Student Details

Name: Christopher Kevin Hamdajani

Id: 15995114

Name: Dianame Altalim

Id: 47885965

In our Trader Adventure application that we have made, we divide the code into two parts which are core classes and user interface classes. In our application, we have a few classes in our core package which are Ship, Island, Store, Item, Route, Goods, Equipment, Random Event, and Game Environment. The first step in developing our application was create all the objects needed in the Main class and call Game Environment. We stored most of the main functions that will be used by both user interface in the Game Environment class where this class will be used to call all the other classes. We created four types of Ships with different characteristics. Each of the ship will have advantage and disadvantage.

In order to create a new object of Island, we created a Store class and Route class. The reason is that Store should be on Island and every Island will have a route that connects that Island to another Island. But on the other hand, Route class should contain Island as it will connecting 2 Islands, therefore, to create Route class we stored two Strings which is the name of the 2 connected Islands.

Store class will contain HashMap of Goods that the store sells, HashMap of Goods that the Store wants to buy, and Equipment that the store sells. The reason we are using HashMap instead of Array List is that the data that Store has should contain two different types which are Goods good and Integer prices. Before creating a Store class, we created Item class. We could differentiate Item class into two types, Goods and Equipment. Here where Inheritance work. Goods and Equipment class will extend Item class and we will store attribute String name in the Item class.

Once Store and Route class was finished, we could construct the Island class which has the Island's name, Store in that Island, and collection of routes that the Island has. This shows that there is a direct association relationship between Island, Store, and Route class. There will be five Islands with their characteristics. Each route has a probability of a random event that might occur but the random event itself has no relation with the route. The random event in the route will be called by the Game Environment class.

Although Command-Line and Graphical User Interface work in the same way, but both of them produce different things. Therefore, we are using an interface for this case and called it GameEnvironmentUi. Another interface that we have is called Screen that will be implemented by all screen classes. In our Gui(Graphical User Interface) class, we define interface Screen as current screen and keep assigning screen class that will be used as a current screen.

Testing that we have done so far covering about 21% of our code. The reason why we only able to cover that amount is that most of our code will call the function in the user interface which cannot be tested in eclipse IDE. Therefore, we should do manual testing for the user interface such as Command-Line and Graphical User Interface. The other reason is that we use a random unction which makes us will not know what is the exact result of that function.

This project, for us, truly open our minds about how working as a developer in the future would like. In the beginning, we thought that it was all about coding and solving problem, but it is not. Working as a developer is much more about communication and teamwork. We glad we had this project as our introduction to the software developing world. Since this is our first big project, where we had to code a lot as we have never done before, we have to admit it is so frustrating and time-consuming. In addition to that, we felt that the course layout is a bit confusing.

For example, we have created our UML Diagram during the semester break, but after the break, we got a full explanation about UML Diagrams, which make us understand better how the UML Diagrams works but also realize that we made a lot of mistake in the UML Diagrams we have created before, therefore we must recreate basically from scratch and of course it takes time. Overall, even though this project is a bit hard for us to do, we glad that we have a chance to do this project.

To be honest, we just lack experience, this problem affects our team performance since we have to learn to do a lot of things at once as fast as possible. For example, we never use Git Lab before, even though the tutorial helps us to understand how to use git, sometimes we faced problems that were not covered in the tutorial, so we had to find a lot of sources to understand that and again it took a lot of time.

We also did not understand the project specification thoroughly when we start to code our project, therefore we had to make a lot of adjustments in the middle and the end of our project, for example, we did not realize that we cannot put the game logic in the GUI classes, therefore we had to make a lot of changes to move the game logic from GUI classes to the classes where it supposed to be. In addition to that, we also could not achieve our expectation of test coverage.

However, we glad we managed to design the graphical user interface like we wanted it to be. We also think that we generate good Javadoc that for us informative and useful. We also manage to implement the Inheritance and Interface in our programs, and it works well. There are a lot of things that we think we should improve on. First, we should understand thoroughly first about the project specification before doing anything.

Second, for the next project, we think it would be much better if every week we ask for feedback from tutor or lecturer about the progress we had to make so that if there is something we could improve on or if we make mistakes, we can discover the problem earlier and fix it, and do the improvements. Third, we also think for our next project, we always should prepare a backup plan for everything that we try to implement which delivers the same functionality with the original plan, so when we meet the problem that we cannot solve easily in our original plan we can shift to our back up plan quickly so that we do not waste much time.

We both spent, on average, 20 hours per week working on this project. Therefore, in total, we worked around 200 hours on this project. Most of the time we work together so there was no big difference in our working time. As the result, we agreed that we do equally distributed the tasks in this project, so 50% each person.