Castillo, Don Q.
casd3720
Nov. 9, 2020

What was the easiest aspect of extending another developer's software?

The easiest aspect of extending another developer's software is being able to reuse methods that already exist in the developer's model classes. For example, I did not have to code the Card, Deck, and Player classes from scratch. I could easily create an object out of those classes when I implemented the Rummy class and call the member methods in those classes to manipulate data. The Game, which is the base class, was also foundational in creating Rummy because it provided me some guidance on how to structure the Rummy class.

What was the most challenging aspect of extending another developer's software?

The most challenging aspect of extending another developer's software is figuring out what each method in the given classes actually does given the fact that the developer did not provide any kind of comments or documentation to his code. For example, I had to examine what getCard() method in the Player class does? Does it only return a copy of the card from the player's hand based on the passed index or does it also remove that card from the player's hand? It turned out that it does not really remove that card from the player's hands which means I had to write another method in the Player class called removeCard().

Another challenging aspect is implementing inheritance so that the user can choose between GoFish and Rummy. The base class only provided a very few methods that I could override and use in the Rummy class. There are also methods in the base class that are pure virtual, forcing me to override those and not use them at all in the Rummy game. Since I had to check if the player's hand contains a run (set of cards with same suit and are sequential), I also ended up creating more methods in the Card class to overload operators.

How would you rate the quality of the code that was provided for extension? Provide a justification for your assessment.

If I were to rate the provided code from 1 (lowest) to 10 (highest), I would rate it 7. The codes were understandable and formatted in such a way that it does not confuse the reviewer. For example, the developer made use of whitespaces to separate function from one another. He also removed braces in loops and conditions that only has one line of statement within it which made it not messy to look at. As mentioned, the provided code would have been more readable if the developer provided documentation to briefly explain what each method does, what the parameters mean, and what kind of data are expected to return.

What could the previous developer have done to make new feature development easier?

As mentioned, documentation is necessary to communicate the intention of the developer to other developers who will be extending the code. The previous developer should have also written a more generic methods in the base class that applies to all kinds of card games. For example, my Rummy game did not use the beforeCardPlayed() and afterCardPlayed() methods which are declared virtually in the base class. I did not want to risk changing the parameters of these methods because I know it would affect the GoFish game which uses those.

The previous developer should also make assumptions about what other developers expect a particular class to have like having a setter and getter methods to access private data members.

What could you have done with your software that would have made the job of the maintenance/new feature developer of your software easier?

Implementing the Rummy game entails writing more methods needed to make the game work. I might have written methods that are too long. I should have broken these methods into shorter methods to make it easier to test. I also should have decoupled more methods and data to allow more flexibility to future developers who might want to extend or reuse some of the code I wrote.