Here's a unit test suite for the unit test in Python. This suite includes tests for different scenarios as described:

**Invalid Triangle Test:** Determines whether the function successfully recognizes combinations of sides that do not form a valid triangle. The triangle type will only be invalid if the summation of two sides is less than the other. This test is important to check if the code captures the invalid triangle value if given.

**Equilateral Triangle Test:** A triangle will be equilateral if its three sides are equal. This test determines whether the function correctly recognizes triangles with all sides equal.

**Isosceles Triangle Test:** A triangle will be isosceles if any of its two sides are equal. This ensures that the function correctly identifies triangles with two equal sides.

**Scalene Triangle Test:** Tests for triangles with varied sides. This test will give input with three different integers that makes a valid triangle and also no sides are equal.

**Edge Cases Test:** Tests for edge cases, particularly where the sum of two sides equals the third, which should be invalid.

**Zero and Negative Sides Test:** No sides of a triangle can never be zero or negative. It makes an invalid triangle. This test checks whether the function handles scenarios with zero and negative values for sides, which should be flagged as invalid.

**Large Value Sides Test:** Ensures that the function operates properly with extremely big values for sides.

**Floating Point Sides Test:** A side of a triangle can be floating value too and the code should handle the floating value. This test verifies the function's validity using floating-point numbers for sides.

As a whole, these tests assure the unit test's resilience and dependability. The unittest framework will run each test and provide the results.