# Intro to Java Week 6 Coding Assignment

**Points possible:** 70

|  |  |  |
| --- | --- | --- |
| Category | Criteria | % of Grade |
| Functionality | Does the code work? | 25 |
| Organization | Is the code clean and organized? Proper use of white space, syntax, and consistency are utilized. Names and comments are concise and clear. | 25 |
| Creativity | Student solved the problems presented in the assignment using creativity and out of the box thinking. | 25 |
| Completeness | All requirements of the assignment are complete. | 25 |

**Instructions:** In Eclipse, or an IDE of your choice, write the code that accomplishes the objectives listed below. Ensure that the code compiles and runs as directed. Take screenshots of the code and of the running program (make sure to get screenshots of all required functionality) and paste them in this document where instructed below. Create a new repository on GitHub for this week’s assignments and push this document, with your Java project code, to the repository. Add the URL for this week’s repository to this document where instructed and submit this document to your instructor when complete.

**Coding Steps:**

For the final project you will be creating an automated version of the classic card game *WAR.*

1. Create the following classes.
   1. Card
      1. Fields
         1. **value** (contains a value from 2-14 representing cards 2-Ace)
         2. **name** (e.g. Ace of Diamonds, or Two of Hearts)
      2. Methods
         1. Getters and Setters
         2. **describe** (prints out information about a card)
   2. Deck
      1. Fields
         1. **cards** (List of Card)
      2. Methods
         1. **shuffle** (randomizes the order of the cards)
         2. **draw** (removes and returns the top card of the Cards field)
         3. In the constructor, when a new Deck is instantiated, the Cards field should be populated with the standard 52 cards.
   3. Player
      1. Fields
         1. **hand** (List of Card)
         2. **score** (set to 0 in the constructor)
         3. **name**
      2. Methods
         1. **describe** (prints out information about the player and calls the describe method for each card in the Hand List)
         2. **flip** (removes and returns the top card of the Hand)
         3. **draw** (takes a Deck as an argument and calls the draw method on the deck, adding the returned Card to the hand field)
         4. **incrementScore** (adds 1 to the Player’s score field)
2. Create a class called App with a main method.
3. Instantiate a Deck and two Players, call the shuffle method on the deck.
4. Using a traditional for loop, iterate 52 times calling the Draw method on the other player each iteration using the Deck you instantiated.
5. Using a traditional for loop, iterate 26 times and call the flip method for each player.
   1. Compare the value of each card returned by the two player’s flip methods. Call the incrementScore method on the player whose card has the higher value.
6. After the loop, compare the final score from each player.
7. Print the final score of each player and either “Player 1”, “Player 2”, or “Draw” depending on which score is higher or if they are both the same.

**Screenshots of Code:**

**App.Java**

package com.company;  
  
public class App {  
  
 public static void main(String[] args) {  
  
 // Make a deck  
 Deck deck = new Deck();  
  
 // Add two players  
 Player player1 = new Player("Max");  
 Player player2 = new Player("Sam");  
  
 // Shuffle deck  
 deck.shuffle();  
  
 // Draw the deck  
 for(int i = 0; i < 52; i+=2) {  
 player1.draw(deck);  
 player2.draw(deck);  
 }  
  
 // Flip cards and compare  
 for(int i = 0; i < 26; i++) {  
 Card card1 = player1.flip();  
 Card card2 = player2.flip();  
 if(card1.getValue() > card2.getValue()) {  
 player1.incrementScore();  
 } else {  
 player2.incrementScore();  
 }  
 }  
  
 // Compare score and declare a winner  
 if(player1.getScore() == player2.getScore()) {  
 System.*out*.println("Draw: " + player1.getName() + " and " + player2.getName());  
 System.*out*.println("Score: " + player1.getScore());  
 } else if (player1.getScore() > player2.getScore()) {  
 System.*out*.println("Winner Player 1: " + player1.getName());  
 System.*out*.println(("Score: " + player1.getScore()));  
 } else {  
 System.*out*.println("Winner Player 2: " + player2.getName());  
 System.*out*.println("Score: " + player2.getScore());  
 }  
 }  
}

**Card.java**

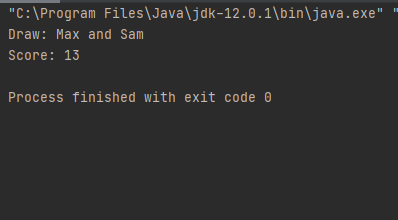
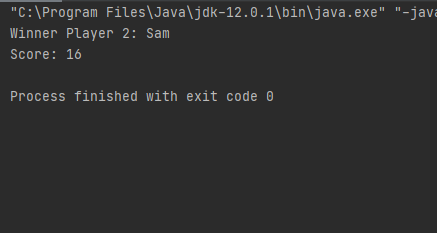
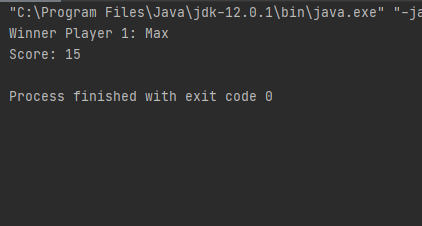
package com.company;  
  
public class Card {  
 private String name;  
 private int value;  
  
 public Card (String name, int value) {  
 this.name = name;  
 this.value = value;  
 }  
  
 public int getValue() {  
 return value;  
 }  
  
 public void setValue(int value) {  
 this.value = value;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public void describe() {  
 System.*out*.println("Card Name: " + name);  
 System.*out*.println("Card Value: " + value);  
 }  
  
}

**Deck.java**

package com.company;  
  
import java.util.ArrayList;  
import java.util.Collections;  
import java.util.List;  
  
public class Deck {  
 private List<Card> cards = new ArrayList<Card>();  
  
 public Deck() {  
 // Add Diamonds  
 makeSuite("Diamonds");  
  
 // Add Hearts  
 makeSuite("Hearts");  
  
 // Add Clubs  
 makeSuite("Clubs");  
  
 // Add Spades  
 makeSuite("Spades");  
 }  
  
 private void makeSuite(String suite) {  
 for(int i = 2; i <=14; i++) {  
 String name = nameCard(i);  
 Card card = new Card(name + " of " + suite, i);  
 cards.add(card);  
 }  
 }  
  
 public void shuffle() {  
 Collections.*shuffle*(cards);  
 }  
  
 public String nameCard(int value) {  
 String name;  
 if(value == 11) {  
 name = "Jack";  
 } else if (value == 12) {  
 name = "Queen";  
 } else if (value == 13) {  
 name = "King";  
 } else if (value == 14) {  
 name = "Ace";  
 } else {  
 name = Integer.*toString*(value);  
 }  
 return name;  
 }  
  
 public Card draw() {  
 Card top = new Card(cards.get(0).getName(), cards.get(0).getValue());  
 cards.remove(0);  
 return top;  
 }  
}

**Player.java**

package com.company;  
  
import java.util.ArrayList;  
import java.util.List;  
  
public class Player {  
 private List<Card> hand = new ArrayList<Card>();  
  
 private int score;  
 private String name;  
  
 public Player (String name) {  
 score = 0;  
 this.name = name;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public int getScore() {  
 return score;  
 }  
  
 public void setScore(int score) {  
 this.score = score;  
 }  
  
 public void describe() {  
 System.*out*.println("Name: " + name);  
 System.*out*.println("Score: " + score);  
 for(int i = 0; i < hand.size(); i++) {  
 hand.get(i).describe();  
 }  
 }  
  
 public Card flip() {  
 Card top = new Card(hand.get(0).getName(), hand.get(0).getValue());  
 hand.remove(0);  
 return top;  
 }  
  
 public void draw(Deck deck) {  
 Card card = deck.draw();  
 hand.add(card);  
 }  
  
 public void incrementScore() {  
 score++;  
 }  
  
}

**Screenshots of Running Application:**

**URL to GitHub Repository:**

[**https://github.com/jvgreen/Promineo-Boot-Camp/tree/master/Week6/coding06/src/com/company**](https://github.com/jvgreen/Promineo-Boot-Camp/tree/master/Week6/coding06/src/com/company)