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:

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

```java
package war_game;

//Question 1a. Create Card class.
public class Card {

    //i. Define value and name fields for Card.
    private int value;
    private String name;

    //ii. Define card methods.
    //1. getters and setters
    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;
    }

    //2. describe method for printing card info
    public String describe() {
        return name;
    }

    //Constructor used to build Deck.
    public Card(int value, int suit) {
        this.value = value;
        switch(value) {
        case 2:
            switch(suit) {
            case 1:
                name = "Two of Spades";
                break;
```

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

```
                break;
            case 2:
                name = "Two of Clubs";
                break;
            case 3:
                name = "Two of Diamonds";;
                break;
            case 4:
                name = "Two of Hearts";;
                break;
            }
            break;

        case 3:
            switch(suit) {
            case 1:
                name = "Three of Spades";
                break;
            case 2:
                name = "Three of Clubs";
                break;
            case 3:
                name = "Three of Diamonds";;
                break;
            case 4:
                name = "Three of Hearts";;
                break;
            }
            break;

        case 4:
            switch(suit) {
            case 1:
                name = "Four of Spades";
                break;
            case 2:
                name = "Four of Clubs";
                break;
```

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

```java
                break;
            case 3:
                name = "Four of Diamonds";;
                break;
            case 4:
                name = "Four of Hearts";;
                break;
            }
            break;

        case 5:
            switch(suit) {
            case 1:
                name = "Five of Spades";
                break;
            case 2:
                name = "Five of Clubs";
                break;
            case 3:
                name = "Five of Diamonds";;
                break;
            case 4:
                name = "Five of Hearts";;
                break;
            }
            break;

        case 6:
            switch(suit) {
            case 1:
                name = "Six of Spades";
                break;
            case 2:
                name = "Six of Clubs";
                break;
            case 3:
                name = "Six of Diamonds";;
                break;
```

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

```java
                break;
            case 4:
                name = "Six of Hearts";;
                break;
            }
            break;

        case 7:
            switch(suit) {
            case 1:
                name = "Seven of Spades";
                break;
            case 2:
                name = "Seven of Clubs";
                break;
            case 3:
                name = "Seven of Diamonds";;
                break;
            case 4:
                name = "Seven of Hearts";;
                break;
            }
            break;

        case 8:
            switch(suit) {
            case 1:
                name = "Eight of Spades";
                break;
            case 2:
                name = "Eight of Clubs";
                break;
            case 3:
                name = "Eight of Diamonds";;
                break;
            case 4:
                name = "Eight of Hearts";;
                break;
```

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

```java
                break;
            }
            break;

        case 9:
            switch(suit) {
            case 1:
                name = "Nine of Spades";
                break;
            case 2:
                name = "Nine of Clubs";
                break;
            case 3:
                name = "Nine of Diamonds";;
                break;
            case 4:
                name = "Nine of Hearts";;
                break;
            }
            break;

        case 10:
            switch(suit) {
            case 1:
                name = "Ten of Spades";
                break;
            case 2:
                name = "Ten of Clubs";
                break;
            case 3:
                name = "Ten of Diamonds";;
                break;
            case 4:
                name = "Ten of Hearts";;
                break;
            }
            break;
```

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

```java
            }
            break;

        case 11:
            switch(suit) {
            case 1:
                name = "Jack of Spades";
                break;
            case 2:
                name = "Jack of Clubs";
                break;
            case 3:
                name = "Jack of Diamonds";;
                break;
            case 4:
                name = "Jack of Hearts";;
                break;
            }
            break;

        case 12:
            switch(suit) {
            case 1:
                name = "Queen of Spades";
                break;
            case 2:
                name = "Queen of Clubs";
                break;
            case 3:
                name = "Queen of Diamonds";;
                break;
            case 4:
                name = "Queen of Hearts";;
                break;
            }
            break;

        case 13:
```

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

```java

        case 13:
            switch(suit) {
            case 1:
                name = "King of Spades";
                break;
            case 2:
                name = "King of Clubs";
                break;
            case 3:
                name = "King of Diamonds";;
                break;
            case 4:
                name = "King of Hearts";;
                break;
            }
            break;

        case 14:
            switch(suit) {
            case 1:
                name = "Ace of Spades";
                break;
            case 2:
                name = "Ace of Clubs";
                break;
            case 3:
                name = "Ace of Diamonds";;
                break;
            case 4:
                name = "Ace of Hearts";;
                break;
            }
            break;
        }


    }
```

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

```java
package war_game;

import java.util.*;

//Question 1b. Create Deck class.
public class Deck {

    //i. Declare list of Card named cards
    private List<Card> cards = new ArrayList<Card>();

    //ii. Declare Deck methods.

    //1. Method that shuffles the Deck.
    public void shuffle() {
        Collections.shuffle(cards);
    }

    //2. Method that draws and removes top card from Deck.
    public Card draw() {
        return cards.remove(0);
    }

    //3. Constructor that instantiates Deck and fills it with 52 cards.
    public Deck() {
        for (int value = 2; value <= 14; value++) {
            for (int suit = 1; suit <= 4; suit++) {
                Card card = new Card(value, suit);
                cards.add(card);
            }
        }
    }
}
```

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

```java
package war_game;

import java.util.*;

//Question 1c. Create the Player class.
public class Player {

    //i. Declare variables associated to Player class.
    private List<Card> hand = new ArrayList<Card>();
    private String name;
    private int score;

    //ii. Declare methods associated to Player class.
    //1. Print out player name and cards in player's hand
    public void describe() {
        System.out.println(name + "'s Hand:");
        for (int i = 0; i < 26; i++) {
          System.out.println(hand.get(i).describe());
        }
    }

    //2. Remove and return the top card in player's hand.
    public Card flip() {
        return hand.remove(0);
    }

    //3. Calls the draw method on the deck and adds Card to hand.
    public void draw(Deck deck) {
        hand.add(deck.draw());
    }

    //4. Add 1 to player score.
    public void incrementScore() {
        setScore(getScore() + 1);
    }
```

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

```java
    //2. Remove and return the top card in player's hand.
    public Card flip() {
        return hand.remove(0);
    }

    //3. Calls the draw method on the deck and adds Card to hand.
    public void draw(Deck deck) {
        hand.add(deck.draw());
    }

    //4. Add 1 to player score.
    public void incrementScore() {
        setScore(getScore() + 1);
    }

    //Constructor
    public Player(String name) {
        this.setScore(0);
        this.name = name;
    }

    //getters and setters
    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;
    }
}
```

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

```java
package war_game;

//Question 2. Create class named App
public class App {

    public static void main(String[] args) {

        //Question 3. Instantiate a Deck and two Players.
        Deck deck = new Deck();
        Player player1 = new Player("Garrett");
        Player player2 = new Player("War Machine");

        //Shuffle the Deck.
        deck.shuffle();

        //Question 4 Draw Player hands.
        for (int i = 0; i < 52; i++) {
            if (i < 26) {
                player1.draw(deck);
            } else if (i >= 26 && i < 52) {
                player2.draw(deck);
            }
        }

        //Show player hands
        System.out.println("*************************************************");
        System.out.println("*****************Player Hands*****************");
        System.out.println("*************************************************");
        player1.describe();
        System.out.println("*************************************************");
        player2.describe();
```

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

```java
        //Show player hands
        System.out.println("*************************************************");
        System.out.println("******************Player Hands******************");
        System.out.println("*************************************************");
        player1.describe();
        System.out.println("*************************************************");
        player2.describe();

        //Question 5. Flip each player's hand.
        System.out.println("*************************************************");
        System.out.println("********************Matchups*********************");
        System.out.println("*************************************************");
        for (int i = 0; i < 26; i++) {
            Card player1Card = player1.flip();
            Card player2Card = player2.flip();
            System.out.println(player1Card.getName() + " vs " + player2Card.getName());
            if (player1Card.getValue() < player2Card.getValue()) {
                player2.incrementScore();
            } else if (player1Card.getValue() > player2Card.getValue()) {
                player1.incrementScore();
            }
        }

        //Question 6 & 7 Compare final scores and print final scores and winner if there is one.
        System.out.println("*************************************************");
        System.out.println("****************Final Result*******************");
        System.out.println("*************************************************");
        System.out.println(player1.getName() + "'s score: " + player1.getScore());
        System.out.println(player2.getName() + "'s score: " + player2.getScore());
        if (player1.getScore() > player2.getScore()) {
          System.out.println(player1.getName() + " Wins!");
        } else if (player1.getScore() < player2.getScore()) {
          System.out.println(player2.getName() + " Wins!");
        } else {
          System.out.println("It's a draw!");
        }
```

Screenshots of Running Application:

```
Problems  Javadoc  Declaration  Console  Terminal
<terminated> App (1) [Java Application] C:\Program Files\Java\jdk-11.0.15\bin\javaw.exe  (Jul 30, 2022, 2:22:59 PM – 2:22:59 PM) [pid: 28108]
************************************************
*****************Player Hands*****************
************************************************
Garrett's Hand:
Four of Hearts
Three of Spades
Three of Hearts
Seven of Hearts
Jack of Spades
Ace of Clubs
Queen of Diamonds
King of Diamonds
Nine of Diamonds
Nine of Clubs
Jack of Diamonds
Seven of Clubs
Six of Hearts
Three of Clubs
Queen of Spades
Four of Spades
Nine of Spades
Ten of Clubs
Two of Clubs
Nine of Hearts
Four of Clubs
Six of Spades
King of Clubs
Six of Diamonds
Two of Hearts
Ace of Diamonds
************************************************
War Machine's Hand:
Jack of Clubs
Eight of Hearts
Two of Spades
Ace of Spades
King of Hearts
Four of Diamonds
Two of Diamonds
Five of Hearts
Six of Clubs
Eight of Diamonds
Eight of Clubs
King of Spades
Ten of Hearts
Seven of Spades
Ten of Diamonds
Jack of Hearts
Queen of Hearts
Seven of Diamonds
Five of Clubs
```

```
Problems  Javadoc  Declaration  Console  Terminal
<terminated> App (1) [Java Application] C:\Program Files\Java\jdk-11.0.15\bin\javaw.exe (Jul 30, 2022, 2:22:59 PM – 2:22:59 PM) [pid: 28108]
King of Spades
Ten of Hearts
Seven of Spades
Ten of Diamonds
Jack of Hearts
Queen of Hearts
Seven of Diamonds
Five of Clubs
Three of Diamonds
Ace of Hearts
Five of Spades
Eight of Spades
Five of Diamonds
Queen of Clubs
Ten of Spades
************************************************
*******************Matchups*******************
************************************************
Four of Hearts vs Jack of Clubs
Three of Spades vs Eight of Hearts
Three of Hearts vs Two of Spades
Seven of Hearts vs Ace of Spades
Jack of Spades vs King of Hearts
Ace of Clubs vs Four of Diamonds
Queen of Diamonds vs Two of Diamonds
King of Diamonds vs Five of Hearts
Nine of Diamonds vs Six of Clubs
Nine of Clubs vs Eight of Diamonds
Jack of Diamonds vs Eight of Clubs
Seven of Clubs vs King of Spades
Six of Hearts vs Ten of Hearts
Three of Clubs vs Seven of Spades
Queen of Spades vs Ten of Diamonds
Four of Spades vs Jack of Hearts
Nine of Spades vs Queen of Hearts
Ten of Clubs vs Seven of Diamonds
Two of Clubs vs Five of Clubs
Nine of Hearts vs Three of Diamonds
Four of Clubs vs Ace of Hearts
Six of Spades vs Five of Spades
King of Clubs vs Eight of Spades
Six of Diamonds vs Five of Diamonds
Two of Hearts vs Queen of Clubs
Ace of Diamonds vs Ten of Spades
************************************************
****************Final Result******************
************************************************
Garrett's score: 14
War Machine's score: 12
Garrett Wins!
```

URL to GitHub Repository:

https://github.com/gyarbrough/War-Game