Questions

Bookmark this page

debugging strategy

5/5 points (graded)

Suppose you are debugging the quadraticRoots function, which appears to be producing wrong answers sometimes.

```
/**
 * Solves quadratic equation ax^2 + bx + c = 0.
 *
 * @param a quadratic coefficient, requires a != 0
 * @param b linear coefficient
 * @param c constant term
 * @return a list of the real roots of the equation
 */
public static List<Double> quadraticRoots(int a, int b, int c) { ... }
```

Put the following items in the order that you should try them: 1, 2, 3, ... Say "wat" for items that are nonsense statements.

Change your code from using ArrayList to using LinkedList.

4 ✔ **Answer:** 4

Put println() statements throughout your method to display the intermediate values of the calculation.

3 ✔ **Answer:** 3

Write a test case that causes the bug to happen.

1 ✔ **Answer:** 1

Run a code coverage tool to see if there are lines that your tests aren't reaching.

2 ✔ **Answer:** 2

Switch from Java 8 back to Java 7.

5 ✔ **Answer:** 5

Explanation
Having a failing test case makes it much easier to debug and is always the best thing to start with.
Running the code coverage tool is not a bad next step because it's easy to run and may help localize the bug to untested code.
Inserting print statements to collect information takes more effort but the effort pays off.
Changing from ArrayList to LinkedList, or Java 8 to Java 7, are good examples of the "swap components" technique, but these are trusted components that are supposed to behave alike, so they would be the last things to try.

Submit Show Answer

Answers are displayed within the problem