Kyle Zalewski

CS202

Object Oriented Design

In Programming Assignment 3, I was finally able to effectively implement a tree structure without having a separate node class. I instead made my base class the “node” and included pointers for left and right in each object of that class. It was much simpler than my mind was making it at first, and my end result was much more readable code. I will continue with this design in Programming Assignments 4 and 5, as well as experiment with ways to even further limit, or possibly completely eliminate, the use of setters and getters. There were a couple of times in older designs where I realized that some setter I was considering using was completely unnecessary because I had no node class anymore. I aim to further develop my skills in this area and hopefully refine the design to eliminate these types of functions entirely.

As my Assignment 3 used a binary search tree, this one will use a Red-Black tree. I hope to practice implementing this data structure to the point where I am fluent, as it seems like a good way to have an elegant self-balancing tree with very little extra overhead versus a standard binary search tree. Since this program will essentially be a tree of trees, the correct implementation of that structure will be all the more important.

The base class “Restaurant” will be abstract. There would be no reason to look into just “Restaurant” types, so I will only instantiate objects with a specific type of food. The three derived classes from “Restaurant” will be “Chinese,” “Mexican,” and “Vegetarian.” This will represent the three types of catalogued meal services my app will support. Any Restaurant object will have one of these three specializations, as well as the traits of the abstract base class.

The abstract base class will carry the left and right pointers for the data structure, as well as price and nutrition ratings for each food service. It will also contain some basic information (strings) about the location of the restaurant and the most popular dish they serve. There will be bool members representing if the restaurant’s food is available for pickup and delivery.

The derived class Chinese will have a field for spicy, as will the Mexican class. The vegetarian class will have a field to determine whether or not vegan options are available.

Contained in the Restaurant base class will be a pointer to the root of a tree containing Dish class objects, representing all the available dishes at the given restaurant. This class will contain fields for number of calories, if the dish is gluten free, vegetarian, or vegan, as well as a string field for the name of the dish.

The main challenge in this assignment is going to be the data structure. I will attempt to round that out first, that way the rest of the program isn’t essentially waiting on that portion to be finished. As for learning the Java programming language, from what I’ve seen, a lot of it seems to be less complex than C++. I believe that I will appreciate some of the built-in features the language has to offer, such as automatic garbage collection. It will be nice not having to worry about finding and fixing memory leaks manually, although I do think in some ways this puts more responsibility on me as a programmer to bear in mind best practices as I work through the design process, so that I don’t use this automatic feature as a crutch. I need to make sure that my design is still lean and efficient, that way I’m not forfeiting quality for ease.