Lab 7
Debugging

10 October 2019

Your "drill" task this week involves reading a function I have written and seeded with bugs, and doing some preliminary analysis on it.

1. First, copy this drill version into your working directory for this lab. In your working directory, type

   ```
   cp /home/shared/160/countDivisors-drill.cpp .
   ```

   (don't forget the lonely dot at the end to put it in the current directory).

2. It has several errors! Read through the file, but don't fix them yet.

3. You'll be tracing the code using a value of 12 for `numToCheck`. Before you do so, write down what you think the correct return value should be (and why).

4. By hand, trace the code using a value of 12 for `numToCheck`. Anytime you encounter a bug:

   - If it is an error that would generate a compiler error with a clear fix (e.g. misspelled identifier), "fix" the eror in your head and keep going with the trace.
   - If it is an error that would generate a compiler error with multiple syntactically valid fixes (e.g. unattached else), pick one such fix, make a note of what you chose, and keep going with the trace.
   - If it is an error that would crash the program at that point, stop the trace and read the next drill step before continuing.
   - If it is an error that you happen to spot, that you know would generate a wrong final answer but would let the program continue from that point, *don't* fix it yet, treat it as-is, and keep going with the trace.

5. Anytime you get to a point in your trace where either the function finishes or the program would crash: make a note of it; make a guess

at how to fix it (and write down the guess); and then start a *fresh* trace with that fix in place. Don't just restart the trace in the middle with the changed lines!

6. If you get through two or three traces and still have not fixed everything, that's sufficient for now; there will be more on this to come later in the lab.