Name: Benjamin R. Olson
Date: April 26, 2014
Course: CS162 - Introduction to Computer Science II
Instructor: Terry Rooker

Project 2 Reflection

Thoughts On Design:

As I was designing, what I found interesting about this project was that most of my design efforts went into considering how to structure the data, because the functions that need to be written depend heavily on this.

One thing I really focused on during the design was keeping the user input functionality separate from the class, and as it turned out during implementation, I was glad I did this because I wrote about same amount of code for prompting and screening user input as I did for storing, changing, and accessing the data.

Changes To Design During Implementation:

I decided not to use three high level functions that each wrap around a few other functions because each of them would only be one to a few lines of code and called in only one place; I thought writing these functions would make the code unnecessarily complex.

Thoughts On Testing:

The testing for the class was better planned out than the testing for the main control flow of the program. I documented the class tests and put more effort into those initial tests, but once I was confident in the class functionality, my testing methodology for the rest of the code was much more loose, relying mainly on running the program itself to see if behaves as expected. Although the latter phase of testing was much less structured, it revealed just as much about where I could make improvements in my code. I quickly realized while test-driving the program that I needed to make many improvements to the program's usability.

Problems Encountered, Lessons Learned:

While I was working with the object, I realized that the struct type I had created within the object's class, although public, was not part of the outside world. I was thinking about writing a function in main.cpp to display all the items but as I was trying to do this, I realize that because the struct type was unique to the class in which it was created, it was much easier to add a function to the class itself – a function to display each item struct (element) of each items vector within the class.

What I did not consider during design but found very helpful during implementation was to add a pointer to a vector. I also got some good practice accessing elements of a vector, reinforcing what I should know – that the vector itself is basically a pointer. So I found myself using the dereferencing syntax “->” on a vector frequently.

What Can Be Improved:

What might reduce the amount of code needed and make the program more adaptable would be to add a large vector that holds each smaller location vector, or to change the data model so that there is simply one vector where each struct itself contains information on its store location. I might also make it more user-friendly by allowing the user to enter everything about a item in a single line, and just parse the input into different variables, even though this would require much more sophisticated input validation and one or more functions to parse the input line.

How This Program Could Be Expanded:

One of the greatest strengths of this program is that the way the data is structured allows for each shopping list item's characteristics to be expanded by adding to the struct definition.