Name:- Deependra Shukla
Roll no. :- 202210101200013

link:- video link

Method	Input	No. of comparison
Left to right	[(1, 3), (1, 5), (1, 7), (1, 7), (1, 9), (2, 2), (2, 4), (2, 5), (3, 2), (3, 5), (3, 6), (4, 8), (5, 4), (5, 7), (8, 0), (8, 2), (8, 5), (9, 1), (9, 6), (9, 7)]	37
Right to left	[(1, 3), (1, 5), (1, 7), (1, 7), (1, 9), (2, 2), (2, 4), (2, 5), (3, 2), (3, 5), (3, 6), (4, 8), (5, 4), (5, 7), (8, 0), (8, 2), (8, 5), (9, 1), (9, 6), (9, 7)]	20

Observation:-

Here I have taken inputs by myself randomly and then I used a sorting method to sort it on the basis of x coordinate. I applied 2 different methods left to right and right to left. When I use a method in which points are taken right to left, it is just comparing y coordinate so it is comparing all elements.
In a method where points are taken from left to right it is a little bit complicated because we need to take that if a point has x coordinate less than next point then it must contain its y coordinate maximum than further upcoming points. That's why it is taking many comparison

```
def leftRight(input):
   maximalPoints = []  # Initialize an empty list to store maximal points
   steps = 0  # Initialize a counter to keep track of the number of steps
taken
   for i in input:  # Iterate over each point in the input list
       steps += 1  # Increment the step counter
       while len(maximalPoints) > 0 and maximalPoints[-1][1] <= i[1]:  #
Check if the current point is greater than the top of the stack
           steps += 1  # Increment the step counter
           maximalPoints.pop()  # Pop the top of the stack until it is
empty or the top of the stack has a greater y-coordinate than the current
point
       maximalPoints.append(i)  # Append the current point to the stack

   return maximalPoints, steps  # Return the list of maximal points and
the number of steps taken

ladder_lst, steps = leftRight(input)
print("List of point, moving towards left to right: ",ladder_lst)
print("Steps taken by left to right: ",steps)
```

```
def rightLeft(input):
   maximalPoints = [input[-1]]  # Initialize an empty list to store
maximal points and add the last point in the input list
   steps = 0  # Initialize a counter to keep track of the number of steps
taken
   for i in input[-1::-1]:  # Iterate over each point in the input list in
reverse order
       steps+=1  # Increment the step counter
       if i[0] == maximalPoints[-1][0] and i[1] > maximalPoints[-1][1]:  #
Check if the current point is greater than the second last maximal point
           maximalPoints.insert(-1, i)  # Insert the current point before
the second last maximal point
       if i[1] > maximalPoints[-1][1]:  # Check if the current point is
greater than the last maximal point
           maximalPoints.append(i)  # Append the current point to the list
of maximal points
   return maximalPoints, steps  # Return the list of maximal points and
the number of steps taken
```

```
ladder_lst, steps = rightLeft(input)
print("List of point, moving towards right to left: ",ladder_lst)
print("Steps taken by right to left: ",steps)
```