What to do if your code isn’t working...
(The Beginner’s Guide to Debugging)

The Debugging Process:

1. **Don’t panic!**

2. Figure out **how** your code is breaking. What is it doing that it’s not supposed to be doing? Or conversely, what is it NOT doing that it should be doing?

3. Figure out **where** your code is breaking -- *isolate the problem!* Which line/block/method is not behaving the way it’s supposed to behave?

4. Figure out **why** your code is behaving this way. What possible scenarios could cause this particular problem to happen? How can you change your code to fix those scenarios?

5. Still stuck? Maybe your code isn’t actually broken. **Double check your tests** to make sure it’s not actually those that are breaking your program.

---

Step 2: **How**
Sometimes programs don’t behave the way you think they’ll behave. So how do you figure out what your code is doing? Well, first you need to understand at a conceptual level what’s going on. Here are some ways to understand your code:

- Explain to a friend/classmate/random person off the street what your code is doing.
- Comment every line/block in your code, explaining what it does.
- Read over your guidelines again, paying careful attention to both general concepts and specific details. Make sure that your code’s functionality matches what the prompt asks.

---

Step 3: **Where**
Perhaps you think your logic is correct, but for some reason, your code doesn’t work and you don’t know where the problem is. How do you find that problem? Here are some things you can try:

- Compile error? First, read the error. It may look cryptic and scary, but often it will pinpoint the exact line where the problem is. However, sometimes the line number
that the compiler gives you isn’t actually where the error is, especially if it has to do
with curly braces. Try looking in the lines immediately above or below the compile
error.

● Got an exception? The console usually tells you which file broke and at which line. For
example, consider the following output:

```java
java.lang.ArrayIndexOutOfBoundsException: -1
    at SomeClass.someMethod(SomeClass.java: 10)
```

This actually tells you that your program broke in a method called “someMethod” in
the class, SomeClass, at line 10.

● Start off by putting a print statement where you know your code should work. Run
your program to make sure it actually prints out. If it does, move the print statement
down one line. Run the program again to make sure it prints out. Rinse, lather,
repeat. If at some point in this process, you put your print statement in a code
block/if statement/loop that your program is supposed to enter, but it doesn’t print
out, then congratulations! You have found the line that’s breaking your code!

● Use print statements to print out what is being stored in your variables. Is what’s
being stored there actually what you think it will be? Or do they end up being null or 0
or some other unexpected value?

● Always backup your code! Make multiple copies of your files, so that you have
previous versions that work that you can refer back to if something breaks. This way
you can see what you’ve added since your last version of your program and where the
new problem may be occurring.

---

**Step 4: Why**

Now you know exactly which part of your code is breaking. What values could possibly make
it behave in this way? Here are some common errors and ways to approach them:

- **NullPointerExceptions** - These typically happen when you are trying to access a
  method or member of an object that is null. Have you instantiated your object? If so,
  is it actually reaching the part of your code that’s supposed to instantiate it? Did you
  instantiate an instance variable as a local variable?

- **IndexOutOfBoundsException** - This typically happens when you have a loop and you
  accidentally try to access a value at an index out of range. Remember to check your
  starting index and ending index (plug in some values!). Array indexing starts at 0, and
  ends at n-1 (i < n).
- Boolean expression not evaluating the way you want to - This might be because you are using the == operator (which compares object references) to compare an object instead of the .equals method.

- If-else statements not working correctly? Check that you have compared two variables using the == (equals) operator and NOT the = (assignment) operator.