This homework assignment is designed to give you more practice writing Java classes. This assignment asks you to write a class that simulates a playing card. Note that there is starter code for this assignment that you should download from the course webpage.

1 Playing Cards

In this assignment, you are asked to write a class named Card to represent a playing card. A playing card has the following attributes:

- A face value which is an integer ranging from 1 to 13. A face value of 1 corresponds to an Ace. A face value of 11, 12, or 13 corresponds to a Jack, Queen, or King respectively. These three are known as face cards because they usually have a face drawn on them.
- A suit which is either: diamond, heart, spade, or club. Diamonds and hearts are red in color. Spades and clubs are black in color.

The starter code for this assignment has a Card class that I have already started for you. Add the appropriate instance variables to represent the card’s face value and suit. Next, add the following methods:

- A constructor that takes a suit and a face value as input arguments.
- A constructor that takes no inputs. This constructor should randomly generate the card’s face value and suit.
- Getter methods for the suit and face value
- isBlack() – this method returns whether the card is black
- isRed() – this method returns whether the card is red
- isFaceCard() – this method returns whether the card is a face card
- hasSameFaceValue(Card other) – this method returns whether this card and the other card have the same face value
- hasSameSuit(Card other) – this method returns whether this card and the other card have the same suit
- equals(Card other) – this method returns whether this card and the other card are equal
- outRanks(Card other) – this method returns true if this card has a strictly greater face value than the other card. The only exception is a face value of 1 (i.e. an Ace) which outranks all other face values.
- A toString() method. Your toString() method must return a string in this precise format:

  [suit, faceValue]
where a face value of 1 is converted to an “A”, an 11 to “J”, a 12 to “Q” and a 13 to “K”. For example, an ace of spades should return \([\text{spade, A}]\) and a three of clubs should return \([\text{club, 3}]\). (My tester code relies upon your \texttt{toString()} method returning this exact format.)

When you are done, write a \texttt{CardController} class that creates 2 cards and calls at least 5 methods on them. Add enough print statements so that when I run your code, it is clear what is happening.

2 Style Guide

Be sure to go through the list below carefully and make sure your code adheres to the style guide:

- Delete any unused instance variables
- If an instance variable isn’t used in multiple methods, then demote it to a local variable inside the method where it is actually used.
- All instance variables should be initialized in the constructor
- You have a Javadoc comment at the top of the class with a brief description (written in full English sentences), you and your partner’s name, and the date.
- Each method you write has a Javadoc comment with appropriate \texttt{@param} and \texttt{@return} statements
- All variable names are lower cased (remember, only classes are capitalized in Java)
- Use inline comments (//) to explain any complicated code

3 Submitting your assignment

Please make sure to rename your folder \texttt{before} zipping. You should rename your folder using both of your first and last names. For example, \texttt{hw6_John_Doe_Jane_Doe}.

Submit your zipped folder via Canvas.