Northeastern University
CS5500 – Managing Software Development
Fall 2018, {Jump, Katz, and Weintraub}

Grading for Homework 2: Design Patterns

Student Name:	Arcylizet Iniguez Flores
Repo Name:	https://github.ccs.neu.edu/cs5500/student-49-F18
Repo Commit Date (UTC)	OK ['2018-10-12', '01:44:16Z']
Graded By:	Praveen
Total Score:	93

1 Project Set-up

1. **QUESTION:** Is Maven used?

   Yes

2. **QUESTION:** Does Maven works correctly?

   Works Correctly

3. **QUESTION:** Did the student follow the artifact and groupID convention?

   Followed

4. **QUESTION:** Did the student put the code into the proper package?

   Proper Package

2 Cards, Suits, and Ranks

5. **QUESTION:** Are Card.getSuit() and Card.getRank() implemented?

   Yes

6. **QUESTION:** Are Suit.getName() and Suit.getSymbol() implemented?

   Yes

7. **QUESTION:** Are Rank.getPips(), Rank.getName() implemented?

   -

   Yes

---

8. **QUESTION:** Does the code compile without errors?

   -

   Yes, compile without errors

---

9. **QUESTION:** Any signs of sloppiness? (e.g. TODO comments, obsolete code that was commented out, duplicated code, major sonarlint smells.)

   -

   No signs of Sloppiness

---

10. **QUESTION:** Are the methods in the classes and interfaces properly documented using javadoc?

    -

    Minus Points: 0

---

11. **QUESTION:** Creating a single card correctly?

    -

    Yes

---

12. **QUESTION:** Creating all cards for a standard deck correctly?

    -

    Yes

---

13. **QUESTION:** Creating all cards for all deck types correctly?

    -

    Yes

---

14. **QUESTION:** Creating a card where at least suit or rank object is null and the creations fails by design?

    -

    -0 pts Warning

---

3 Deck, Euchre, Pinochle, Standard, and Vegas

15. **QUESTION:** Are the following methods implemented for Deck: 1. getCards(), 2. shuffle(),3. sort(String guidance) and supports three types of guidance, byRank, bySuit, or both. The literals aren't important., 4. cut(int cutPoint), 5. emptyDeck(), 6. officialSize()

    -

    Yes

---

16. **QUESTION:** Are any methods repeated in the subtypes?

    -

    No repetition

---

17. **QUESTION:** Does each subtype keep its own "official size"?

    -

    Yes

---

18. **QUESTION:** Does the Vegas deck have one Deck or collection of decks?

    -

    Yes

---

19. **QUESTION:** Does the code compile without errors?

    -

    Yes, Compiles without errors

---

20. **QUESTION:** Any signs of sloppiness? (e.g. TODO comments, obsolete code that was commented out, duplicated code, major sonarlint smells.)

    -

    No signs of sloppiness

---

21. **QUESTION:** Are the methods in the classes and interfaces properly documented using javadoc?

    -

    Minus Points: 0

---

22. **QUESTION:** Do the tests pass without errors?

    -

    Yes, tests pass without errors

---

23. **QUESTION:** Evaluating if shuffle and cut work properly?

    -

    Yes, works properly

---

24. **QUESTION:** Trying to cut the deck with cutPoint values set to ¡ 0 or ¿ officialSize() +1;

    -

    Satisfactory

---

25. **QUESTION:** Evaluating if pullCard() works properly?

    -

    Yes, works properly

---

26. **QUESTION:** Trying to pullCard() from an empty deck?

    -

    -0 pts Warning

---

27. **QUESTION:** Evaluating if emptyDeck() and officialSize() work properly?

    -

    Yes, works properly

---

28. **QUESTION:** Creating all cards for all deck types correctly?

    -

Yes, works properly

---

29. **QUESTION:** Evaluating if shuffle and cut work properly?

- -

Yes, works properly

---

30. **QUESTION:** Evaluating if sorting works on all three criteria?

- -

Yes, works properly

---

4 Hand and Game

31. **QUESTION:** Does the code compile without errors?

- -

Yes, All methods implemented

---

32. **QUESTION:** Any signs of sloppiness? (e.g. TODO comments, obsolete code that was commented out, duplicated code, major sonarlint smells.)

- -

Yes, Compiles without errors

---

33. **QUESTION:** Are the methods in the classes and interfaces properly documented using javadoc?

- -

Minus Points: No signs of sloppiness

---

34. **QUESTION:** EDo the tests pass without errors?

- -

0

---

35. **QUESTION:** Creating each type of deck correctly?

- -

Yes, tests pass without errors

---

36. **QUESTION:** Trying values set to ¡ 0 for game.setNumberOfHands()?

- -

Yes, created correctly

---

37. **QUESTION:** Evaluating if createDeck(anythingButVegas, n¿1) fails?

- -

-0 pts Warning

---

38. **QUESTION:** Evaluating if createDeck(anything, n¡0) fails?

- -

-0 pts Warning

---

39. **QUESTION:** Evaluating if deal() works, meaning after dealing n cards, the hands have the n cards and the deck is smaller by n cards?

    -

    -0 pts Warning

---

40. **QUESTION:** Evaluating if showCards() returns the list of cards and that the hand still has the cards?

    -

    -1 pts No

---

41. **QUESTION:** Evaluating if after acceptCard() is run, that the Hand has one more card?

    -

    Yes

---

42. **QUESTION:** Evaluating if after acceptCard() is run on a "full hand", that request fails?

    -

    Yes

---

43. **QUESTION:** Evaluating hasCard() for when it's true and when it's false?

    -

    -0 pts Warning

---

44. **QUESTION:** Evaluating Hand.sort() where there's only one type of deck used?

    -

    -1 pts No

---

45. **QUESTION:** Evaluating Hand.sort() where tall types of decks are used?

    -

    -1 pts No

---

46. **QUESTION:** Evaluating Hand.shuffle() where there's at least one type of deck used?

    -

    -2 pts No

---

47. **QUESTION:** Evaluating Hand.shuffle() where all types of deck are used?

    -

    Yes

---

5 Flexible Playing Cards

48. **QUESTION:** Does the code compile without errors?

    -

    Yes, Compiles without errors

---

49. **QUESTION:** Any signs of sloppiness? (e.g. TODO comments, obsolete code that was commented out, duplicated code, major sonarlint smells.)

    -

    No signs of sloppiness

---

50. **QUESTION:** Are any methods excessively long? (¿30 LOC not counting white space or comments)

    -

    No, they are not excessively long

---

51. **QUESTION:** Are the methods in the classes and interfaces properly documented using javadoc?

    -

    Minus Points: 0

---

52. **QUESTION:** Do the tests pass without errors?

    -

    Yes, tests pass without errors

---

53. **QUESTION:** Does the test suite cover all types of decks?

    -

    Yes, covers everything

---

54. **QUESTION:** Does the test suite cover at least two sizes for hand?

    -

    -2 pts: Miss ¡=2 methods

---

55. **QUESTION:** Does the solution make proper use of a creational design pattern for games?

    -

    Yes

---

56. **QUESTION:** Do the tests use the design pattern correctly?

    -

    Yes

---

57. **QUESTION:** Does the solution make proper use of a creational design pattern for Deck?

    -

    No

---

6 General Comments

58. **QUESTION:** Missing Tests?

    -

    Minus Points: 0

---

59. **QUESTION:** Breaking Abstraction or inheritance?

    -

Minus Points: 0

---

60. **QUESTION:** Comments to Student

- -

Good work, just missed some unit test cases.

---

61. **QUESTION:** Screenshots if any

- -

`None`
