**Reflection on the Final Project**

Daniel Schmidt

CPSC 60000-001

Professor: Dr. Michael Nowak

12 October 2020

**Challenge One: Multiple Patterns**

Understanding and implementing one pattern was easy to do. After reading through the book, looking over the examples and then playing with the code made this process straightforward and simple. I started the final project early and implemented single patterns immediately. I started with the decorator pattern and then moved onto the factory pattern. I didn’t have any trouble with this. I started to run into trouble when I had to implement more than two patterns.

After the decorator and factory pattern I attempted to move onto the command pattern and ran into many issues. The first problem was I had no plan in place to implement the command pattern. I wrote the program and it worked well with just the decorator and factory pattern and I found myself attempting to force the command pattern into the code. I moved onto the iterator pattern and had a little bit more success but again since I had an already working program and no plan to implement the pattern.

I learned much while struggling with multiple patterns. The most valuable thing I learned was not to program one pattern into the code at a time. I painted myself into a corner with each pattern I did and implementing another pattern within the existing code proved difficult. I also learned to come up with a plan for all the patterns up front and have them work together, not act as individual pieces that need to be forced to work together.

**Challenge Two: GUI**

The second challenge I faced was building the graphical user interface. I have used Java before but never programmed a GUI with it. This proved time consuming and difficult. Getting the buttons and text to look and work correctly was one of the hardest parts of the project. After much tinkering I got the GUI to look somewhat normal and the buttons to do what they needed to do. This involved at lot of Google and Github.

The first thing I learned was GUI’s require an entirely different skill set. Even when you can get it to look somewhat similar to what you want it looks no where close to appealing. I realize now why projects are often broken into a front end and a back end. The second thing I learned was after the initial frustration and tinkering the GUI is easy to manipulate. I have to admit after I got the hang of the GUI I could at lease get it to do what I wanted it to.

**Conclusion**

My biggest challenge was incorporating multiple patterns within the same program and programming the GUI.  I ended up with a GUI that worked and did what I intended but it looked terrible and cause me plenty of headaches along the way.  I now know why projects are often split between front and back end, the presentation requires an entirely different skillset.  The first problem of incorporating multiple patterns was my own doing.  I took one pattern and implemented it, then tried to force another one on top of it and tried to do the same with the third.  I didn’t have a plan for all the patterns going into it and therefore had to unnaturally force the patterns into the code.  I have a much better understanding of not using a pattern just to use it.  If it makes sense to use a pattern then it should be used but it should not be used just to use it. I didn’t have too many surprises along the way but looking back I would have had a plan to incorporate all four patterns into the design before starting.  Just having a simple concept of how I would do that would have helped immensely.  I might have also not attempted a GUI and stuck with print statements.  Figuring out the GUI ate up a significant amount of time and took away from the learning experience. This was not a GUI class, it was an object oriented class and therefore the GUI should have taken the back seat or not been worried about at all. I didn’t learn too much about my work habits but learned my planning process may need to incorporate more planning up front to address the issues already discussed. I need to figure out the big picture first before moving on to implementation.