Tanner Krewson
Dr. Matthews
CS 260
3 March 2017

Toodle Design Critique

In my final design of Toodle, I ended up changing more things from my original design than I expected. One of the first things I decided to do was to combine my proposed TodoList object and TaskList object into one. I didn't feel as though the functionality differences between the two justified having a separate class for both. Next, I changed the array of tasks to an ArrayList of tasks. With a regular array, the array would have had to have been recreated every time a task was added or removed. The ArrayList provides this kind of functionality inherently, so it was the better choice for the job. In my original design, the Task class, which was to be extended by IncompleteTask, CancelledTask, and CompletedTask, was not abstract. I made it abstract in the final because all types types of tasks are a subclass of Task, and it would not be sensible to have just a Task with no type on its own. Other than those issues, the general design structure stayed the same as my original design. I did, however, add some extra methods to these classes that I did not consider when I created my original design.

My original design for the Task class did not include any methods. I realized when implementing my design that it would be beneficial to include a few methods. Namely, as opposed to making the four instance variables in Task public, I decided to make them private and provide getters for all four. This allowed the Task class to be immutable. When I began implementing a system to sort the tasks in the TaskList object, I needed an easy way to compare Tasks to each other. So, I decided to add two public methods to make comparison of two Task objects simple. One compares them by their relative priority order, and the other compares them by their unique identifier. When I decided to combine the TodoList and TaskList classes that I created in my original design into one TaskList object, I made the made the methods that were originally public in the original TaskList object, such as addTask, removeTask, etc. private. The reasoning for this was that while the original public methods took Task objects through their parameters, I wanted the public interface of TaskList to not deal in Task objects. So the createNewTask public method of the final TaskList class takes in each of the four details of a Task individually, creates a new Task object internally, and calls the private addTask method with the newly created Task object. This prevents users of the TaskList class from accidentally adding a method to the TaskList with an identifier that is not unique. This was one of the most major changes, and I believe that my final implementation of Toodle was an effective and easy to understand solution to the design that the functional specification proposed.