```
# C# program to find convex hull of a set of points. Refer
# https://www.geeksforgeeks.org/orientation-3-ordered-points/
# for explanation of orientation()

# point class with x, y as point
class Point:
        def __init__(self, x, y):
                self.x = x
                self.y = y

def Left_index(points):

        '''
        Finding the left most point
        '''
        minn = 0
        for i in range(1,len(points)):
                if points[i].x < points[minn].x:
                        minn = i
                elif points[i].x == points[minn].x:
                        if points[i].y > points[minn].y:
                                minn = i
        return minn

def orientation(p, q, r):
        '''
        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
        '''
        val = (q.y - p.y) * (r.x - q.x) - \
                (q.x - p.x) * (r.y - q.y)

        if val == 0:
                return 0
        elif val > 0:
                return 1
        else:
                return 2

def convexHull(points, n):

        # There must be at least 3 points
        if n < 3:
                return

        # Find the leftmost point
        l = Left_index(points)

        hull = []

        '''
        Start from leftmost point, keep moving counterclockwise
        until reach the start point again. This loop runs O(h)
        times where h is number of points in result or output.
        '''
        p = l
        q = 0
        while(True):

                # Add current point to result
                hull.append(p)

                '''
                Search for a point 'q' such that orientation(p, q,
                x) is counterclockwise for all points 'x'. The idea
                is to keep track of last visited most counterclock-
                wise point in q. If any point 'i' is more
counterclock-
                wise than q, then update q.
                '''
                q = (p + 1) % n

                for i in range(n):

                        # If i is more counterclockwise
                        # than current q, then update q
                        if(orientation(points[p],points[i], points[q])
== 2):
                                q = i

                '''
                Now q is the most counterclockwise with respect to p
                Set p as q for next iteration, so that q is added to
                result 'hull'
                '''
                p = q

                # While we don't come to first point
                if(p == l):
                        break

        return_points=[]
        # Print Result
        for each in hull:
                return_points.append(points[each])
                print(points[each].x, points[each].y)

        return return_points


# Driver Code
'''
points = []
points.append(Point(6, 8))
points.append(Point(1, 6))
points.append(Point(4, 8))
points.append(Point(6, 7))

convexHull(points, len(points))

# This code is contributed by
# Akarsh Somani, IIIT Kalyani
'''
```