**Experiment Report - 54 - test1\_TriangleValidation**

1. **Summary Table of Errors Found**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Error ID | Line Number | Error Type | Self-Detected? | Peer 1 Found? | Peer 2 Found? |
| E01 | line 17 | Syntax | √ | × | × |
| E02 | line 35 | Semantic | √ | √ | √ |
| E03 | line 49 | Semantic | √ | × | × |

Additional Errors Found by Self: 0

Self-Review Detection Rate: 100%

Peer 1 Detection Rate: 33%

Peer 2 Detection Rate: 33%

1. **Source Code**
2. package a;
3. import com.code\_intelligence.jazzer.api.FuzzedDataProvider;
4. import code.c01\_TriangleValidation;
5. public class c01\_TriangleValidation {
6. public static boolean isValidTriangle(double a, double b, double c) {
7. boolean flag;
8. if((a > 0) && (b > 0) && (c > 0)
9. && (a + b > c)
10. && (a + c > b)
11. && (b + c > a)) {
12. flag = true;
13. }
14. else flag = false;
15. return;
16. }
17. public static void main(String[] args) {
18. // testcase-VT(vibration testing):
19. double[][] testCases = {
20. {1.7, 2.845, 0.99}, //a > 0, b > 0, c > 0, a + b - c > 0, b + c - a > 0, a + c - b <= 0
21. {5.3284, 0.712, 6.9703}, //a > 0, b > 0, c > 0, a + b - c <= 0, b + c - a > 0, a + c - b > 0
22. {4.73336, 19.0923, 15.002}, //a > 0, b > 0, c > 0, a + c - b > 0, a + b - c > 0, b + c - a > 0
23. {0.179, 6.90, 7.779}, //a > 0, b > 0, c > 0, a + b - c <= 0, b + c - a > 0, a + c - b > 0
24. {18, 1.3413, 13.544},//a > 0, b > 0, c > 0, a + b - c > 0, b + c - a <= 0, a + c - b > 0
25. {-16.6171, 6.9286, 12.4}, // a <= 0
26. {0.323, -1, 3.223}, // b <= 0
27. {4.73336, 19.0923, -10.643}, // c <= 0
28. };
30. for (double[] testCase : testCases) {
31. boolean result = isValidTriangle(testCase[0], testCase[1], testCase[2]);
32. System.out.printf("isValidTriangle(%.1f, %.1f, %.1) = %b%n",
33. testCase[0], testCase[1], testCase[2], result);
34. }
35. }
36. }
37. class t01 {
38. public static void fuzzerTestOneInput(FuzzedDataProvider data) {
39. // generate 3 random double value
40. double a = data.consumeDouble();
41. double b = data.consumeDouble();
42. double c = data.consumeDouble();
43. // call target method
44. boolean result = C01\_TriangleValidation.isValidTriangle(a, b, c);
45. // System.out.println("Testing with: a=" + a + ", b=" + b + ", c=" + c + " => ");
46. System.out.println("Testing with: a=" + a + ", b=" + b + ", c=" + c + " => Result: " + result);
47. }
48. }