Liz Brondos

Super Tester to the Rescue!

Programmer: I just spent all day working on a great new program, and I'm trying to run it, but it doesn't work! Oh, what am I to do? Help! Help!
(Super Tester, the superhero, flies in.)
Super Tester: Did I hear someone calling me?
Programmer: Oh, Super Tester, thank goodness you're here! I'm trying to run this program but the computer just beeps at me!
Super Tester: That's because you have a bunch of syntax errors! Check your code; you may have left out some semicolons or are making other Java grammatical errors that the computer doesn't like.
Programmer: OK, I'll do that. *(He fixes the errors and tries to run the program again.)* Well, it works now, but it doesn't give me the result I want! What's wrong now?
Super Tester: Uh-oh, it looks like you have a logical error. Logical errors are harder to find than syntax errors, and they are not easy to detect.
Programmer: Oh, what am I to do? Is there any hope for me at all?
Super Tester: Why yes, you're in luck. You can use testing! Testing is the activity of finding out whether a piece of code (whether it's a method, a class, or a program) behaves the way you want it to.
Programmer: So, once I find the error, what do I do?
Super Tester: Once tests have shown that an error is present, debugging techniques can be used to find where the error is and attempt to fix it.
Programmer: How do I go about testing my program? Should I test a little bit at a time, or should I test the whole program?
Super Tester: Well, it's up to you. You can either do unit testing or application testing. Unit testing is testing individual parts of an application, while application testing is testing of the complete application.
Programmer: So all I do is test to make sure everything works correctly?
Super Tester: One common mistake of many beginners is that they conduct only positive tests, which means that they only test cases that are expected to succeed. But you also need to do negative testing and test cases that are expected to fail. Otherwise, for example, you could end up with a program that tells you that an empty string is a palindrome when it's really not.
Programmer: What happens if I'm sitting at the computer for a really long time and can't find any mistakes?
Super Tester: Well, you can do a manual walkthrough! That means that you work through a segment of code line by line while you observe changes of state and other behavior of the application. In other words, you pretend to be the computer that is executing the code. Another method of testing is to annotate methods temporarily with print statements. This way, you can check the status of your objects at certain points in the program. The advantage of print statements is that it makes it easier to find out where in your program you might have gone wrong. But the problem is that they can be a pain to add and remove, and adding too many of them can overload you with too much information.

Programmer: Hurray, my program is not doomed! I will start testing right away. Thank you for your help, Super Tester!
Super Tester: No problem! Feel free to call me again if you need more help. And remember to eat your vegetables so you can be big and strong like me! *(Flies away.)*