Kyle Zalewski

CS202

Project 4 Efficiency

For Programming Assignment 4, by far the biggest difficulty was the data structure. I find Java to be a much more intuitive language to code with, so I “saved the best for last” by using a standard binary search tree in Programming Assignment 3. The IDE helped immensely with saving time troubleshooting, however.

As for efficiency, I have successfully eliminated the Node class from my program, which was a goal of mine since the beginning of the term. Without the Node class, I essentially treated my abstract base class as my “node” and was able to do everything I formerly did with a node class, but with less of a need for setters and getters (at the very least, less calls thereto). Since it was abstract, I had overridden functions for display in each of the derived classes, allowing me to have outputs that were specifically tailored to each type of object.

An interesting issue I ran into was that I was able to directly access members of a different class. Specifically, I was able to access left and right pointers in the Restaurant class from the RBT class. I found this to be strange and not object-oriented, so I changed my code to call get functions instead. It wasn’t exactly perfect, but I felt that was better than directly accessing protected data members from an outside class. This coupled with the use of function overloading for constructors (with arguments) helped me to keep the entire program more intuitive as well as object-oriented.

I much prefer working with the class files in the IDE to the header and implementation file separation we have been using in C++. Being able to simply begin designing a new method right there in-line in a class helps tremendously with the workflow, and it’s easier to rearrange functions after the fact for readability. Since the declaration and implementation are both in one spot in the code, moving and changing them in the future (adding more passed in variables, etc.) is much simpler than having to move and edit it more than one place (file).

Another brilliant advantage to the IDE is the ability to graphically click through a data structure to open it up and ensure that the pointers (references) are in the right locations. It’s a far sight more efficient when troubleshooting data structure code to be able to do that versus the need for endless chained function calls or steps that GDB tends to require on the command line (or in the text user interface, for that matter). With the IdeaVim extension that allows virtually all common Vim commands to be used in the IDE, I can think of no real disadvantage to this coding environment versus the Linux environment. I feel I can still competently code in either, but if given the choice, I think I would choose the IDE ten times out of ten.