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:

```java
package cardGameWar;



public class App {

    public static void main(String[] args) {
        Deck deck = new Deck();
        Player playerOne = new Player("Sam");
        Player playerTwo = new Player("Jack");
        deck.shuffle();

        for(int i = 0; i < 52; i++) {
            if(i % 2 == 0) {
                playerOne.draw(deck);
            }
            else
            {
                playerTwo.draw(deck);
            }

        }


        Card cardOne;
        int valueOne;

        Card cardTwo;
        int valueTwo;
        for (int i = 0; i< 26; i++) {
            cardOne = playerOne.flip();
            cardTwo = playerTwo.flip();
            valueOne = cardOne.getValue();
            valueTwo = cardTwo.getValue();
            if(valueOne > valueTwo) {
```

```java
package cardGameWar;

import java.util.*;

public class Card {


    private int value;
    private String name;




    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;
    }
```

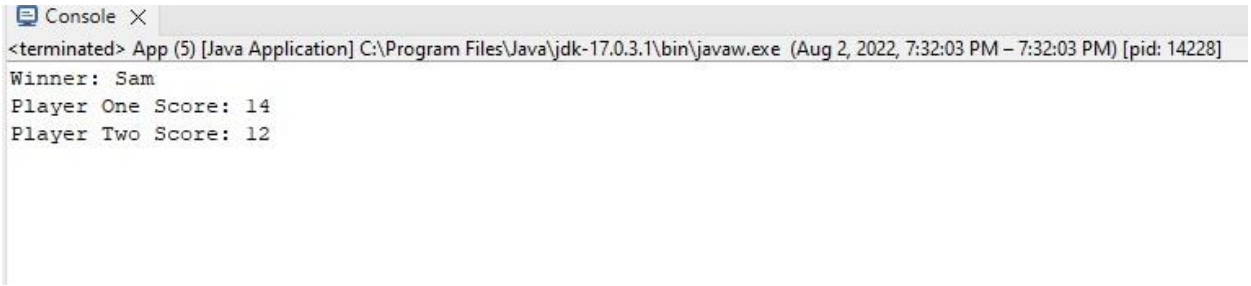
```java
package cardGameWar;

import java.util.*;

public class Deck {

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

    public Deck() {

        Card cardOne;
        Card cardTwo;
        Card cardThree;
        Card cardFour;
        for (int i= 0; i < 13; i++) {




            switch (i) {
            case 0:
                cardOne = new Card("Two of Hearts", 2);
                cards.add(cardOne);

                cardTwo = new Card("Two of Spades", 2);
                cards.add(cardTwo);

                cardThree = new Card("Two of Clubs", 2);
                cards.add(cardThree);

                cardFour = new Card("Two of Diamonds", 2);
                cards.add(cardFour);
                     break;
```

```java
package cardGameWar;

import java.util.ArrayList;

public class Hand {
    private ArrayList<Card> hand;

    public Hand() {
        hand = new ArrayList<Card>();

    }
    public void clear() {
        hand.clear();

    }

    public void addCard(Card card) {
    }

    public void removeCard(Card card) {
}

    public void sortByValue() {

    }
}
```

```java
package cardGameWar;

import java.util.ArrayList;

public class Player {

    private String name;
    private int score;
    public List<Card> hand = new ArrayList<Card>();


    public Player(String name) {
        this.setName(name);
        score = 0;

}

    public String getName() {
        return name;
    }

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

    public Card flip() {
        return hand.remove(0);
    }

    public void draw(Deck deck) {
        hand.add(deck.draw());
    }
    public void describe() {
        System.out.println(name);
        for(Card card : hand)
```

Screenshots of Running Application:

```
Winner: Jack
Player One Score: 7
Player Two Score: 19
```



URL to GitHub Repository:

https://github.com/bmason1969/Week6CodingAssignment.git