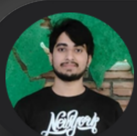
DSA

A MUST KNOW USE OF GCD IF YOU ARE LEARNING DSA



Mayank

01

Slope of Points

You are given an array points[] of n points on cartesian plane

Each points[i] is represented by a pair<int, int> of it's x and y co-ordinate

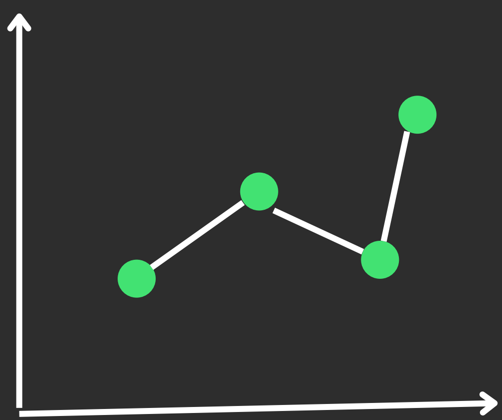
These points are also sorted in order of their x co-ordinate

You need to give minimum count of lines required to represent all these points in form of a histogram

Think! How to approach !!!

02

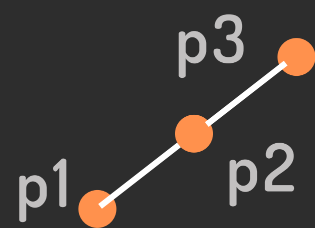
SLOPE = (Y2-Y1)/(X2-X1)



Suppose you are given co-ordinates like in the diagram & we can see we need atleast 3 lines to connect all these points

Suppose there are 3 adjacent points p1, p2, p3 , when can we tell if they lie on a same line or different ??

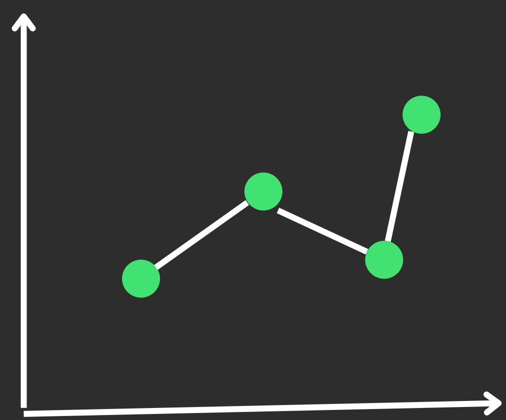
A line is made by connecting 2 points



So if slope of lines connecting p1-p2 and p2-p3 is same then we can say that these 3 points lie on same line

03

SLOPE = (Y2-Y1)/(X2-X1)



Suppose you are given co-ordinates like in the diagram & we can see we need atleast 3 lines to connect all these points

Suppose out of 3 points p1,p2,p3 slope of line p1-p2 comes out to be 15/17

Now 15/17 = 0.8823529.....

Can you see there would be loss of precision even if we use double data type (c++) to store the slope as double can be precise upto 6 digits (by default)

So calculating slope with this formula is not feasable

04

4/5 == 16/20

So our problem is with division as after division if we store the resultant slop we may loose precisions thus we may not get exact slopes

Again consider those 3 points p1, p2, p3

Slope of p1-p2 = 4/5 ((y2-y1)/(x2-x1))

Slope of p2-p3 = 16/20 ((y2-y1)/(x2-x1))

But still you are using division operator... right ??

Let's reduce these slopes in their fraction representation (ignore division)

Slope(p1,p2) = 4/5

Slope(p2,p3) = 16/20 = 4*4/5*4 = 4/5

Slope(p1,p2) = 4/5

Slope(p2,p3) = 4/5

05

4/5 == 16/20

Slope(p1,p2) = 4/5

Slope(p2,p3) = 4/5

After reducing fractions of both slopes they come out to be 4/5 and we can say their slopes are same so lie on a same line

In our previous solution what we did ...

- Maintain a variable 'slope'
- Iterate over all the points and calculate slope of line connecting current point (p2) to previous one (p1)
- let's call it 'currSlope'
- If slope != currSlope which means we need a different line than previous ones

Approach is nice but the problem was with the way we calculated slope i.e. slope = (y2-y1)/(x2-x1)

06

4/5 == 16/20

Slope(p1,p2) = 4/5

Slope(p2,p3) = 16/20 = 4/5

A better solution would be store dx and dy values separately

dx = x2-x1, dy = y2-y1

slope(p1,p2) = 4/5 i.e. dx1 = 4 & dy1 = 5

slope(p2,p3) = 16/20 = 4/5 i.e. dx2 = 4 & dy2 = 5

Now if we consider dx & dy values of 2 lines (p1-p2, p2-p3) separately then
dx1 == dx2 && dy1 == dy2, thus their slopes would be equal

GCD

So the final approach would be

- Maintain 2 variables 'dx' & 'dy'
- Iterate over all the points and calculate curr_dx & curr_dy wrt to line connecting current point & previous one (points[i] with points[i-1])
- if(dx == curr_dx && dy == curr_dy)
- we can say slope of curr line is same as previous, so we do not need a different line (just extend previous one as slopes as same)

Where is use of GCD ???

When you calculate dx & dy as we saw in previous slide, we would need to reduce their fractions, so 16/20 should be 4/5

So here comes GCD, GCD(16,20) = 4, so divide 16 & 20 by 4 reducing them to 4 & 5, now store these reduced 4 & 5

08

A Small Note-

See, there are few other ways as well

Just tried to keep things simple

Main idea was to show you a practical use-case of GCD in DSA problems

Mayank

Software Engineer | StoryTeller

Code Smarter

I hope you found it useful

Follow for content related to DSA | System Design