Orientation of 3 ordered points - GeeksforGeeks

Source: https://www.geeksforgeeks.org/orientation-3-ordered-points/

 Orientation of 3 ordered points Last Updated : 23 Jul, 2025 Given three points p1, p2, and p3, the task is to determine the orientation of these three points. Orientation of an ordered triplet of points in the plane can be counterclockwise clockwise collinear The following diagram shows different possible orientations of (a, b, c) If orientation of (p1, p2, p3) is collinear, then orientation of (p3, p2, p1) is also collinear. If orientation of (p1, p2, p3) is clockwise, then orientation of (p3, p2, p1) is counterclockwise and vice versa is also true. Example: Input: p1 = [0, 0], p2 = [4, 4], p3 = [1, 2] Output: Counterclockwise Input: p1 = {0, 0}, p2 = {4, 4}, p3 = {1, 1} Output: Collinear Approach: The idea is to find the slopes of line segments formed by points [p1, p2] and [p2, p3] and calculate their cross product. Let the slope formed by point p1 & p2 be x, and the other one be y. Now there are three possibilities: if x - y == 0 : If the slopes formed by points are equal, it means that all three points lies in a straight line, and they are collinear . if x - y > 0 : If the slope formed by point [p1, p2] is greater than [p2, p3], it means that right turn structure is forming, thus it is clockwise orientation. if x - y < 0 : If the slope formed by point [p1, p2] is smaller than [p2, p3], it means that left turn structure is forming, thus it is counterclockwise orientation. Considering the above image, the slope of line segment formed by point p1 and p2 will be: x = (y2 - y1) / (x2 - x1). And the slope formed by line segment formed by point p2 and p3 will be: y = (y3-y2) / x3 - x2). The cross product of both the slopes will be (y2 - y1) * (x3 - x2) - (x2 - x1) * (y3 - y2); Below is the implementation of above idea: C++ // C++ program to find the orientation // of the three points #include <bits/stdc++.h> using namespace std ; // Function to find the orientation void orientation (int x1 , int y1 , int x2 , int y2 , int x3 , int y3) { // orientation of an (x, y) triplet int val = ((y2 - y1) * (x3 - x2)) - ((x2 - x1) * (y3 - y2)) ; if (val == 0) cout << "Collinear" ; else if (val > 0) cout << "Clockwise" ; else cout << "CounterClockwise" ; } int main () { int x1 = 0 , y1 = 0 , x2 = 4 ; int y2 = 4 , x3 = 1 , y3 = 1 ; orientation (x1 , y1 , x2 , y2 , x3 , y3); return 0 ; } Java // Java program to find the orientation // of the three points class GfG { // Function to find the orientation static void orientation (int x1 , int y1 , int x2 , int y2 , int x3 , int y3) { // orientation of an (x, y) triplet int val = ((y2 - y1) * (x3 - x2)) - ((x2 - x1) * (y3 - y2)); if (val == 0) System . out . println ("Collinear"); else if (val > 0) System . out . println ("Clockwise"); else System . out . println ("CounterClockwise"); } public static void main (String [] args) { int x1 = 0 , y1 = 0 , x2 = 4 ; int y2 = 4 , x3 = 1 , y3 = 1 ; orientation (x1 , y1 , x2 , y2 , x3 , y3); } } Python # Python program to find the orientation # of the three points # Function to find the orientation def orientation (x1 , y1 , x2 , y2 , x3 , y3): # orientation of an (x, y) triplet val = ((y2 - y1) * (x3 - x2)) - \ ((x2 - x1) * (y3 - y2)) if val == 0 : print ("Collinear") elif val > 0 : print ("Clockwise") else : print ("CounterClockwise") if __name__ == "__main__" : x1 , y1 , x2 , y2 , x3 , y3 = 0 , 0 , 4 , 4 , 1 , 1 orientation (x1 , y1 , x2 , y2 , x3 , y3) C# // C# program to find the orientation // of the three points using System ; class GfG { // Function to find the orientation static void orientation (int x1 , int y1 , int x2 , int y2 , int x3 , int y3) { // orientation of an (x, y) triplet int val = ((y2 - y1) * (x3 - x2)) - ((x2 - x1) * (y3 - y2)); if (val == 0) Console . WriteLine ("Collinear"); else if (val > 0) Console . WriteLine ("Clockwise"); else Console . WriteLine ("CounterClockwise"); } static void Main (string [] args) { int x1 = 0 , y1 = 0 , x2 = 4 ; int y2 = 4 , x3 = 1 , y3 = 1 ; orientation (x1 , y1 , x2 , y2 , x3 , y3); } } JavaScript // JavaScript program to find the orientation // of the three points // Function to find the orientation function orientation (x1 , y1 , x2 , y2 , x3 , y3) { // orientation of an (x, y) triplet const val = ((y2 - y1) * (x3 - x2)) - ((x2 - x1) * (y3 - y2)); if (val === 0) console . log ("Collinear"); else if (val > 0) console . log ("Clockwise"); else console . log ("CounterClockwise"); } const x1 = 0 , y1 = 0 , x2 = 4 , y2 = 4 , x3 = 1 , y3 = 1 ; orientation (x1 , y1 , x2 , y2 , x3 , y3); Output Collinear Time Complexity: O(1) Auxiliary Space: O(1) The concept of orientation is used in below articles: Find Simple Closed Path for a given set of points How to check if two given line segments intersect? Convex Hull | Set 1 (Jarvis's Algorithm or Wrapping) Convex Hull | Set 2 (Graham Scan) Comment Article Tags: Article Tags: Geometric DSA