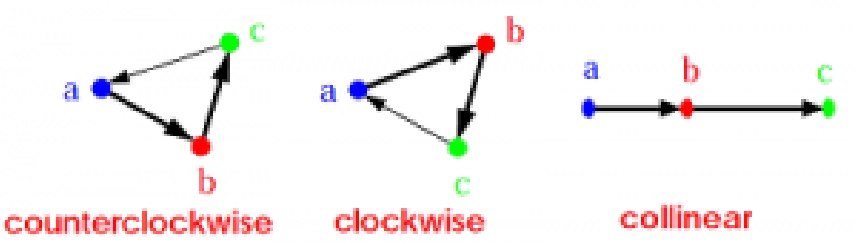
How to check if two given line segments intersect?

Given two line segments (p1, q1) and (p2, q2), find if the given line segments intersect with each other.

Before we discuss solution, let us define notion of **orientation**. Orientation of an ordered triplet of points in the plane can be
–counterclockwise
–clockwise
–colinear
The following diagram shows different possible orientations of (a, b, c)



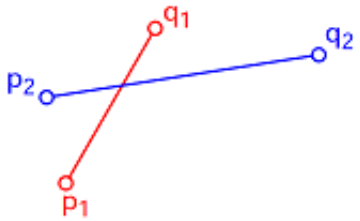
Note the word 'ordered' here. Orientation of (a, b, c) may be different from orientation of (c, b, a).

How is Orientation useful here?
Two segments (p1,q1) and (p2,q2) intersect if and only if one of the following two conditions is verified

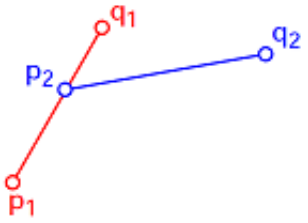
1. *General Case:*
– (p1, q1, p2) and (p1, q1, q2) have different orientations and
– (p2, q2, p1) and (p2, q2, q1) have different orientations

2. *Special Case*
– (p1, q1, p2), (p1, q1, q2), (p2, q2, p1), and (p2, q2, q1) are all collinear and
– the x-projections of (p1, q1) and (p2, q2) intersect
– the y-projections of (p1, q1) and (p2, q2) intersect

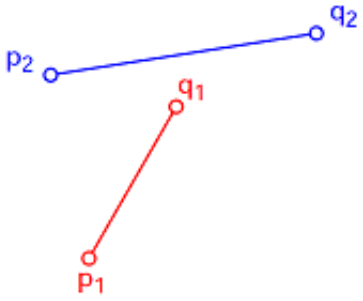
Examples of General Case:



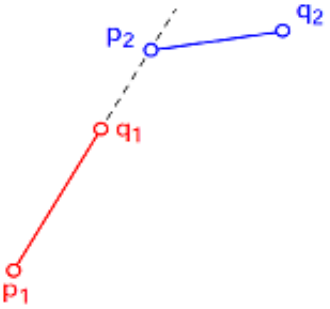
Example 1: Orientations of (p1, q1, p2) and (p1, q1, q2) are different. Orientations of (p2, q2, p1) and (p2, q2, q1) are also different



Example 2: Orientations of (p1, q1, p2) and (p1, q1, q2) are different. Orientations of (p2, q2, p1) and (p2, q2, q1) are also different

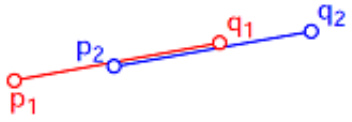


Example 3: Orientations of (p1, q1, p2) and (p1, q1, q2) are different. Orientations of (p2, q2, p1) and (p2, q2, q1) are **same**

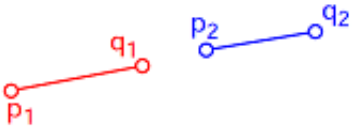


Example 4: Orientations of (p1, q1, p2) and (p1, q1, q2) are different. Orientations of (p2, q2, p1) and (p2, q2, q1) are **same**

Examples of Special Case:



Example 1: All points are colinear. The x-projections of (p1, q1) and (p2, q2) intersect. The y-projections of (p1, q1) and (p2, q2) intersect



Example 2: All points are colinear. The x-projections of (p1, q1) and (p2, q2) do not intersect. The y-projections of (p1, q1) and do not (p2, q2) intersect

Following is C++ implementation based on above idea.

```
// A C++ program to check if two given line segments int
#include <iostream>
using namespace std;

struct Point
{
    int x;
    int y;
};

// Given three colinear points p, q, r, the function che
// point q lies on line segment 'pr'
bool onSegment(Point p, Point q, Point r)
{
    if (q.x <= max(p.x, r.x) && q.x >= min(p.x, r.x) &&
        q.y <= max(p.y, r.y) && q.y >= min(p.y, r.y))
       return true;
```

```
    return false;
}

// To find orientation of ordered triplet (p, q, r).
// The function returns following values
// 0 --> p, q and r are colinear
// 1 --> Clockwise
// 2 --> Counterclockwise
int orientation(Point p, Point q, Point r)
{
    // See 10th slides from following link for derivatio
    // http://www.dcs.gla.ac.uk/~pat/52233/slides/Geomet
    int val = (q.y - p.y) * (r.x - q.x) -
              (q.x - p.x) * (r.y - q.y);

    if (val == 0) return 0;  // colinear

    return (val > 0)? 1: 2; // clock or counterclock wis
}

// The main function that returns true if line segment '
// and 'p2q2' intersect.
bool doIntersect(Point p1, Point q1, Point p2, Point q2)
{
    // Find the four orientations needed for general and
    // special cases
    int o1 = orientation(p1, q1, p2);
    int o2 = orientation(p1, q1, q2);
    int o3 = orientation(p2, q2, p1);
    int o4 = orientation(p2, q2, q1);

    // General case
    if (o1 != o2 && o3 != o4)
        return true;

    // Special Cases
    // p1, q1 and p2 are colinear and p2 lies on segment
    if (o1 == 0 && onSegment(p1, p2, q1)) return true;

    // p1, q1 and p2 are colinear and q2 lies on segment
    if (o2 == 0 && onSegment(p1, q2, q1)) return true;

    // p2, q2 and p1 are colinear and p1 lies on segment
    if (o3 == 0 && onSegment(p2, p1, q2)) return true;
```

```
     // p2, q2 and q1 are colinear and q1 lies on segment
    if (o4 == 0 && onSegment(p2, q1, q2)) return true;

    return false; // Doesn't fall in any of the above cases
}

// Driver program to test above functions
int main()
{
    struct Point p1 = {1, 1}, q1 = {10, 1};
    struct Point p2 = {1, 2}, q2 = {10, 2};

    doIntersect(p1, q1, p2, q2)? cout << "Yes\n": cout <<

    p1 = {10, 0}, q1 = {0, 10};
    p2 = {0, 0}, q2 = {10, 10};
    doIntersect(p1, q1, p2, q2)? cout << "Yes\n": cout <<

    p1 = {-5, -5}, q1 = {0, 0};
    p2 = {1, 1}, q2 = {10, 10};
    doIntersect(p1, q1, p2, q2)? cout << "Yes\n": cout <<

    return 0;
}
```

Output:

```
No
Yes
No
```

Sources:

http://www.dcs.gla.ac.uk/~pat/52233/slides/Geometry1x1.pdf

Introduction to Algorithms 3rd Edition by Clifford Stein, Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest