CIS613 - Assignment01 - Ahsan - W26

My Test cases check:

- for a valid equilateral triangle.
- for valid scalene triangles. Used P(3, 3) to check for any ordering anomaly.
- for valid isosceles triangles. Used P(3, 3)/2! to check for any ordering anomaly.
- if inputting all negative side lengths returns an invalid triangle.
- if inputting two negative side lengths returns an invalid triangle. Used P(3, 3)/2! to check for any ordering anomaly.
- if inputting a negative side length returns an invalid triangle. Used P(3, 3)/2! to check for any ordering anomaly.
- if inputting all zero side lengths returns an invalid triangle.
- if inputting two zero side lengths returns an invalid triangle. Used P(3, 3)/2! to check for any ordering anomaly.
- if inputting a zero side length returns an invalid triangle. Used P(3, 3)/2! to check for any ordering anomaly.
- if inputting invalid side lengths returns an invalid triangle. Used $3a = 2a + a$, $3a > a + a$ formulas.
- if inputting valid side lengths on the boundary returns a valid triangle. Used $3a < 2a + (a + \text{delta})$ where delta is the minimum possible 1.
- if inputting valid side lengths on integer boundary returns a valid triangle. This valid case fails.

Usefulness

I covered the happy paths first. Then covered the invalid triangle possibilities with negative and zero lengths. Next, I tried the invalid side lengths with positive numbers. This led me to test with values on the boundary. Next, I tested with the programming language’s boundary value for an integer. This revealed that the code is error-prone to integer overflow.