

# 题意

给出平面上4条线段，判断这4条线段是否恰好围成一个面积大于0的矩形。

每组数据包含4行，每行包含4个整数 $x_1, y_1, x_2, y_2$  ( $0 \leq x_1, y_1, x_2, y_2 \leq 100000$ )；其中 $(x_1, y_1), (x_2, y_2)$ 代表一条线段的两个端点。

# 分析

先求四条线段的斜率（考虑斜率不存在的情况为INF）

如果能围成矩形 → 对边斜率相等，邻边斜率乘积为-1（或者一个为0，一个为INF），所有邻边都相交

定义结构体

```
1 struct Point{
2     int x,y;
3 };
4 struct Seg{
5     Point p1,p2;
6     double k;
7 };
```

计算斜率

```
1 if(a[i].p1.x == a[i].p2.x) a[i].k = INF;
2 else a[i].k = (a[i].p2.y - a[i].p1.y)*1.0 / (a[i].p2.x - a[i].p1.x);
3 if(a[i].k == -0) a[i].k = 0;
```

判断线段相等模板

```
1 inline double mult(Point a,Point b,Point c){
2     return (a.x - c.x ) * (b.y - c.y) - (b.x - c.x) * (a.y - c.y);
3 }
4 inline bool judge(Seg a,Seg b){
5     if ( max(a.p1.x , a.p2.x) < min(b.p1.x , b.p2.x) ) return false;
6     if ( max(a.p1.y , a.p2.y) < min(b.p1.y , b.p2.y) ) return false;
7     if ( max(b.p1.x , b.p2.x) < min(a.p1.x , a.p2.x) ) return false;
8     if ( max(b.p1.y , b.p2.y) < min(a.p1.y , a.p2.y) ) return false;
```

```

9      if ( mult(b.p1 , a.p2 , a.p1) * mult(a.p2 , b.p2 , a.p1)<0 ) return false;

10     if ( mult(a.p1 , b.p2 , b.p1) * mult(b.p2 , a.p2 , b.p1)<0 ) return false;

11     return true;
12 }

```

## 代码

```

1  #include<algorithm>
2  #include<bitset>
3  #include<cstdio>
4  #include<cstring>
5  #include<cstdlib>
6  #include<cmath>
7  #include<deque>
8  #include<iostream>
9  #include<map>
10 #include<queue>
11 #include<set>
12 #include<stack>
13 #include<string>
14 #include<vector>
15 #include<list>
16 #define For(i,a,b) for(int i=(a); i<=(b) ; i++)
17 #define _For(i,a,b) for(int i=(a); i>=(b) ; i--)
18 #define Memset(a,b); memset((a),(b),sizeof((a)));
19 #define Cout(a,b); printf("%d",(a));printf(b);
20 #define Coutc(a,b); printf("%c",(a));printf(b);
21 #define Couts(a,b); printf("%s",(a));printf(b);
22 using namespace std;
23 const int INF = 0x3f3f3f3f;
24 typedef long long LL;typedef unsigned long long ULL;typedef long double
    LDB;
25 inline LL CinLL(){LL x=0,f=1;char ch=getchar();while(ch<'0' || ch>'9'){if(ch=='-'
    ')f=-1;ch=getchar();}while(ch>='0'&&ch<='9'){x=x*10+ch-
    '0';ch=getchar();}return x*f;}
26 inline int Cin(){int x=0,f=1;char ch=getchar();while(!isdigit(ch)){if(ch=='-'
    ')f=-1;ch=getchar();}while(isdigit(ch))x=x*10+ch-'0',ch=getchar();return f*x;}
27 const double eps = 1e-7;
28 struct Point{
29     int x,y;
30 };

```

```

31 struct Seg{
32     Point p1,p2;
33     double k;
34 };
35 bool cmp(Seg x,Seg y){
36     return x.k < y.k;
37 }
38 inline double mult(Point a,Point b,Point c){
39     return (a.x - c.x ) * (b.y - c.y) - (b.x - c.x) * (a.y - c.y);
40 }
41 inline bool judge(Seg a,Seg b){
42     if ( max(a.p1.x , a.p2.x) < min(b.p1.x , b.p2.x) ) return false;
43     if ( max(a.p1.y , a.p2.y) < min(b.p1.y , b.p2.y) ) return false;
44     if ( max(b.p1.x , b.p2.x) < min(a.p1.x , a.p2.x) ) return false;
45     if ( max(b.p1.y , b.p2.y) < min(a.p1.y , a.p2.y) ) return false;
46     if ( mult(b.p1 , a.p2 , a.p1) * mult(a.p2 , b.p2 , a.p1)<0 ) return false;
47     if ( mult(a.p1 , b.p2 , b.p1) * mult(b.p2 , a.p2 , b.p1)<0 ) return false;
48     return true;
49 }
50 int main()
51 {
52     ios::sync_with_stdio(false);
53     int _;
54     Seg a[5];
55     cin>>_;
56     while(_-->0)
57     {
58         For(i,1,4)
59         {
60             cin>>a[i].p1.x>>a[i].p1.y>>a[i].p2.x>>a[i].p2.y;
61             if(a[i].p1.x == a[i].p2.x) a[i].k = INF;
62             else a[i].k = (a[i].p2.y - a[i].p1.y)*1.0 / (a[i].p2.x -
a[i].p1.x);
63             if(a[i].k == -0) a[i].k = 0;
64         }
65         sort(a+1,a+5,cmp);
66         if(a[1].k == a[2].k && a[3].k == a[4].k)
67             if(abs((a[1].k - 0.0)< eps && abs(a[3].k - INF) <eps )|| a[1].k *
a[3].k == -1.0 )
68                 if(judge(a[1],a[3]) && judge(a[1],a[4]) && judge(a[2],a[3]) &&
judge(a[2],a[4]))
69                     cout<<"YES"<<endl;
70             else
71                 cout<<"NO"<<endl;

```

```
72         else
73             cout<<"NO"<<endl;
74     else
75         cout<<"NO"<<endl;
76 }
77 }
```