover(double r){
687     int ans=0,i,j;
688     vector<pair<double,int> >v;
689     for (i=0;i<n;i++){
690         v.clear();
691         for (j=0;j<n;j++)if (i!=j){
692             point q=p[i].sub(p[j]);
693             double d=q.len();
694             if (dblcmp(d-2*r)<=0){
695                 double arg=atan2(q.y,q.x);
696                 if (dblcmp(arg)<0)arg+=2*pi;
697                 double t=acos(d/(2*r));
698                 v.push_back(make_pair(arg-t+2*pi,-1));
699                 v.push_back(make_pair(arg+t+2*pi,1));
700             }
701         }
702         sort(v.begin(),v.end());
703         int cur=0;

```

```

704         for (j=0;j<v.size();j++){
705             if (v[j].second==-1)++cur;
706             else --cur;
707             ans=max(ans,cur);
708         }
709     }
710     return ans+1;
711 }
712 int pointinpolygon(point q){
713     if (getdir())reverse(p,p+n);
714     if (dblcmp(q.sub(p[0]).det(p[n-1].sub(p[0]))==0){
715         if (line(p[n-1],p[0]).pointonseg(q))return n-1;
716         return -1;
717     }
718     int low=1,high=n-2,mid;
719     while (low<=high){
720         mid=(low+high)>>1;
721         if (dblcmp(q.sub(p[0]).det(p[mid].sub(p[0]))>=0&&dblcmp(q.sub(p[0]).
det(p[mid+1].sub(p[0]))<0){
722             polygon c;
723             c.p[0]=p[mid];
724             c.p[1]=p[mid+1];
725             c.p[2]=p[0];
726             c.n=3;
727             if (c.relationpoint(q))return mid;
728             return -1;
729         }
730         if (dblcmp(q.sub(p[0]).det(p[mid].sub(p[0]))>0) low=mid+1;
731         else high=mid-1;
732     }
733     return -1;
734 }
735
736 double xmult(point a, point b, point c){
737     return (b - a).det(c - a);
738 }
739
740 int inside_polygon(point q, int on_edge=1){
741     point q2;
742     int i=0,count;
743     while (i<n){
744         for(count = i = 0, q2.x = rand()+10000, q2.y = rand() + 10000; i<n; i
745 ++){
746             if(dblcmp(xmult(q,p[i],p[(i+1)%n]))==0 && (p[i].x-q.x)*(p[(i+1)%n].
x-q.x)<eps && (p[i].y-q.y)*(p[(i+1)%n].y-q.y)<eps)
747                 return on_edge;
748             else if(dblcmp(xmult(q,q2,p[i]))==0)
749                 break;
750             else if(xmult(q,p[i],q2)*xmult(q,p[(i+1)%n],q2)<-eps&&xmult(p[i],q,
p[(i+1)%n])*xmult(p[i],q2,p[(i+1)%n])<-eps)
751                 count++;
752         }
753         return count&1;
754     }
755 }
756 int isdiagonal(int a, int b){
757     int i;
758     if(a == b || (a + 1)%n == b || (b + 1)%n == a) return 0;
759     getline();
760     line x(p[a], p[b]);
761     for(i = 0; i < n; i++) if(a != i && b != i)
762         if(x.pointonseg(p[i]))
763             return 0;
764     for(i = 0; i < n; i++)
765         if(l[i].segcrossseg_inside(x))
766             return 0;
767     point y = p[a].add(p[b]).div(2.);

```

```

766         if(inside_polygon(y, 0) == 0) return 0;
767         return 1;
768     }
769 };
770
771 const int maxn=500;
772 struct circles
773 {
774     circle c[maxn];
775     double ans[maxn];
776     double pre[maxn];
777     int n;
778     circles() { }
779     void add(circle cc) { c[n++]=cc; }
780     bool inner(circle x,circle y){
781         if (x.relationcircle(y)!=1)return 0;
782         return dblcmp(x.r-y.r)<=0?1:0;
783     }
784     void init_or(){
785         int i,j,k=0;
786         bool mark[maxn]={0};
787         for (i=0;i<n;i++){
788             for (j=0;j<n;j++)if (i!=j&&!mark[j]){
789                 if ((c[i]==c[j])||inner(c[i],c[j]))break;
790             }
791             if (j<n)mark[i]=1;
792         }
793         for (i=0;i<n;i++)if (!mark[i])c[k++]=c[i];
794         n=k;
795     }
796     void init_and(){
797         int i,j,k=0;
798         bool mark[maxn]={0};
799         for (i=0;i<n;i++){
800             for (j=0;j<n;j++)if (i!=j&&!mark[j])
801                 if ((c[i]==c[j])||inner(c[j],c[i]))break;
802             if (j<n)mark[i]=1;
803         }
804         for (i=0;i<n;i++)if (!mark[i])c[k++]=c[i];
805         n=k;
806     }
807     double areaarc(double th,double r){
808         return 0.5*sqr(r)*(th-sin(th));
809     }
810     void getarea(){
811         int i,j,k;
812         memset(ans,0,sizeof(ans));
813         vector<pair<double,int> >v;
814         for (i=0;i<n;i++){
815             v.clear();
816             v.push_back(make_pair(-pi,1));
817             v.push_back(make_pair(pi,-1));
818             for (j=0;j<n;j++)if (i!=j){
819                 point q=c[j].p.sub(c[i].p);
820                 double ab=q.len(),ac=c[i].r,bc=c[j].r;
821                 if (dblcmp(ab+ac-bc)<=0){
822                     v.push_back(make_pair(-pi,1));
823                     v.push_back(make_pair(pi,-1));
824                     continue;
825                 }
826                 if (dblcmp(ab+bc-ac)<=0)continue;
827                 if (dblcmp(ab-ac-bc)>0) continue;
828                 double th=atan2(q.y,q.x),fai=acos((ac*ac+ab*ab-bc*bc)/(2.0*ac*ab));
829                 double a0=th-fai;
830                 if (dblcmp(a0+pi)<0)a0+=2*pi;
831                 double a1=th+fai;

```

```

832         if (dblcmp(a1-pi)>0)a1-=2*pi;
833         if (dblcmp(a0-a1)>0){
834             v.push_back(make_pair(a0,1));
835             v.push_back(make_pair(pi,-1));
836             v.push_back(make_pair(-pi,1));
837             v.push_back(make_pair(a1,-1));
838         }else{
839             v.push_back(make_pair(a0,1));
840             v.push_back(make_pair(a1,-1));
841         }
842     }
843     sort(v.begin(),v.end());
844     int cur=0;
845     for (j=0;j<v.size();j++){
846         if (cur&&dblcmp(v[j].first-pre[cur])){
847             ans[cur]+=areaarc(v[j].first-pre[cur],c[i].r);
848             ans[cur]+=0.5*point(c[i].p.x+c[i].r*cos(pre[cur]),c[i].p.y+c[i]
).r*sin(pre[cur])).det(point(c[i].p.x+c[i].r*cos(v[j].first),c[i].p.y+c[i].r*sin(v[j].
first)));
849         }
850         cur+=v[j].second;
851         pre[cur]=v[j].first;
852     }
853 }
854 for (i=1;i<=n;i++) ans[i]-=ans[i+1];
855 }
856 };
857 struct halfplane:public line
858 {
859     double angle;
860     halfplane() { }
861     halfplane(point _a,point _b){ a=_a; b=_b; }
862     halfplane(line v) { a=v.a; b=v.b; }
863     void calcangle() { angle=atan2(b.y-a.y,b.x-a.x); }
864     bool operator<(const halfplane &b)const{
865         return angle<b.angle;
866     }
867 };
868 struct halfplanes
869 {
870     int n;
871     halfplane hp[maxp];
872     point p[maxp];
873     int que[maxp];
874     int st,ed;
875     void push(halfplane tmp){ hp[n++]=tmp; }
876     void unique(){
877         int m=1,i;
878         for (i=1;i<n;i++){
879             if (dblcmp(hp[i].angle-hp[i-1].angle))hp[m++]=hp[i];
880             else if (dblcmp(hp[m-1].b.sub(hp[m-1].a).det(hp[i].a.sub(hp[m-1].a))>0
))hp[m-1]=hp[i];
881         }
882         n=m;
883     }
884     bool halfplaneinsert(){
885         int i;
886         for (i=0;i<n;i++)hp[i].calcangle();
887         sort(hp,hp+n);
888         unique();
889         que[st=0]=0;
890         que[ed=1]=1;
891         p[1]=hp[0].crosspoint(hp[1]);
892         for (i=2;i<n;i++){
893             while (st<ed&&dblcmp((hp[i].b.sub(hp[i].a).det(p[ed].sub(hp[i].a)))<0)
ed--;

```

```

894         while (st<ed&&dblcmp((hp[i].b.sub(hp[i].a).det(p[st+1].sub(hp[i].a)))<
0)st++;
895         que[++ed]=i;
896         if (hp[i].parallel(hp[que[ed-1]]))return false;
897         p[ed]=hp[i].crosspoint(hp[que[ed-1]]);
898     }
899     while (st<ed&&dblcmp(hp[que[st]].b.sub(hp[que[st]].a).det(p[ed].sub(hp[que[
st]].a)))<0)ed--;
900     while (st<ed&&dblcmp(hp[que[ed]].b.sub(hp[que[ed]].a).det(p[st+1].sub(hp[
que[ed]].a)))<0)st++;
901     if (st+1>=ed)return false;
902     return true;
903 }
904 void getconvex(polygon &con){
905     p[st]=hp[que[st]].crosspoint(hp[que[ed]]);
906     con.n=ed-st+1;
907     int j=st,i=0;
908     for (;j<=ed;i++,j++) con.p[i]=p[j];
909 }
910 };
911 /*
912 3d point
913 rotate function is not complete. every else is same as 2d
914 */
915 struct point3
916 {
917     double x,y,z;
918     point3() { }
919     point3(double _x,double _y,double _z):
920     x(_x),y(_y),z(_z) { }
921     void input() { scanf("%lf%lf%lf",&x,&y,&z); }
922     void output() { printf("%.2lf %.2lf %.2lf\n",x,y,z); }
923     bool operator==(point3 a){
924         return dblcmp(a.x-x)==0&&dblcmp(a.y-y)==0&&dblcmp(a.z-z)==0;
925     }
926     bool operator<(point3 a)const {
927         return dblcmp(a.x-x)==0?dblcmp(y-a.y)==0?dblcmp(z-a.z)<0:y<a.y:x<a.x;
928     }
929     double len() { return sqrt(len2()); }
930     double len2() { return x*x+y*y+z*z; }
931     double distance(point3 p){
932         return sqrt((p.x-x)*(p.x-x)+(p.y-y)*(p.y-y)+(p.z-z)*(p.z-z));
933     }
934     point3 add(point3 p){ return point3(x+p.x,y+p.y,z+p.z); }
935     point3 sub(point3 p){ return point3(x-p.x,y-p.y,z-p.z); }
936     point3 mul(double d){ return point3(x*d,y*d,z*d); }
937     point3 div(double d){ return point3(x/d,y/d,z/d); }
938     double dot(point3 p){ return x*p.x+y*p.y+z*p.z; }
939     point3 det(point3 p){
940         return point3(y*p.z-p.y*z,p.x*z-x*p.z,x*p.y-p.x*y);
941     }
942     double rad(point3 a,point3 b){
943         point3 p=(*this);
944         return acos(a.sub(p).dot(b.sub(p))/(a.distance(p)*b.distance(p)));
945     }
946     point3 trunc(double r){
947         r/=len();
948         return point3(x*r,y*r,z*r);
949     }
950     point3 rotate(point3 o,double r)
951     {
952     }
953 };
954
955 /*
956 3d line

```

```

957
958 */
959 struct line3
960 {
961     point3 a,b;
962     line3() {}
963     line3(point3 _a,point3 _b){ a=_a; b=_b; }
964     bool operator==(line3 v){ return (a==v.a)&&(b==v.b); }
965     void input() { a.input(); b.input(); }
966     double length() { return a.distance(b); }
967     bool pointonseg(point3 p){
968         return dblcmp(p.sub(a).det(p.sub(b)).len())==0&&dblcmp(a.sub(p).dot(b.sub(p
969     ))<=0;
970 }
971 double dispointtoline(point3 p){
972     return b.sub(a).det(p.sub(a)).len()/a.distance(b);
973 }
974 double dispointtoseg(point3 p){
975     if (dblcmp(p.sub(b).dot(a.sub(b)))<0||dblcmp(p.sub(a).dot(b.sub(a)))<0){
976         return min(p.distance(a),p.distance(b));
977     }
978     return dispointtoline(p);
979 }
980 point3 lineprog(point3 p){
981     return a.add(b.sub(a).trunc(b.sub(a).dot(p.sub(a))/b.distance(a)));
982 }
983 point3 rotate(point3 p,double ang){
984     if (dblcmp((p.sub(a).det(p.sub(b)).len())==0)return p;
985     point3 f1=b.sub(a).det(p.sub(a));
986     point3 f2=b.sub(a).det(f1);
987     double len=fabs(a.sub(p).det(b.sub(p)).len()/a.distance(b));
988     f1=f1.trunc(len);f2=f2.trunc(len);
989     point3 h=p.add(f2);
990     point3 pp=h.add(f1);
991     return h.add((p.sub(h)).mul(cos(ang*1.0)).add((pp.sub(h)).mul(sin(ang*1.0
992 )));
993 }
994 */
995 plane
996 struct plane
997 {
998     point3 a,b,c,o;
999     plane() {}
1000     plane(point3 _a,point3 _b,point3 _c){
1001         a=_a;
1002         b=_b;
1003         c=_c;
1004         o=pvec();
1005     }
1006     plane(double _a,double _b,double _c,double _d){
1007         //ax+by+cz+d=0
1008         o=point3(_a,_b,_c);
1009         if (dblcmp(_a)!=0){ a=point3((-_d-_c-_b)/_a,1,1); }
1010         else if (dblcmp(_b)!=0){
1011             a=point3(1,(-_d-_c-_a)/_b,1);
1012         }
1013         else if (dblcmp(_c)!=0){a=point3(1,1,(-_d-_a-_b)/_c);}
1014     }
1015     void input(){
1016         a.input();
1017         b.input();
1018         c.input();
1019         o=pvec();
1020     }

```



```

1021     point3 pvec()          {   return b.sub(a).det(c.sub(a));   }
1022 bool pointonplane(point3 p){
1023     return dblcmp(p.sub(a).dot(o))==0;
1024 }
1025
1026 int pointontriangle(point3 p){
1027     if (!pointonplane(p))return 0;
1028     double s=a.sub(b).det(c.sub(b)).len();
1029     double s1=p.sub(a).det(p.sub(b)).len();
1030     double s2=p.sub(a).det(p.sub(c)).len();
1031     double s3=p.sub(b).det(p.sub(c)).len();
1032     if (dblcmp(s-s1-s2-s3))return 0;
1033     if (dblcmp(s1)&&dblcmp(s2)&&dblcmp(s3))return 2;
1034     return 1;
1035 }
1036 bool relationplane(plane f){
1037     if (dblcmp(o.det(f.o).len()))return 0;
1038     if (pointonplane(f.a))return 2;
1039     return 1;
1040 }
1041 double angleplane(plane f){
1042     return acos(o.dot(f.o)/(o.len()*f.o.len()));
1043 }
1044 double dispoint(point3 p){
1045     return fabs(p.sub(a).dot(o)/o.len());
1046 }
1047 point3 pttoplane(point3 p){
1048     line3 u=line3(p,p.add(o));
1049     crossline(u,p);
1050     return p;
1051 }
1052 int crossline(line3 u,point3 &p){
1053     double x=o.dot(u.b.sub(a));
1054     double y=o.dot(u.a.sub(a));
1055     double d=x-y;
1056     if (dblcmp(fabs(d))==0)return 0;
1057     p=u.a.mul(x).sub(u.b.mul(y)).div(d);
1058     return 1;
1059 }
1060 int crossplane(plane f,line3 &u){
1061     point3 oo=o.det(f.o);
1062     point3 v=o.det(oo);
1063     double d=fabs(f.o.dot(v));
1064     if (dblcmp(d)==0)return 0;
1065     point3 q=a.add(v.mul(f.o.dot(f.a.sub(a))/d));
1066     u=line3(q,q.add(oo));
1067     return 1;
1068 }
1069 };
1070
1071 polygon a,b;
1072 int main()
1073 {
1074     int i,j,k;
1075     a.n=b.n=4;
1076     double w,h,g;
1077     cin>>w>>h>>g;
1078     g/=180.0;
1079     g*=pi;
1080     a.p[0]=point(-w/2.0,-h/2.0);
1081     a.p[1]=point(w/2.0,-h/2.0);
1082     a.p[2]=point(w/2.0,h/2.0);
1083     a.p[3]=point(-w/2.0,h/2.0);
1084     for (i=0;i<4;i++)
1085     {
1086         b.p[i]=(a.p[i]).rotate(point(0,0),g);

```

```
1087     }
1088     polygons p;
1089     p.push(a);p.push(b);
1090     printf("%.12lf\n",p.polyareaunion());
1091     return 0;
1092 }
```