Peer Review

For: 1DV607, Workshop 3, Grade 2
To: Adam(aj223bm),Mohamed Osman(my222au)
From: Rasmus(rl22dv), Henry(hp222fq), Seldin(smubc09)

Try to compile/use the source code provided. Can you get it up and running? Is anything problematic?
- We got the code running without any errors.

Test the runnable version of the application in a realistic way. Note any problems/bugs.
- We found no problems or bugs when testing the code in a realistic way. The Ace is calculated correctly which took some time due to the delay and some bad luck.

Does the implementation and diagrams conform (do they show the same thing)? Are there any missing relations? Relations in the wrong direction?
- There is an association between PlayGame and IView not a dependency. Otherwise the UML class diagram is correct and its also updated.

Is the dependency between controller and view handled? How? Good? Bad?
- You guys did no solve the hidden dependency. We believe the hidden dependency is still there.

> (hint) we swedes maybe want 's' to stand for spela (play)
> or if user does not have a 'p' on keyboard (e.g. a chinese keyboard)
> we will need to hardcode the logic to change this, instead of just
> adding a different view (e.g a chineseView)
> (although a chinese keyboard does have a 'p' you should get the point)

Is the Strategy Pattern used correctly for the rule variant Soft17?
- The Strategy pastern is implemented correctly Soft17 but the oneAce in Soft17Strategy is dead code. That's probably just an artifact from when you guys programmed.

Is the Strategy Pattern used correctly for the variations of who wins the game?
- Yes the win strategy is implemented correctly. The solution is both elegant and correct according to GoF found in Larmans book[1, 26.7 Strategy]

Is the duplicate code removed from everywhere and put in a place that does not add any dependencies (What class already knows about cards and the deck)? Are interfaces updated to reflect the change?
- Your dealing of a card strategy is not the best approach, you reduced the duplicate code by adding a duplicate code in the Dealer (you can use the 2nd method and just pass in a true instead of having another method for only the case of true) also you are just adding more dependency, try using an alternative method like an abstract class or something.
What we are trying to explain is you already have all you need to make this method in the hit class, why make it so that dealer have to deal with it? You don't even use the Deck you are provided with.

Is the Observer Pattern correctly implemented?

- Player to subject but still Player has the method notify() which the observer should have although this is not a major problem. The pattern shows that if observer A makes a changes observer B should get the change, in this case the subject 'triggers' the change. We also miss some out-prints because of the printing method is called somewhere but then just keeps on adding breaklines (1500 lines to be exact which is a bit too much...).

Is the class diagram updated to reflect the changes?
Yes the UML class diagram is updated the new classes and correct arrows except for one which is mentioned above.

Do you think the design/implementation has passed the grade 2 criteria?
No we dont think this is a pass for Grade2 but we think you guys have made a great effort and can probably do it the next time.

To think about
- In Player there is a method called Addsub(). Try to use standard java naming convention when you program. Instead it should be AddSub() according to java naming convention.

Not implemented or not implemented correctly
: Remove the bad, hidden, dependency between the controller and view (new game, hit, stand)
: The code for getting a card from the deck, show the card and give it to a player is duplicated in a number of places. Make a refactoring to remove this duplication and that supports low coupling/high cohesion.
: Observer may be working but it's not following the pattern Observer pattern in GoF.
: Update UML class diagram between PlayGame and IView.

References
1.Larman, C., Applying UML and Patterns 3rd Ed, 2005, ISBN: 0-13-148906-2