#include <cstdio>

#include <algorithm>

using namespace std;

#define inf 1000000000

struct P

{

int x, y, i;

} pt[100010];

bool operator < (P a, P b)

{

return a.x < b.x;

}

int np[100010];

int t[200010], ed[200010], c;

int g[100010][5], e[100010][5];

bool find\_path(int s)

{

int p=s, k=0, l=0;

while (l < 4)

{

if (g[p][k] != -1 && t[e[p][k]] < 2 && (ed[e[p][k]] == -1 || ed[e[p][k]] == c))

{

t[e[p][k]]++;

ed[e[p][k]] = c;

p = g[p][k];

k = (k+3)%4;

if (p == s)

return 1;

l = -inf;

}

else

{

k = (k+1)%4;

l++;

}

}

return 0;

}

int main()

{

int n, m, p, q, k, i;

scanf("%d", &n);

for (i = 0; i < n; i++)

{

scanf("%d%d", &pt[i].x, &pt[i].y);

pt[i].i = i;

}

sort(pt, pt+n);

for (i = 0; i < n; i++)

np[pt[i].i] = i;

memset(g, -1, sizeof(g));

scanf("%d", &m);

for (i = 0; i < m; i++)

{

scanf("%d%d", &p, &q);

p = np[p-1];

q = np[q-1];

if (pt[p].x == pt[q].x)

if (pt[p].y < pt[q].y)

k = 3;

else

k = 1;

else

if (pt[p].x < pt[q].x)

k = 2;

else

k = 0;

g[p][k] = q;

g[q][(k+2)%4] = p;

e[p][k] = e[q][(k+2)%4] = i;

}

c = 0;

memset(t, 0, sizeof(t));

memset(ed, -1, sizeof(ed));

for (i = 0; i < n; i++)

while (find\_path(i))

c++;

p = 0;

for (i = 0; i < m; i++)

if (t[i] == 2)

p++;

printf("%d\n", p);

for (i = 0; i < m; i++)

if (t[i] == 2)

printf("%d\n", i+1);

return 0;

}