

## **COE318 – Lab 4: Cards and BlackJack**

### **Objectives**

- Implement a `Card` class.
- Implement a `CardPile` class.
- Implement a `SimpleUI` class.
- Fix the `BlackjackGame` class.
- Use an `ArrayList`
- Perform user I/O.

**Duration:** two weeks.

### **Grading Scheme:**

50% submitted source code

20% in-class demonstration and questions (during week 8 lab hours)

25% in-class quiz – Held during the first 5 mins of the lab class (during week 5 lab hours)

5% attendance

### **Overview**

Blackjack is the most popular casino gambling game in the world. (Note: the rules of the game favor the casino. The Casino is guaranteed to win in the long run.) Here's how it works:

1. Both you and the House (the Casino) are dealt two cards: one is face up and the other face down. So you can see only one of the House's cards and it can see only one of yours. But both you and the House can discreetly peek at your face down cards.
2. Each card has a score as follows:
  - An Ace has a score of 1. (This differs from real Blackjack where the Ace can have a score of either 1 or 11 at the player's discretion. The simplified version for this lab is easier to program.)
  - A Jack, Queen or King has a score of 10.
  - All other cards have a score equal to their rank. (For example, the 4 of Hearts or the 4 of any suit have a rank of 4 and a score of 4.)
3. The House will obtain additional cards until its score is 17 or more.
4. You are asked if you want another card. If you answer yes, you get another face down card and you will be asked again. This continues until you say "no". This ends the game.
5. All cards are now turned face-up and the scores of you and the House are calculated.
  - You lose if your score is more than 21 (no matter what the House's score is).
  - You lose if your score is the same as the House's.
  - You win if:

- You don't go over 21 and the House does go over 21
- Both your scores are 21 or under and your score is more than the House's.

## The Card Class

The Card class will consist of instance variables, a constructor and the methods. The template contains javadocs for all the methods. You should generate the html representation of the class to have an easier representation of the class's API (Application Programmer Interface). Many of the methods are only stubs and you have to modify them so that they implement the API.

The template will compile and it has a main method that will run but it produces the wrong output.

## Source Code

`Card.java`, `CardPile.java`, `BlackJackGame.java`, `UserInterface.java` and `SimpleUI.java` classes are provided with the lab. More details about these classes are in the instructions below.

### Step 1: Create a Netbeans Project and *Card* class

1. Create a Netbeans project called `Blackjack`.
2. Create a Java file (class library type) called `Card` with package `coe318.lab5`. (All java files in this lab should have that package declaration.)
3. Determine your instance variables and implement the constructor.
4. Implement the other methods.
5. **Important:** Do **not** continue to the next class until this class works!

### Step 2: Implement the remaining classes

You should get the `CardPile` class to work before proceeding. (It has its own main method.)

You next have to create the `BlackjackGame` and `SimpleUI` classes starting with the templates provided. You also have to create an **interface** (`UserInterface`) copying the code given in D2L. To create an interface in Netbeans, create a new file and have an interface (not a class) created. We have not covered interfaces yet in lectures. However, the only thing you need to know for now is that this file is essential and must not be modified.

Note: All three of these templates have to be loaded in order for them to compile without error.

### Additional Notes

1. The `Card` class `compareTo(Card c)` method should return a negative, zero or positive value depending on whether “this” is less than, equal to or greater than the other card. For 2 cards of unequal rank, the one with the higher rank is “bigger”. If the ranks are the same, the suit is considered; the suit orders (from lowest to highest) are Clubs, Diamonds, Hearts, Spades.
2. The `Card` `equals(Card c)` method should only return true if the cards have the same suit and

rank.

### Sample Gaming Sessions

The following are 2 sample sessions. In one the House wins, in the other you win.

**Note:** The format of the output is not defined. Within SimpleUI you can format the output of `display()`, `hitMe()` and `gameOver()` any way you want.

```
House holds:
  ?
  7 of Hearts

You hold:
  Queen of Hearts
  Jack of Clubs

House holds:
  ?
  7 of Hearts
  6 of Spades

You hold:
  Queen of Hearts
  Jack of Clubs

Another card? n
House holds:
  ?
  7 of Hearts
  6 of Spades
  King of Spades

You hold:
  Queen of Hearts
  Jack of Clubs

Game over
House holds:
  Ace of Diamonds
  7 of Hearts
  6 of Spades King of
  Spades

You hold:
  Queen of Hearts Jack of
  Clubs

Your score: 20, House score: 24 You win
```

```
House holds:
  ?
```

4 of Clubs

You hold:

King of Spades  
3 of Spades

House holds:

?  
4 of Clubs  
Ace of Diamonds

You hold:

King of Spades  
3 of Spades

Another card? y House  
holds:

?  
4 of Clubs  
Ace of Diamonds

You hold:

King of Spades  
3 of Spades  
9 of Clubs

House holds:

?  
4 of Clubs  
Ace of Diamonds Queen of  
Clubs

You hold:

King of Spades  
3 of Spades  
9 of Clubs

Another card? n Game

over House holds:

Jack of Spades  
4 of Clubs  
Ace of Diamonds Queen of  
Clubs

You hold:

King of Spades  
3 of Spades  
9 of Clubs

Your score: 22, House score: 25 The  
House wins

#### **Step 4: Submit your lab**

You must submit your lab electronically on D2L. Please make sure you hand over the quiz answer sheet to the TA at the end of the in-class quiz.

Please zip up your NetBeans project containing all source files and submit to the respective assignment folder on D2L.