FRÓÐSKAPARSETUR FØROYA
UNIVERSITY OF THE FAROE ISLANDS

Design Patterns with Java (5041.24)

Assignment 01 – Strategy & Observer

Deadline 17. Jan. 2025 23:59:59

Óðin Ellefsen - Not approved
Bartal Clementsen 18/01/2025

General Notes

Assignemnt 1 is fine but assignement 2 is not okey at all. We will make a quick walkthrough of the solutions on Monday. Please be there an look how this can be implemented.

You will have to read through the chapter 2 again .

Then resubmit again for next week at least part 2, but you are welcome to try part 1 without using the enum if you want 😊 .

01: Refactoring Deck Storage to Strategy

This is a fine solution 😊 and a good diagram

Comment

1) You've decided to handle the setStorage(StorageFormat format) inside the Deck class using the enum. This is OK, but a bit redundant. If you decide to add a new storage strategy you also need to change the enum and the setStorage(StorageFormat format) method.
   The better way is to have the constructor accept the DeckStorage interface as a parameter at construction time. Something like this

```
public Deck(DeckStorage deckStorage) {
    this.storage = deckStorage;
    cards = new ArrayList<>();
    for (Suit suit : Suit.values()) {
        for (Rank rank : Rank.values()) {
            cards.add(new Card(suit, rank));
        }
    }
}
```

   Then you can stop using the StorageFormat enum and the Deck never needs to "know" what type of DeckStorage(s) there are only that one of them is given at startup.

   You could keep the setStorage but change it to accept an object that implements the DeckStorage interface, then you can change the storage strategy at runtime. Something like this.

```
public void setStorage(DeckStorage deckStorage) {
```

   I see that you handle the case where storage is null. If you get the DeckStorage at construction time you would not need to handle this here, because you could guarantee that it would not be null.

02: My awsome stockmarket

You haven't implemented the observer pattern in this solution. The only thing you've done is the naive (wrong) solution that is demonstrated in the first part of the chapter in the book :S. The diagram is correct for your solution but doesn't represent the Observer pattern. It should be something like this: