Polyline Challenge – Fugro

This challenge I have to find the minimum distance of a perpendicular line for a point related to a list of points that is connect (polyline)

I will be explaining what each part of my code does and what each variable means. I will put some graphic images so I can also explain my reasoning.

Methods:

ReadPointsFromCSV – This method is the only method that does not include metathetical logic, this method will receive a CSV file with values speared by comma that will represent points on a Cartesian plane.

The input will be like this:

150,200

100,45

20,-40

-100,75

50,220

125,100

200,150

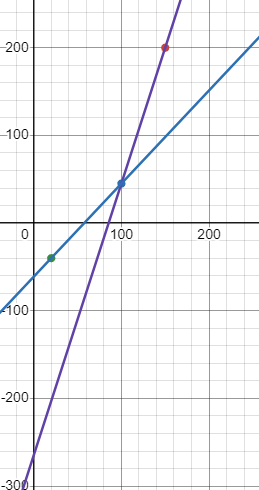
300,175

DefineLine – The method responsible for “drawing” the line for of 2 points. Defines the equation of the line passing through two points, returning the slope(m) and y-intercept(c).

To calculate the slope/angular coefficient(m) we take the 2 points that we are current checking and use their X and Y.

And after having the slope, we use it to find where those points will “cross”/ intersect(c) in the Y axle.

Here is how would look in a Cartesian plane:



IsPointPerpendicularToSegment – This method checks if a point is perpendicular to a line segment, after getting the slope(m) we will try to check if the point that we input are inside the “area” of the two points that we are getting from the polyline.

To check that, we need to invert the result of our slope(m) and calculate the new intersect(c) with our new slope(m), use the equation C = y1 – mx1.

After that we will have two new lines, that will be the opposite of our current ones.

IsPointBetweenLines – This method is used after finding the lines, using the equation of a straight line (y=mx+c), and inverting it.

On this method we will check whether a point is between two lines by comparing the y values ​​for the lines at point x.

The variables ‘x0’ and ‘y0’ will be the values that we input.

We will use ‘x0’ on the equation of a straight line (y=mx+c), with that we will be able to get the ´y1´ and ´y2´ from our lines and see if the ´y0´ is inside of those two new points. If yes, then the input that we did has a perpendicular between those two points of the polyline.

CalculatePerpendicularDistance – We will calculate the distance between the point and the section of polyline that we are working in.

This equation above give us the distance between line-point, on code I broke this equation in 2 parts, the ‘numerator’ and ‘denominator’ to make it easier to look and understand what is going on.

double numerator = Math.Abs((y2 - y1) \* x0 - (x2 - x1) \* y0 + x2 \* y1 - y2 \* x1);

double denominator = Math.Sqrt(Math.Pow(y2 - y1, 2) + Math.Pow(x2 - x1, 2));

return numerator / denominator;

These are all the methods that exist in our program, we still have some logic with the variables 'offset' and 'station', where in 'offset' we will store the value there only when we find a perpendicular, and we will only replace its value with the new one, if the new value is smaller than the one stored. And for the 'station' we will store the distance of each line of the polyline, but we will only put this value in the 'station' when we have a perpendicular, and we will also calculate the distance from the beginning of the session of that line to where the perpendicular meets the point.

RUN APPLICATION

To run the application you can just execute the .exe file, the path for that is …\PolylineChallenge\bin\Debug\net8.0

Inside this path you will run “PolylineChallenge.exe”, when running this you will have to enter the “filePath” to your input.csv.

Something like this: E:\\Download\\input.csv , after that you have to provide 2 values, the ‘x’ and the ‘y’ for your point.

To close the application you can press any key after the message “Finished”