One Digital at Nordea

Documentation for triangle classification

Mustafa Al-Dailemi

27th April 2022

Contents

Introduction

This document will describe the methods and thought process throughout the creation of the solution. The problem that is given is as follows:
"Given a triangle, find out if it is scalene, equilateral, isosceles or neither."

1 Problem analysis

This is a classic geometry problem which can be solved using some pre defined mathematical functions that just need the data inserted into the equation. The data input has to be determined before looking into the equations. Then the input has to be changed and the side length and angles have to be calculated. From these calculations it is then possible to determine the type of triangle.

1.1 Data input

The data input could be many things it could be the lengths of a triangle, the angles of a triangle, or just some points. It is also possible to mix some of these data elements to define a triangle. There might possible be more ways to define a triangle but this solution will focus on these three data elements.

The lengths of a triangle can be an simple way to define a triangle, because if the sides of a triangle is given then the angles can be determined from that. But looking at the corner points of a triangle can also tell us everything else, not only the length of the sides but also the angles of the triangle. Moreover it is easier for the user to define the triangle with points or length of the sides.

Mixing both angles and lengths as input will not be a part of this solution since it is deemed out of the scope. Therefore this solution will look into the either having the user input as side lengths or corner points on a coordinate system.

1.2 Equations

The solution will need to have the length of the sides and the angle of the corners.

1.2.1 Calculating the distance

The first calculation that have to be done is the lengths of the sides. If the user gives the lengths then this step can be skipped. Otherwise the lengths have to be calculated using the euclidean distance equation.[2]

$$d(p,q) = \sqrt{\Sigma_{i=1}^{n}(q_i - p_i)^2}$$

Where p and q are the two points, p_i and q_i are vectors starting from the origin space, and n is the n-space.

The n can be replaced with 2 since this solution is working with a two dimensional figure. Therefore we end up with this equation.

$$d(p,q) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2}$$

1.2.2 Calculating the angles

It is possible to calculate the angles of the triangle, because we have calculated the length of the sides. Using the laws of cosine it is possible to calculate the sides. Here is the equation to calculate the angle:

$$\gamma = arccos\left(\frac{a^2 + b^2 - c^2}{2ab}\right)$$

γ is the angle that has to be calculated, a is the lenght of the side to the left of the angle point that is being calculated, b is the length of the side to the right of the angle point that is being calculated, and c is the length of the side that is opposite of the point being calculated.

1.3 Determining the type

There are three types of triangles, equilateral, isosceles, and scalene. These can be seen in Figure 1.1. They can be classified as such. [4]

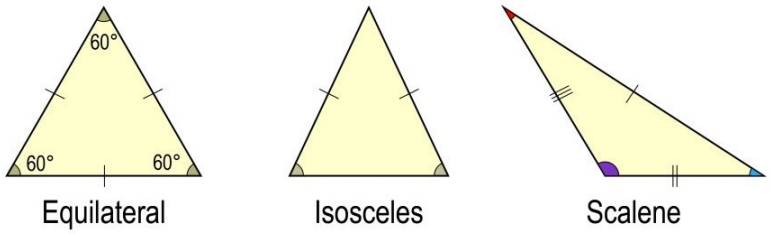

Figure 1.1: Types of triangles

1.3.1 Equilateral

A triangle can be determined to be equilateral if all of its sides are the same length, and all of its angles are the same.

1.3.2 Isosceles

The isosceles triangles only have two sides which have the same length, and the opposing angles of these sides are also the same angles.

1.3.3 Scalene

Scalene does not have any sides that have the same length, and none of its angles are the same either.

1.4 Invalid triangles

Triangles are simple to construct, but sometimes a given measurement can not be used to construct a triangle. This equation can help us determine if a triangle is valid, if it the triangle is within these constraints then it is valid:

$$A + B > C$$

$$A + C > B$$

$$C + B > A$$

Basically the sum of two sides must be larger than the third side. This can be used to determine if a triangle is invalid or valid. [1]

2 solution

The solution will be written in Java. There will be created two classes one smaller and a much larger. The smaller class will contain the coordinates given by the user. The other class is the triangle itself, it will contain all the information and the methods that can calculate the data.

2.1 Coordinate class

The coordinate class contains a constructor and two variables. The two variables are the x and y coordinates. The constructor takes in two integers and saves them to the variables.

```
1 public class Corner {
2     public int x;
3     public int y;
4
5     Corner(int inx, int iny) {
6         x = inx;
7         y = iny;
8     }
9 }
```

Listing 2.1: Corner class

2.2 Triangle class

The triangle class contains all of the information about the triangle as well as the functions that does the calculations of the different parts of the triangle.

2.2.1 Variables

The triangle function has variables containing information about the triangle. The variables are the angles of the triangle and the lengths of the sides. The other variable is an enum which is the type of the triangle.

```
1 private double length_A, length_B, length_C,
2                    angle_A, angle_B, angle_C;
3
4 private type triangle_type;
5
6 enum type {
7     EQUILATERAL,
8     ISOSCELES,
9     SCALENE,
```

```
10      INVALID
11  }
```

Listing 2.2: Triangle class variables

2.2.2 Constructor and setup

The user is prompted first prompted if they have the length of all the sides for the triangle. This information is used to determine what type of input the program will expect from the user. Then the program will prompt the user accordingly. The coordinate prompt can be seen in Listing 2.3 this is prompted if the user do not know the length of all the sides. Then it calls the triangleSetup function to finish the setup. If the user knew the lengths then it would be the same but instead of calculating the lengths it would simply be inserted into the variables.

```
1  private void triangleSetup(){
2      check_validity();
3
4      this.angle_A = calc_angle(this.length_B, this.length_C, this.length_A);
5      this.angle_B = calc_angle(this.length_C, this.length_A, this.length_B);
6      this.angle_C = calc_angle(this.length_A, this.length_B, this.length_C);
7
8      determine_type();
9  }
10
11 public void prompt_coordinates(){
12     String input;
13     String[] strArray = new String[6];
14     Scanner sc = new Scanner(System.in);
15     boolean no_answer = true;
16
17
18     while(no_answer){
19         System.out.println("Enter the three coordinates coordinate x,y x,y x,
               y");
20         input = sc.nextLine();
21         strArray = input.split(",|\\s+|\\(|\\)");
22
23         if(strArray.length != 6){
24             System.out.println("Something went wrong :/");
25         } else {
26             no_answer = false;
27         }
28     }
29
30     Corner[] corner_points = new Corner[3];
31
32     corner_points[0] = new Corner(Integer.parseInt(strArray[0]),
33                                   Integer.parseInt(strArray[1]));
34     corner_points[1] = new Corner(Integer.parseInt(strArray[2]),
35                                   Integer.parseInt(strArray[3]));
36     corner_points[2] = new Corner(Integer.parseInt(strArray[4]),
37                                   Integer.parseInt(strArray[5]));
38
39     this.length_A = calc_length(corner_points[1], corner_points[2]);
40     this.length_B = calc_length(corner_points[0], corner_points[2]);
```

```
41      this.length_C = calc_length(corner_points[0], corner_points[1]);
42
43      this.triangleSetup();
44  }
```

Listing 2.3: Setup of the triangle class

2.2.3 Calculate the length

The length is calculated only if the user has entered coordinates. This step is skipped if the user have entered the length of the sides. This functions calculates the length using the euclidean distance equation, which was mentioned in subsection 1.2.1

```
1  private double calc_length(Corner first, Corner second){
2      double result, x_side, y_side;
3
4      x_side = second.x - first.x;
5
6      x_side = Math.pow(x_side ,2);
7
8      y_side = second.y - first.y;
9
10     y_side = Math.pow(y_side ,2);
11
12     result = Math.sqrt(x_side + y_side);
13
14     return result;
15 }
```

Listing 2.4: Calculation of the lengths

The variable *x_side* is the first parentheses in the square root of the equation, and *y_side* is the other parentheses in the square root of the equation. They are then added to each other and the square root is taken and the result is returned by the function.

2.2.4 Calculate the angle

The angles are always calculated regardless of the user input. They are used later for verification to determine the type of the triangle.

```
1  private double calc_angle(double l_len, double r_len, double opp_len){
2      double result, num, den;
3
4      num = Math.pow(l_len, 2) + Math.pow(r_len, 2) - Math.pow(opp_len, 2);
5
6      den = 2 * l_len * r_len;
7
8      result = num / den;
9
10     result = Math.acos(result);
11
12     result = result * (180/Math.PI);
13
```

```
14      return result;
15  }
```

Listing 2.5: Calculation of the angles

The variable *num* is the numerator of the equation, and *den* is the denominator. They are then divided with their respective placements on the division, then the arccosine is taken of the result. This gives us the angle in radiants, but the result should be in degrees. According to this documentation [3] this equation $rad * (180/\pi)$ can be used to convert from radian to degree. Therefore it is done before returning the result.

2.2.5 Determine type

To determine the type of the triangle we can either use the lengths of the sides, or the angles. This solution will use both methods, to make sure that the result is always right.

```
1  private void determine_type(){
2      if(this.triangle_type == type.INVALID){
3          return;
4      }
5
6      if((this.length_A == this.length_B) && (this.length_B == this.length_C)
           &&
7         (this.angle_A == this.angle_B) && (this.angle_B == this.angle_C)){
8          this.triangle_type = type.EQUILATERAL;
9          return;
10     }
11
12     if((this.length_A == this.length_B && this.angle_A == this.angle_B) ||
13        (this.length_A == this.length_C && this.angle_A == this.angle_C) ||
14        (this.length_B == this.length_C && this.angle_B == this.angle_C)){
15         this.triangle_type = type.ISOSCELES;
16         return;
17     }
18
19     this.triangle_type = type.SCALENE;
20 }
```

Listing 2.6: Determine the type of the triangle

The first lines check if the triangle is even valid. If a triangle is invalid then there is no need for further calculations.

Then the function checks if it is an equilateral triangle, to do this all sides must be the same length, and all angles must also be the same. Because of transitivity, we only need to check if a == b and b ==c, because if these are true then they imply that a == c. The same goes for the angles.

The function then checks if it is an isosceles triangle, but only if it is not an equilateral triangle. Here only two sides must be the same length, but their opposing angles must also be the same.

If it is not an isosceles triangle either, then it must be a scalene triangle.

2.2.6 Check validity

The validity of the triangle is checked by looking at the length of the sides. It is checked by looking at the length of the sides. If one side is longer than the sum of the two others then it is invalid.

```
1 private void check_validity(){
2     if(this.length_A + this.length_B <= this.length_C ||
3        this.length_A + this.length_C <= this.length_B ||
4        this.length_B + this.length_C <= this.length_A){
5         this.triangle_type = type.INVALID;
6     }
7 }
```

Listing 2.7: Check validity

The function goes through and checks all the possible scenarios. If it is valid nothing is done. The triangle is marked invalid if fails the test.

Bibliography

[1] Check whether triangle is valid or not if sides are given. https://www.geeksforgeeks.org/check-whether-triangle-valid-not-sides-given/, Apr 2022.

[2] Euclidean distance. https://en.wikipedia.org/wiki/Euclidean_distance, Apr 2022.

[3] Radian. https://en.wikipedia.org/wiki/Radian, Apr 2022.

[4] Ashutosh. Properties of triangle - types amp; formulas. https://e-gmat.com/blogs/properties-of-triangles/, Nov 2021.