Kevin Turkington

6/29/16

CS161

Tim Alcon

Assignment 3 (reflection)

- **Understanding**: What did you learn about the problem as you went? Why or how did you learn it?

  I learned that planning makes for clean code. And sometimes redoing some code can drastically make what you planned out to be simpler and smaller.

- **Testing plan**: What tests didn't work out the way you expected? What alterations did you have to make to your program due to failed tests? How could your planned tests have been more complete?

  All tests workout as I expected even the overflow errors. If I were to add more tests it would be for floats, accidental letter characters, and spaces in the inputs.

- **Design**: What was missing or needed to be altered from your initial design, and why?

  For the minmax problem I needed a special case when the first integer is set, it should be both the min and max value at the same time. Originally I wanted to set both min and max to NULL and have an if statement indicating if both min and max where NULL then they would be set to the first integer. However, in C++ integers being set to null is not supported, so I used a workaround from a previous assignment involving a Boolean IF statement that would no longer be true after the first iteration. If this wasn't created some cases would have a min or max or null.

- **Implementation**: What problems did you encounter during implementation? How were you able to solve those problems? What outside sources (sites, books, or other materials) did you find helpful?

  The only problem I had was remembering how to pass in a file correctly to a program, that would indicate if it was properly found/opened. I looked on the Cplusplus forum for a simple snippet of code that showed the proper syntax to do what I wanted. In addition to the eof (end of file) function which helped with shortening my program significantly.

- **Improvement**: How can you generalize any parts of your problem solving experience in a way that might help you on future assignments?

Planning out code on paper is extremely helpful, and sometimes writing different way to complete the task can result in more compact programs that will be easy for others to understand.