class Point:

     def \_\_init\_\_(self, x, y):

         self.x = x

         self.y = y

     def subtract(self, p):

          return Point(self.x - p.x, self.y - p.y)

def cross\_product(p1, p2):

      return p1.x \* p2.y - p2.x \* p1.y

def direction(p1, p2, p3):

     return  cross\_product(p3.subtract(p1), p2.subtract(p1))

def on\_segment(p1, p2, p):

     return min(p1.x, p2.x) <= p.x <= max(p1.x, p2.x) and min(p1.y, p2.y) <= p.y <= max(p1.y, p2.y)

def intersect(p1, p2, p3, p4):

     d1 = direction(p3, p4, p1)

     d2 = direction(p3, p4, p2)

     d3 = direction(p1, p2, p3)

     d4 = direction(p1, p2, p4)

     if ((d1 > 0 and d2 < 0) or (d1 < 0 and d2 > 0)) and ((d3 > 0 and d4 < 0) or (d3 < 0 and d4 > 0)):

         return True

     elif d1 == 0 and on\_segment(p3, p4, p1):

         return True

     elif d2 == 0 and on\_segment(p3, p4, p2):

         return True

     elif d3 == 0 and on\_segment(p1, p2, p3):

         return True

     elif d4 == 0 and on\_segment(p1, p2, p4):

         return True

     else:

         return False

#False

point1 = Point(1,1)

point2 = Point(10,1)

point3 = Point(1,2)

point4 = Point(10,2)

result = intersect(point1, point2, point3, point4)

print(result)

#True

point1 = Point(10,1)

point2 = Point(0,10)

point3 = Point(0,0)

point4 = Point(10,10)

result = intersect(point1, point2, point3, point4)

print(result)

#False

point1 = Point(-5,-5)

point2 = Point(0,0)

point3 = Point(1,1)

point4 = Point(10,10)

result = intersect(point1, point2, point3, point4)

print(result)