Question 1 of 1

Method 1 1 pt

Suppose that the following test fails in the buggy version:

```
public void testSSENonNegative() {

   // Data points used for regression
   double[] x = { 1.107178495E2, 1.107264895E2, 1.107351295E2 };
   double[] y = { 8915.102, 8919.302, 8923.502 };

   // With SimpleRegression, we perform a linear regression.
   SimpleRegression reg = new SimpleRegression();

   // We find a linear regression model that fits the data best.
   for (int i = 0; i < x.length; i++) {
       reg.addData(x[i], y[i]);
   }

    // The following assertion fails in the buggy version.
    assertTrue(reg.getSumSquaredErrors() >= 0.0);
}
```

In the above test, getSumSquaredErrors() returns the sum of squared errors (SSE) of the obtained linear regression model. Note that SSE can be calculated using the following formula:

$$\sum_{i=0}^{n-1} (y[i] - f(x[i]))^2$$

In the above formula, x and y correspond to the array x and y of the failing test testSSENonNegative. For example, x[0] is 1.107178495E2 and y[0] is 8915.102. Meanwhile, notation f refers to the obtained regression model. Note that in the implementation, f(x[i]) can be obtained through reg.predict(x[i]) where reg refers to a SimpleRegression object.

SSE should always be non-negative, and the current buggy implementation fails to satisfy this condition for the given test.

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 testSSENonNegative(double d1, double d2, double d3,
   double d4, double d5, double d6) {

   // Data points used for regression
   double[] x = { d1, d2, d3 };
   double[] y = { d4, d5, d6 };

   SimpleRegression reg = new SimpleRegression();
   for (int i = 0; i < x.length; i++) {
       reg.addData(x[i], y[i]);
   }

   final double sse = reg.getSumSquaredErrors();

   // Fill in the following blank with a boolean expression.
   // Note that the following condition will become false, if the patch is incorrect.
   // Also note that the original assertion condition, reg.getSumSquaredErrors() >= 0.0,
   // is not strong enough since an incorrect patch can also satisfy
   // reg.getSumSquaredErrors() >= 0.0.
   assertTrue(______________________________);
}
```