Part III - Testing (30 Points)

Perform white-box testing analysis of the code shown on the next page to generate a set of test cases to thoroughly test the code. Indicate the criteria (e.g. testing methodology) you are using to select your test cases.

- int trilateration(ifstream & infile, ofstream & outfile, double & xmin, double & xmax, double & ymin, double & ymax);

  Reads data from an input file describing a series of triangles, each having a common base plus two other sides that vary from triangle to triangle. The program determines the (x, y) coordinates of the peak of each triangle, assuming the base of the triangle extends from the origin along the positive x axis, and writes those values into an output file. The format of the input file is a single line containing the common base length followed by a series of lines containing two sides per line. The function also determines the minimum and maximum values of x and y calculated, thereby determining the range over which the results are spread. The return value is the number of triangles processed, or a negative error code in the event of errors.

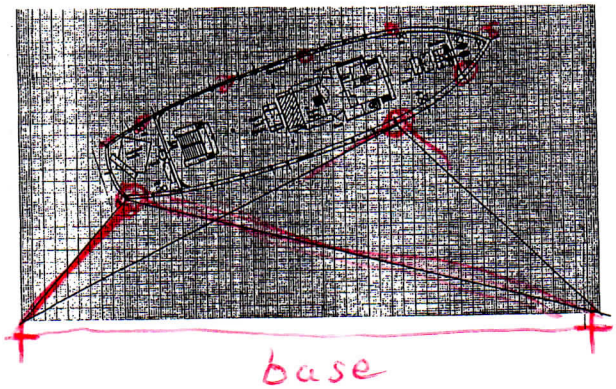
  The mathematics for calculating the (x, y) values are as follows:

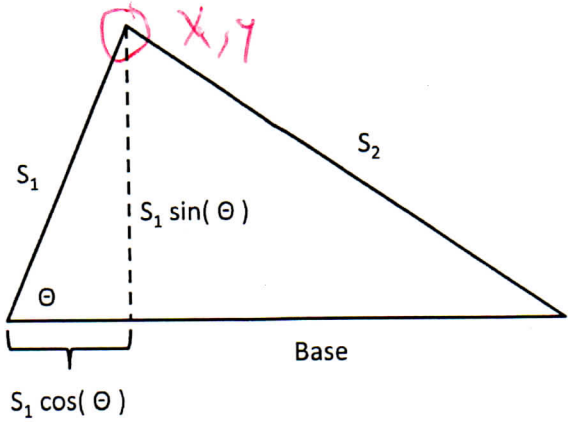
$$\theta = \cos^{-1}\left(\frac{Base^2 + S_1^2 - S_2^2}{2\,Base\,S_1}\right) \quad (1)$$

$$X = S_1 \cos(\theta) \quad (2)$$

$$Y = S_1 \sin(\theta) \quad (3)$$

The code is shown on the following page. The following diagrams will be explained in class:

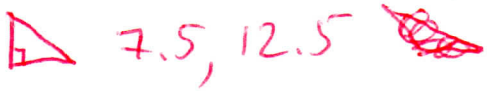




```
int trilateration( ifstream & infile, ofstream & outfile, double & xmin,
  double & xmax, double & ymin, double & ymax ) {

  double base, side1, side2, theta, x, y;
  bool first = true;
  int nTriangles = 0;

  if( !infile.good( ) || !outfile.good( ) )
      return -1;

  // First read in the base length, then loop to read in triangle data

  infile >> base;

  while( infile >> side1 >> side2 ) // Exits loop on end of file
  {
        // Cosine law and basic trigonometry

        theta = acos( ( base * base + side1 * side1 - side2 * side2 )
              / ( 2.0 * base * side1 ) );
        x = side1 * cos( theta );
        y = side1 * sin( theta ) ;

        // Initialize the extrema on the first pass through the loop

        if( first ) {
            xmin = xmax = x;
            ymin = ymax = y;
            first = false;
        }

        // Update the extrema if a new one is found

        xmin = x < xmin ? x : xmin;
        xmax = x > xmax ? x : xmax;
        ymin = y < ymin ? y : ymin;
        ymax = y > ymax ? y : ymax;

        // Output the results to the output file and increase the counter

        outfile << x << "\t" << y << endl;
        nTriangles++;

  } // end of while reading file

  return nTriangles;
}
```

Testing

Start with black box tests, based on functionality and inputs:

Test	Inputs	Expected Results
BB-∅	All valid inputs with valid data in file: Base = 10.0 Sides = 10.0, 10.0  10, 5  5, 10  7.5, 12.5  12.5, 7.5  7.5, 15  15, 7.5 	$X = 5 \quad y = 5\sqrt{3}$ $5 < X < 10 \quad y < 5\sqrt{3}$ opposite of above $X = \emptyset, \ y = 7.5$ (3-4-5 △) $X = 10, \ y = 7.5$ " $X < 0 \quad y < 7.5$ $X > 10 \quad y < 7.5$ return value: 7 xmin < ∅ xmax > 10 ymin, ymax > ∅ and correct
BB-1	Degenerate but valid △s Base = 10  Sides = 3, 7 3, 13 13, 3	$X = 3, \ y = \emptyset$ $X = -3, \ y = \emptyset$ $X = 13, \ y = \emptyset$ return 3 xmin = -3 xmax = 13 ymin = ymax = ∅

Test	Inputs	Results
BB-2	Valid but empty file	No output return < 0 min, max unchanged
BB-3 BB-4	Invalid input/output files	"
BB-5	Valid input file w some valid Δ and some invalid Same data as BB-0, followed by site = 1, 2 10, 10	Output for initial valid Δs up to first bad Δ only return < 0 min, max correct for output results only. Good data after bad ignored
BB-6 BB-7 BB-8 BB-9	Same as BB-5, replacing 1, 2 with: -10, 10 10, -10 20, 5 5, 20	Same as for BB-5, with different negative return codes
BB-10	Base = -10	Same as BB-2, with different return code.

White Box Testing

Examine code to identify additional cases not yet covered;

Test	Inputs	Results
WB-1	valid base, but no sides	Same as BB-2
WB-2	" " , 1 side only	" "
WB-3	" ", some valid sides, last row one side only	Valid results up to last row. Warning.
WB-4	Base = Ø.Ø	Same as BB-2, different result < Ø
WB-5	Base = 10, side = 9 10	X = Ø, Y = Ø
WB-6	Base = 10, Side = 10, Ø	X = 10, Y = Ø
WB-7	First Δ has max Y, Base = 10, Side = 50, 50	X = 5, Y = maxY = $\sqrt{50^2 - 25}$
WB-8,9,10	First Δ has min Y, max X, min X	Correct Results. Also make sure tests above find extrema after 1st Δ.
WB-11	Only 1 triangle	minX = maxX = X " Y = " Y = Y return = 1