

CS161: Introduction to Computer Science
Homework Assignment 6
Due: 3/7 by 11:59pm

Playing Cards

In this assignment, you will use conditionals to implement a Java class representing a single 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 a **diamond**, **heart**, **spade**, or **club**. Diamonds and hearts are red in color. Spades and clubs are black in color.



This assignment also asks you to write a second Java class (with a `main` method) to test your playing card class.

Your code for this assignment will be graded in part by its adherence to our class's Java style guide. At the end of this writeup is a checklist to help you out.

The Card class

Download the starter code from the course webpage. The starter code contains a `Card` class for you to finish. The `Card` class should have 2 constructors:

- A constructor that takes no input arguments. Instead, randomly generate both a face value and a suit for the card
- A constructor that takes both a suit and a face value as input arguments. Now that we can use conditionals, we can check that the user passed correct values to the constructor. If the suit or the face value passed by the user are not legal, go ahead and generate random values.

Having multiple constructors in a class is simply a convenience for whoever uses our class – it gives the user options. When someone creates an object of type `Card` they choose which constructor to call. This means it's very important for *both* constructors to fully initialize all instance variables.

Your class should also have the following methods:

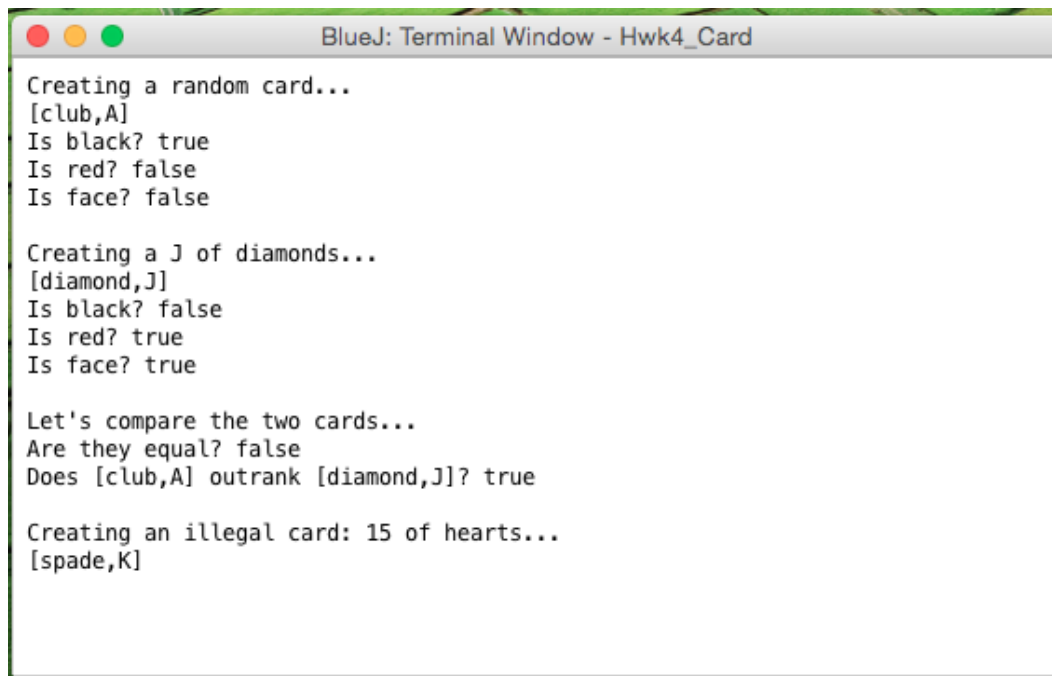
- Accessor methods for the face value and suit
- The following methods should return a boolean
 - `isBlack()` returns whether the card is black
 - `isRed()` returns whether the card is red
 - `isFaceCard()` returns whether the card is a face card
 - `hasSameFaceValue(Card other)` returns whether this card and the other card have the same face value

- `hasSameSuit(Card other)` returns whether this card and the other card have the same suit
 - `equals(Card other)` returns whether this card and the other card are equal (i.e. have the same suit and the same face value).
 - `outranks(Card other)` 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 should convert a face value of 1 to an “A”, an 11 to “J”, a 12 to “Q” and a 13 to “K”.

Testing Your Class

Write a second class called `CardTester` that creates objects of type `Card` and uses the dot operator to call methods from the card class. Since you have two constructors, create at least two instances of the `Card` class.

Below is an example of what my `CardTester` prints when executed:



```
BlueJ: Terminal Window - Hwk4_Card
Creating a random card...
[club,A]
Is black? true
Is red? false
Is face? false

Creating a J of diamonds...
[diamond,J]
Is black? false
Is red? true
Is face? true

Let's compare the two cards...
Are they equal? false
Does [club,A] outrank [diamond,J]? true

Creating an illegal card: 15 of hearts...
[spade,K]
```

Notice that when I pass in an illegal value to the constructor (a face value of 15), it instead randomly initializes the card resulting in a King of spades.

Style Guide

Before you submit your assignment, go through the checklist below and make sure your code conforms to the style guide.

Checklist

- All unused variables are deleted
- All instance variables are used in more than one method (if not, make them local)
- Javadoc comment for the `Card` class
- Javadoc comment for the `CardTester` class
- All methods have Javadoc comments (except for the `main` method)
- All numbers have been replaced with constants (i.e. no magic numbers)
- Proper capitalization of variables, methods, and classes
- Use white space to separate different sections of your code

Read the “Style Guide” (under “Resources” on the course website) for more information.

Submitting your lab assignment

You should submit your `hw6` folder with your `Card` class and your `CardTester` class.