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.
    a. Card
        i. Fields
            1. **value** (contains a value from 2-14 representing cards 2-Ace)
            2. **name** (e.g. Ace of Diamonds, or Two of Hearts)
        ii. Methods
            1. Getters and Setters
            2. **describe** (prints out information about a card)
    b. Deck
        i. Fields
            1. **cards** (List of Card)
        ii. 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.
    c. Player
        i. Fields
            1. **hand** (List of Card)
            2. **score** (set to 0 in the constructor)
            3. **name**
        ii. 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.
    a. 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:

**** Note****

I created a second War Game that plays the way I understand traditional War. When cards are flipped, both cards go to the winners hand. When the flipped cards are equal, a war occurs where 3 cards are laid on the table and the 4th card determines the winner of the war, with all 8 cards then going to the winner. Feel free to check it out in my Github repo :)

*App.java | Card.java | Deck.java | Player.java

```java
package war_game;

import java.util.ArrayList;

public class App {

    public static void main(String[] args) {
        String winner = "";
        int winnerScore = 0;
        int loserScore = 0;

        Deck deck = new Deck();
        deck.shuffle();
        List<Card> p1Hand = new ArrayList<Card>();
        List<Card> p2Hand = new ArrayList<Card>();

        Player p1 = new Player(p1Hand, 0 , "Jason");
        Player p2 = new Player(p2Hand, 0, "Hillary");

        // populates each player's hand
        for (int i = 0; i < 26; i++) {
            p1Hand.add(deck.draw());
            p2Hand.add(deck.draw());
        }

        // main game play
        for (int j = 0; j < 26; j++) {
            Card p1Flip = p1.flip();
            Card p2Flip = p2.flip();
            System.out.println("Round " + j);
            System.out.print(p1.getName() + ": " + p1Flip.getName() + " vs ");
            System.out.println(p2.getName() + ": " + p2Flip.getName());
            if (p1Flip.getValue() > p2Flip.getValue()) {
                p1.incrementScore();
            } else if (p2Flip.getValue() > p1Flip.getValue()) {
                p2.incrementScore();
            }
            // print result of flip
        }

        // comparing final scores and declaring winner
        if (p1.score > p2.score) {
            winner = p1.getName();
            winnerScore = p1.getScore();
            loserScore = p2.getScore();
        } else if (p2.score > p1.score) {
            winner = p2.getName();
            winnerScore = p2.getScore();
            loserScore = p1.getScore();
        } else winner = "NO ONE";

        String endGame = "The winner is " + winner + " with a score of " + winnerScore + ":" + loserScore;
        String dash = "=";
        System.out.println(dash.repeat(endGame.length()));
        System.out.println(endGame);
    }

}
```

Card.java × Deck.java Player.java

War Game/src/war_game/Card.java

```java
public class Card {

    int value;
    String name;

    public Card(int value, String name) {
        super();
        this.value = value;
        this.name = name;
    }

    public void describe() {
        System.out.println(name + " = " + value);
    }

    public int getValue() {
        return value;
    }
    public void setValue(int value) { // needed? not using
        this.value = value;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {  // needed? not using
        this.name = name;
    }
}
```

Deck.java × Player.java

```java
package war_game;

import java.util.ArrayList;

public class Deck {

    List<Card> cards = new ArrayList<Card>();

    Deck() {
        String[] rank = {"two", "three", "four", "five", "six", "seven", "eight", "nine", "ten", "jack", "queen", "king", "ace"};
        String[] suit = {"\u2666", "\u2663", "\u2665", "\u2663"};
        for (int i = 0; i < 4; i++) {
            for (int j = 2; j < 15; j++) {
                cards.add(new Card(j, rank[j - 2] + " of " + suit[i]));
            }
        }
    }

    // removes and returns the top card of the Cards field
    public Card draw() {
        Card drawnCard = cards.get(0);
        cards.remove(0);
        return drawnCard;
    }

    // randomizes the order of the cards
    public void shuffle() {
        Collections.shuffle(cards);
    }

}
```

*Player.java

```java
package war_game;

import java.util.List;

public class Player {

    // fields
    List<Card> hand;
    int score;
    String name;

    //constructors
    Player(List<Card> hand, int score, String name) {
        this.hand = hand;
        this.score = score;
        this.name = name;
        score = 0;
    }

    // public methods

    // prints out information about the player and
    // calls the describe method for each card in the Hand List
    public void describe() {
        System.out.println("PLAYER INFO:");
        System.out.println("    Name: " + name + ", " + "Score: " + score);
        System.out.println("    Cards in hand:");

        for (int i = 0; i < hand.size(); i++) {
            hand.get(i).describe();
        }
    }

    // removes and returns the top card of the Hand
    public Card flip() {
        Card flippedCard = hand.get(0);
        hand.remove(0);
        return flippedCard;
    }
```

```java
    // takes a Deck as an argument and calls the draw method on the deck,
    // adding the returned Card to the hand field
    public void draw(Deck deck) {
        Card returnedCard = deck.draw();
        hand.add(returnedCard);
    }

    // adds 1 to the Player's score field
    public void incrementScore() {
        score ++;
    }

    // getters and setters
    public List<Card> getHand() {
        return hand;
    }

    public void setHand(List<Card> hand) {
        this.hand = hand;
    }

    public int getScore() {
        return score;
    }

    public void setScore(int score) {
        this.score = score;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

}
```

Screenshots of Running Application:

```
Problems  @ Javadoc  Declaration  Console ×
<terminated> App (1) [Java Application] /Users/leeshawver/.p2/
Round 0
Jason: four of ♣ vs Hillary: three of ♥
Round 1
Jason: four of ♦ vs Hillary: nine of ♥
Round 2
Jason: queen of ♣ vs Hillary: queen of ♣
Round 3
Jason: three of ♣ vs Hillary: two of ♣
Round 4
Jason: ace of ♦ vs Hillary: queen of ♦
Round 5
Jason: eight of ♣ vs Hillary: five of ♣
Round 6
Jason: king of ♥ vs Hillary: seven of ♣
Round 7
Jason: four of ♥ vs Hillary: ten of ♥
Round 8
Jason: five of ♣ vs Hillary: king of ♣
Round 9
Jason: two of ♥ vs Hillary: jack of ♦
Round 10
Jason: jack of ♥ vs Hillary: king of ♦
Round 11
Jason: nine of ♣ vs Hillary: nine of ♣
Round 12
Jason: six of ♦ vs Hillary: eight of ♥
Round 13
Jason: six of ♥ vs Hillary: ten of ♣
Round 14
Jason: two of ♣ vs Hillary: eight of ♣
Round 15
Jason: ten of ♦ vs Hillary: eight of ♦
Round 16
Jason: three of ♦ vs Hillary: ace of ♥
Round 17
Jason: king of ♣ vs Hillary: five of ♦
Round 18
Jason: five of ♥ vs Hillary: nine of ♦
Round 19
Jason: ace of ♣ vs Hillary: four of ♣
Round 20
Jason: seven of ♦ vs Hillary: three of ♣
Round 21
Jason: six of ♣ vs Hillary: two of ♦
Round 22
Jason: queen of ♥ vs Hillary: jack of ♣
Round 23
Jason: seven of ♥ vs Hillary: jack of ♣
```

```
Round 24
Jason: seven of ♣ vs Hillary: ace of ♣
Round 25
Jason: six of ♣ vs Hillary: ten of ♣
==========================================
The winner is Hillary with a score of 13:11
```

URL to GitHub Repository:

https://github.com/leeshawver/Automated_War_Game