Question 1 of 1

Method 1 1 pt

Suppose that the following test fails in the buggy version:

```
public void testAddNaN() {
        // x represents a complex number (복소수 in Korean), 3.0 + 4.0i
        Complex x = new Complex(3.0, 4.0);

        // z represents a complex number (복소수 in Korean), 1+ (NaN)i
        Complex z = new Complex(1, Double.NaN);

        // Perform w = x + z
        // Note that (3.0 + 4.0i) + (1+ (NaN)i) = NaN + (NaN)i
        Complex w = x.add(z);

        // w.getReal() returns the real part (실수부) of w
        // w.getImaginary() returns the imaginary part (허수부) of w
        assertTrue(Double.isNaN(w.getReal()) && Double.isNaN(w.getImaginary()));
}
```

The above test checks whether the addition between two complex numbers (복소수 in Korean) is performed correctly.

It turns out that the current buggy implementation produces an incorrect output when NaN is involved in the addition, and the above test fails because of this reason. When NaN is not involved, the addition performs correctly in the current implementation. That is, the following property is satisfied for two complex numbers, (a + bi) and (c + di) where a, b, c, and d are real numbers:

(a + bi) + (c + di) = (a+c) + (b+d)i

Now suppose that we have a patch for this bug, and we are going to use the following test to validate this patch. Complete this test by filling in the underlined blank.

```
public void testAddNaN(double a, double b, double c, double d) {
        Complex x = new Complex(a, b);
        Complex z = new Complex(c, d);
        Complex w = x.add(z);

        // Fill in the following blank with a boolean expression.
        // Note that the following condition will become false, if the patch is incorrect.
        assertTrue(________________________________);
   }
```