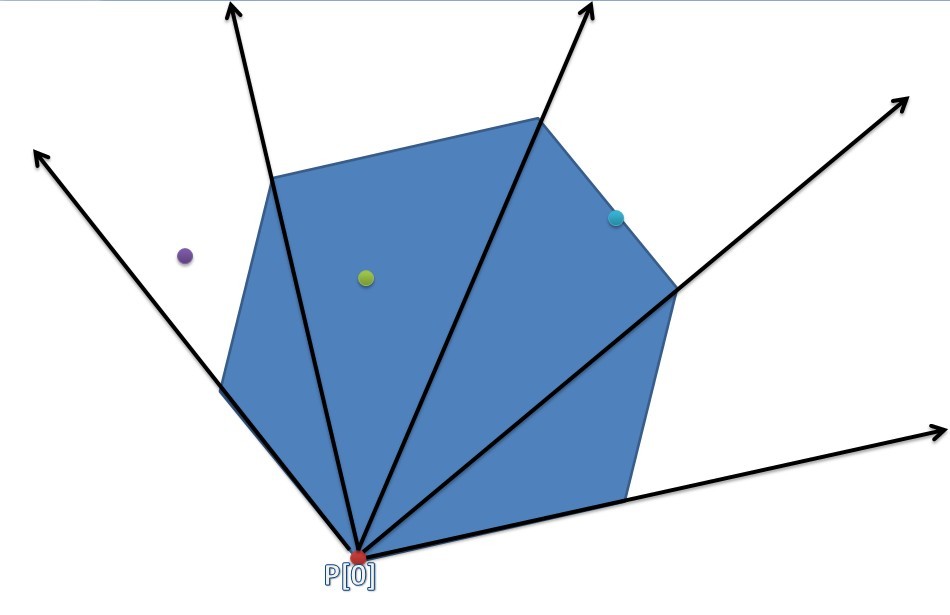
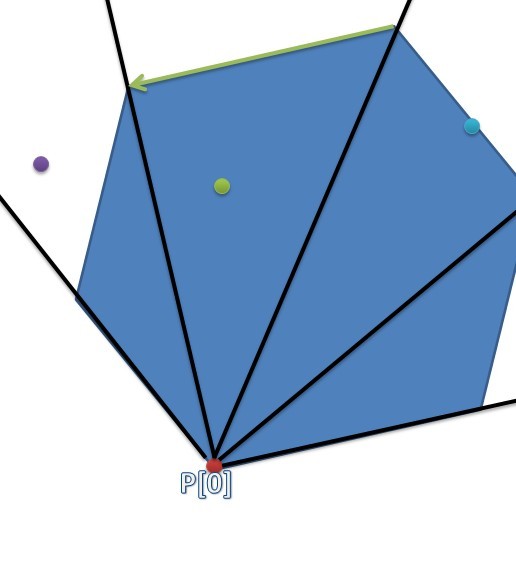
题意简单易懂...给你n个点的凸包(已经按逆时针顺序给出)...判断m个点是否在凸包内...数量>=k就输出YES





首先，将一个凸多边形划分为 N 个三角形的区域。

对于某一个点，如果不在这些三角形区域内，则必不在凸包内。否则就能通过二分的方法得到点所在的三角形区间（因为此点的与p[0]的斜率必处于某个三角形两个向量斜率的中间）

最后只需判断点与原凸包边的关系即可。

假设我们查询绿色的点是否在凸包内，我们首先二分得到了它所在的区间，然后判断它和绿色的向量的关系，蓝色和紫色的点类似，蓝色的点在边界上，紫色的点在边界右边。

由于应用了二分，时间复杂度为O(logN)。

#include<bits/stdc++.h>

#define ll long long

#define inf 1e18

#define mn 100005

using namespace std;

const double dinf=12345678910,eps=1e-10,pi=acos(-1);

struct point{

double x,y,k;

point(double x=0,double y=0):x(x),y(y){ }//构造函数

}a[mn],b[mn];

typedef point vect;

vect operator + (vect A,vect B){return vect(A.x+B.x,A.y+B.y);}

vect operator - (point A,point B){return vect(A.x-B.x,A.y-B.y);}

vect operator \* (vect A,double p){return vect(A.x\*p,A.y\*p);}

vect operator / (vect A,double p){return vect(A.x/p,A.y/p);}

bool operator < (const point& a,const point& b) {return a.x<b.x||(a.x==b.x&&a.y<b.y);}

int sgn(double x){if(fabs(x)<eps)return 0;else return x<0?-1:1;}//求一个小数的符号

bool operator == (const point& a, const point& b){return !sgn(a.x - b.x) && !sgn(a.y - b.y);}

double polar\_angle(vect A){return atan2(A.y,A.x);}//求向量的倾斜角

inline double d\_to\_r(double D){return pi/180\*D;}//角度转弧度

double cross(vect A,vect B){return A.x\*B.y-B.x\*A.y;}//求两个向量的叉积

double side(point a,point b,point c){return cross(b-a,c-a);}//为正则表示点c(ac向量)在ab向量的逆时针方向

double dis(point a,point b){return sqrt((a.x-b.x)\*(a.x-b.x)+(a.y-b.y)\*(a.y-b.y));}//两点间距离

vect rotate(vect A,double rad)//计算向量逆时针旋转rad后的向量

{return vect(A.x\*cos(rad)-A.y\*sin(rad),A.x\*sin(rad)+A.y\*cos(rad));}

ll ans=0,n,m,p;

bool cmp(point A,point B){return A.k<B.k||(A.k==B.k&&dis(A,a[1])<dis(B,a[1]));}//一种常见的将凸包极角排序的方法

ll check(point x)

{ll ret,l=3,r=n-1;

if(x.k<a[l-1].k||x.k>a[r].k)return 0;

while(l<=r)

{

ll mid=(l+r)/2;

if(a[mid].k>=x.k){r=mid-1;ret=mid;}

else l=mid+1;

}

return side(a[ret-1],a[ret],x)>=0;

}

int main()

{

ll x,y,z,i,j,k;

char ch;

cin>>n>>m>>p;

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

{

scanf("%lf%lf",&a[i].x,&a[i].y);

if(a[i]<a[1])swap(a[i],a[1]);

}

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

a[i].k=atan2(a[i].y-a[1].y,a[i].x-a[1].x);

sort(a+2,a+n+1,cmp);

a[n+1]=a[1];

n++;//这样a里面的点集就是首尾相接的

for(i=1;i<=m;i++)

{

scanf("%lf%lf",&b[i].x,&b[i].y);

b[i].k=atan2(b[i].y-a[1].y,b[i].x-a[1].x);

}

for(i=1;i<=m;i++)

if(check(b[i]))ans++;

if(ans>=p) puts("YES");

else puts("NO");

return 0;

}