Peer Review

Group: Sebastian Svensson

We believe that the registry class has too much responsibility of the member and boat classes. See if you can refactor most of the methods into the appropriate class.

Every attribute is a string. The boat type should be a description class on it's own. When we tested the application we could add a member with empty name, personal identification number and the boat with any kind of type we wanted (eg. type = "foo"). Add validation to the boat and member class.

We found a naming error in Repository where Member is named Membert.

The sequence diagrams are good, but the class diagram is more confusing than helping. It's a correct representation of the code, but the code is too cluttered and focused on the registry class.

The ConsoleView should know about the MemberRegistry and accept Member objects instead of strings from the controller. This is information that the view can collect and organize on its own from the model instead of the controller having to loop through the members.

The controller is using "Console.ReadLine();" and not the ConsoleView, this means that it would be impossible to change the view to a different UI since the controller is dependent on the Console methods. [Larman chapter 13 section 6]

Member does not have an association to any kind of boat or list of boats, everything is based on an index in a list in the registry, we counted that [i-1] is used 23 times and [a-1] is used 12 times in your application. See the requirements about the use of id.

An error we found when "Viewing a member":

- Viewing a member of id 1 shows member with the id of 2
- Viewing a member of id 5 shows member with the id of 7
- Viewing a member of id 7 returns an error because member with the id of 9 doesn't exist

Also, If we delete a member and then add a new one, they will have had the same id, it's not unique since you use count() to determine the next id.

We do not believe that the assignment has passed the criteria.

Larman - Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development, Third Edition