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

      a.i. Fields

         a.i.1. **value** (contains a value from 2-14 representing cards 2-Ace)

         a.i.2. **name** (e.g. Ace of Diamonds, or Two of Hearts)

      a.ii. Methods

         a.ii.1. Getters and Setters

         a.ii.2. **describe** (prints out information about a card)

   b. Deck

b.i. Fields

b.i.1. **cards** (List of Card)

b.ii. Methods

b.ii.1. **shuffle** (randomizes the order of the cards)

b.ii.2. **draw** (removes and returns the top card of the Cards field)

b.ii.3. In the constructor, when a new Deck is instantiated, the Cards field should be populated with the standard 52 cards.

c. Player

c.i. Fields

c.i.1. **hand** (List of Card)

c.i.2. **score** (set to 0 in the constructor)

c.i.3. **name**

c.ii. Methods

c.ii.1. **describe** (prints out information about the player and calls the describe method for each card in the Hand List)

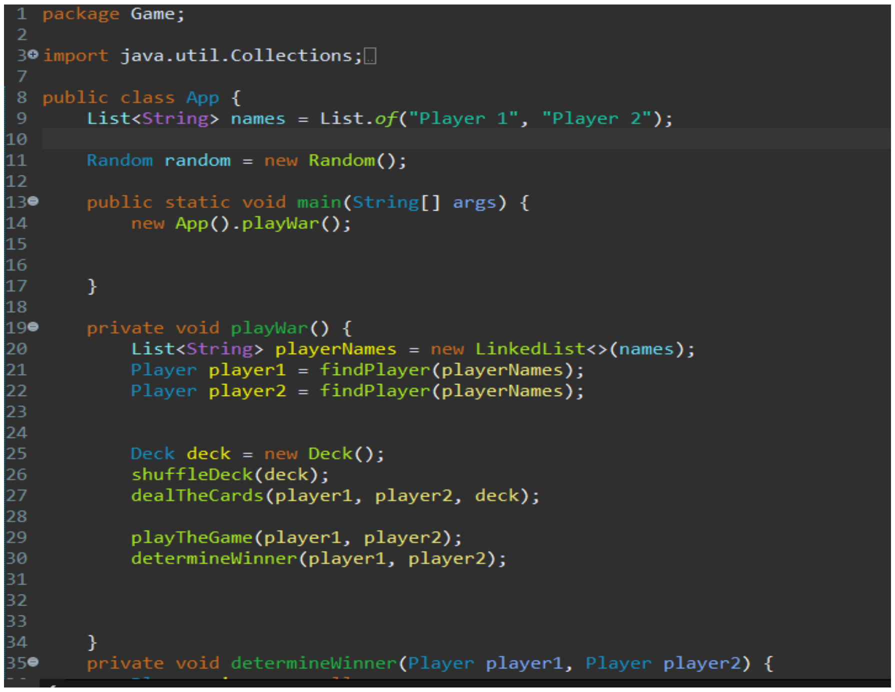
c.ii.2. **flip** (removes and returns the top card of the Hand)

c.ii.3. **draw** (takes a Deck as an argument and calls the draw method on the deck, adding the returned Card to the hand field)

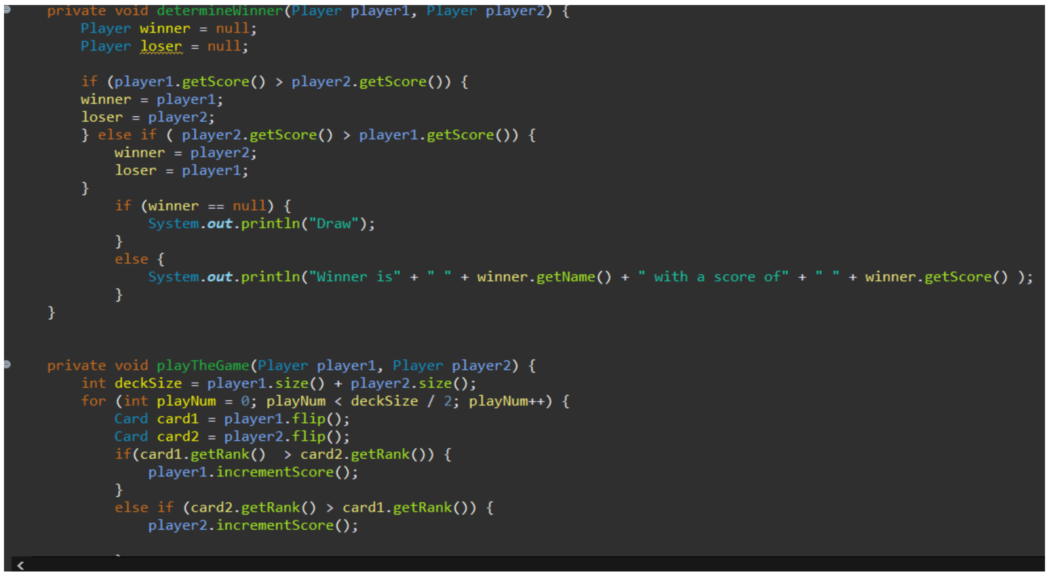
c.ii.4. **incrementScore** (adds 1 to the Player's score field)

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.

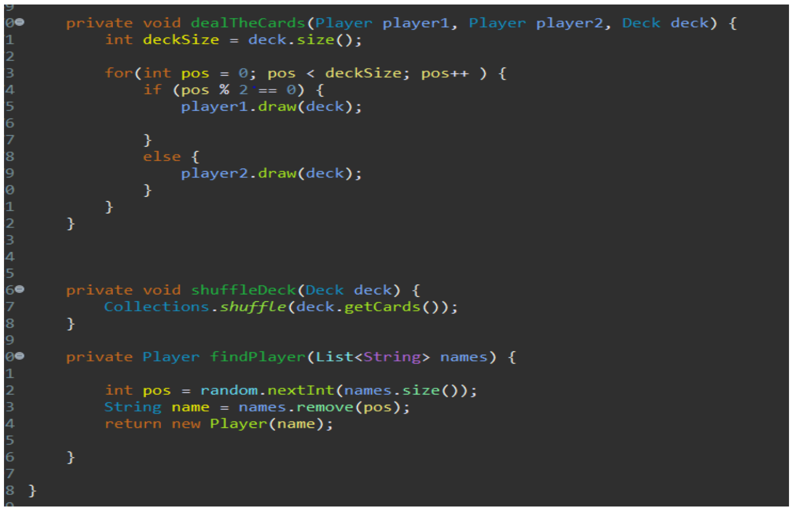
Screenshots of Code:



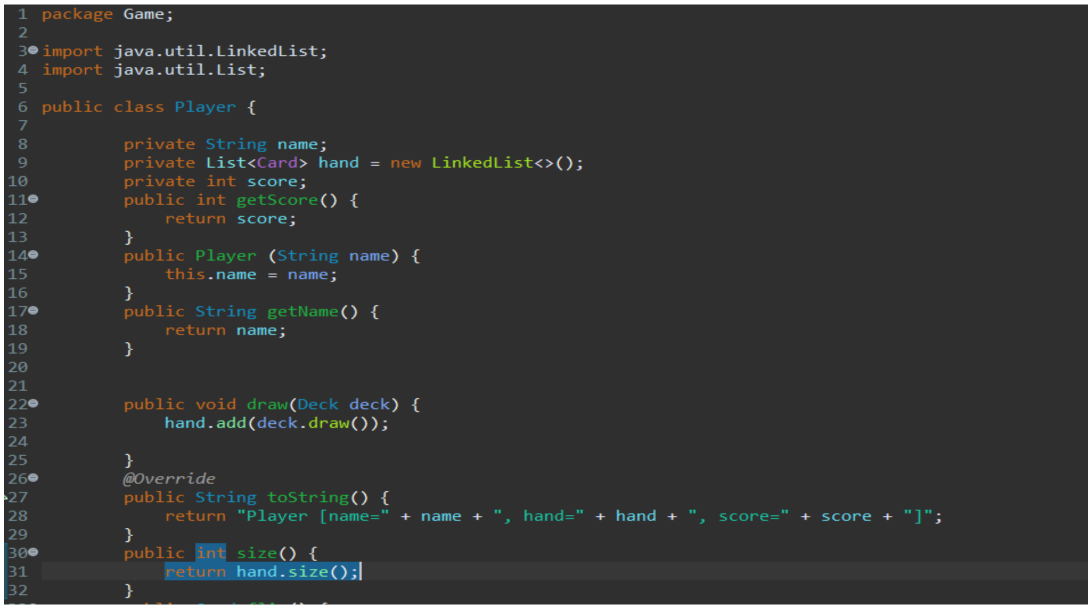
```java
package Game;

import java.util.Collections;

public class App {
    List<String> names = List.of("Player 1", "Player 2");

    Random random = new Random();

    public static void main(String[] args) {
        new App().playWar();


    }

    private void playWar() {
        List<String> playerNames = new LinkedList<>(names);
        Player player1 = findPlayer(playerNames);
        Player player2 = findPlayer(playerNames);


        Deck deck = new Deck();
        shuffleDeck(deck);
        dealTheCards(player1, player2, deck);

        playTheGame(player1, player2);
        determineWinner(player1, player2);



    }
    private void determineWinner(Player player1, Player player2) {
```



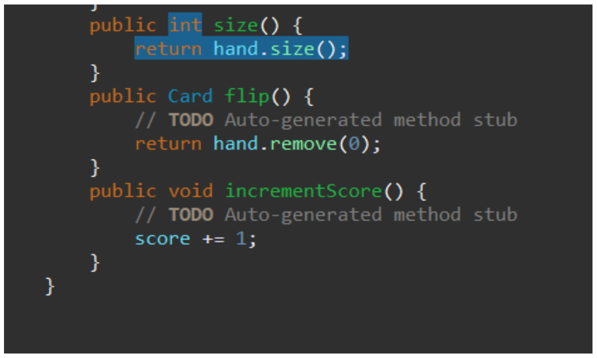
```java
private void determineWinner(Player player1, Player player2) {
    Player winner = null;
    Player loser = null;

    if (player1.getScore() > player2.getScore()) {
    winner = player1;
    loser = player2;
    } else if ( player2.getScore() > player1.getScore()) {
        winner = player2;
        loser = player1;
    }
        if (winner == null) {
            System.out.println("Draw");
        }
        else {
            System.out.println("Winner is" + " " + winner.getName() + " with a score of" + " " + winner.getScore() );
        }
}


private void playTheGame(Player player1, Player player2) {
    int deckSize = player1.size() + player2.size();
    for (int playNum = 0; playNum < deckSize / 2; playNum++) {
        Card card1 = player1.flip();
        Card card2 = player2.flip();
        if(card1.getRank()  > card2.getRank()) {
            player1.incrementScore();
        }
        else if (card2.getRank() > card1.getRank()) {
            player2.incrementScore();
```



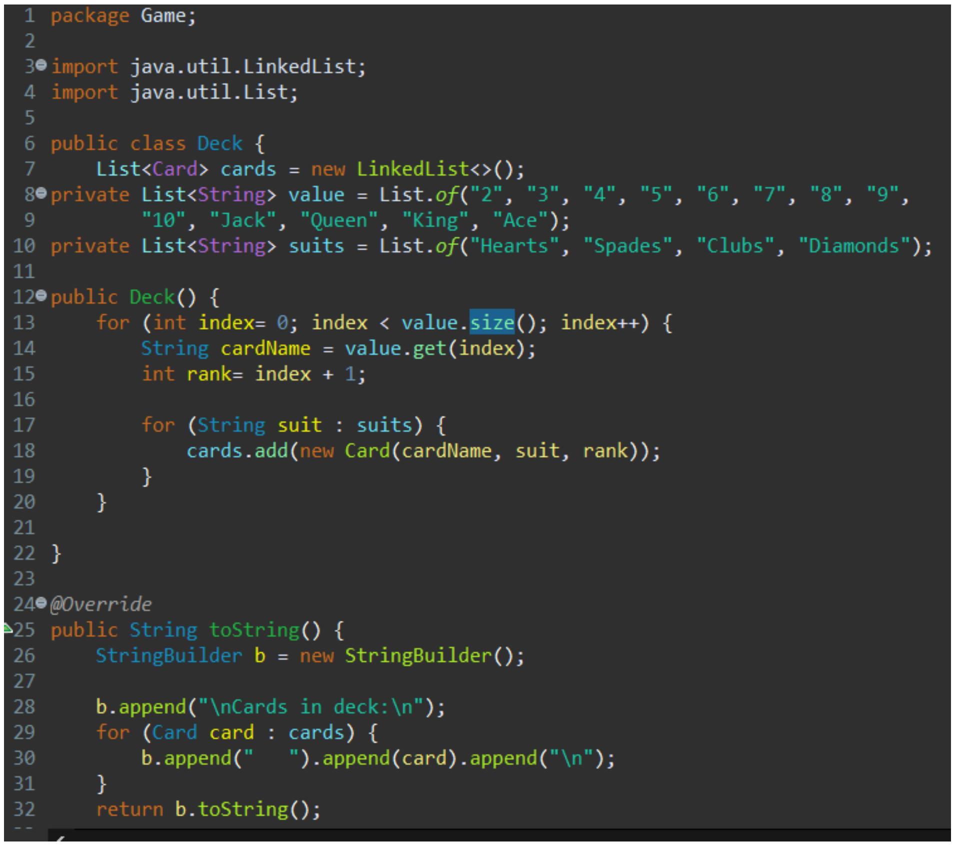
```java
    private void dealTheCards(Player player1, Player player2, Deck deck) {
        int deckSize = deck.size();

        for(int pos = 0; pos < deckSize; pos++ ) {
            if (pos % 2 == 0) {
                player1.draw(deck);

            }
            else {
                player2.draw(deck);
            }
        }
    }


    private void shuffleDeck(Deck deck) {
        Collections.shuffle(deck.getCards());
    }

    private Player findPlayer(List<String> names) {

        int pos = random.nextInt(names.size());
        String name = names.remove(pos);
        return new Player(name);

    }

}
```



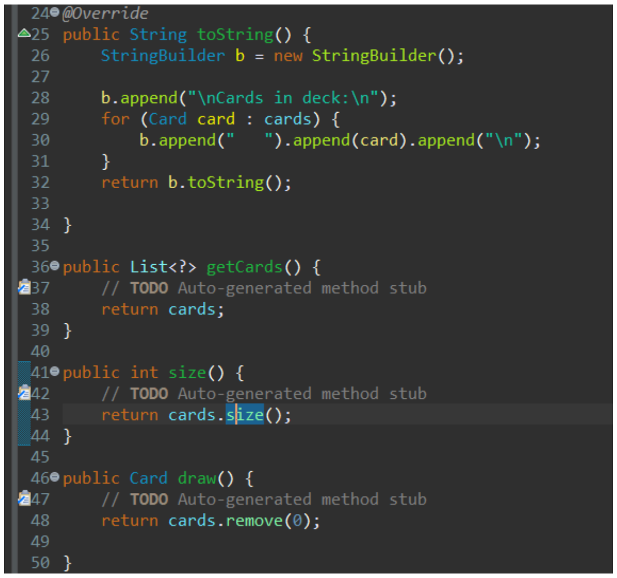
```java
package Game;

import java.util.LinkedList;
import java.util.List;

public class Player {

        private String name;
        private List<Card> hand = new LinkedList<>();
        private int score;
        public int getScore() {
            return score;
        }
        public Player (String name) {
            this.name = name;
        }
        public String getName() {
            return name;
        }


        public void draw(Deck deck) {
            hand.add(deck.draw());

        }
        @Override
        public String toString() {
            return "Player [name=" + name + ", hand=" + hand + ", score=" + score + "]";
        }
        public int size() {
            return hand.size();
        }
```



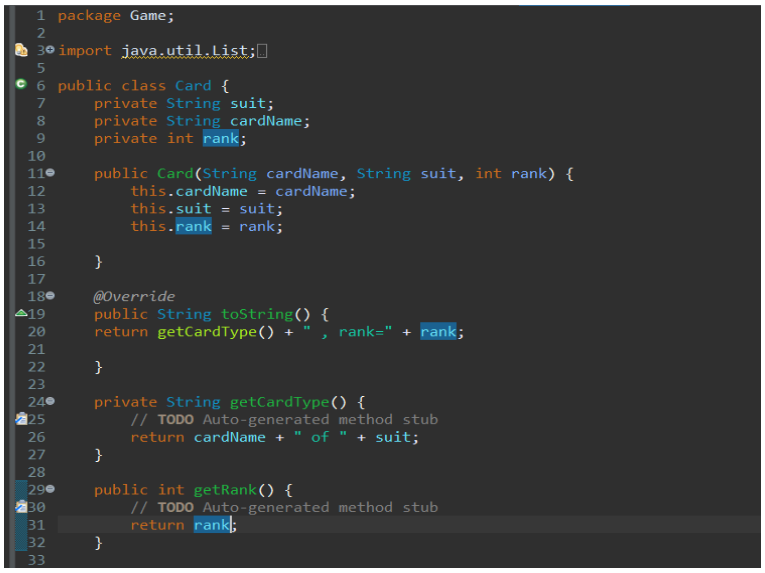
```java
        public int size() {
            return hand.size();
        }
        public Card flip() {
            // TODO Auto-generated method stub
            return hand.remove(0);
        }
        public void incrementScore() {
            // TODO Auto-generated method stub
            score += 1;
        }
    }
```



```java
package Game;

import java.util.LinkedList;
import java.util.List;

public class Deck {
    List<Card> cards = new LinkedList<>();
private List<String> value = List.of("2", "3", "4", "5", "6", "7", "8", "9",
        "10", "Jack", "Queen", "King", "Ace");
private List<String> suits = List.of("Hearts", "Spades", "Clubs", "Diamonds");

public Deck() {
    for (int index= 0; index < value.size(); index++) {
        String cardName = value.get(index);
        int rank= index + 1;

        for (String suit : suits) {
            cards.add(new Card(cardName, suit, rank));
        }
    }

}

@Override
public String toString() {
    StringBuilder b = new StringBuilder();

    b.append("\nCards in deck:\n");
    for (Card card : cards) {
        b.append("    ").append(card).append("\n");
    }
    return b.toString();
```

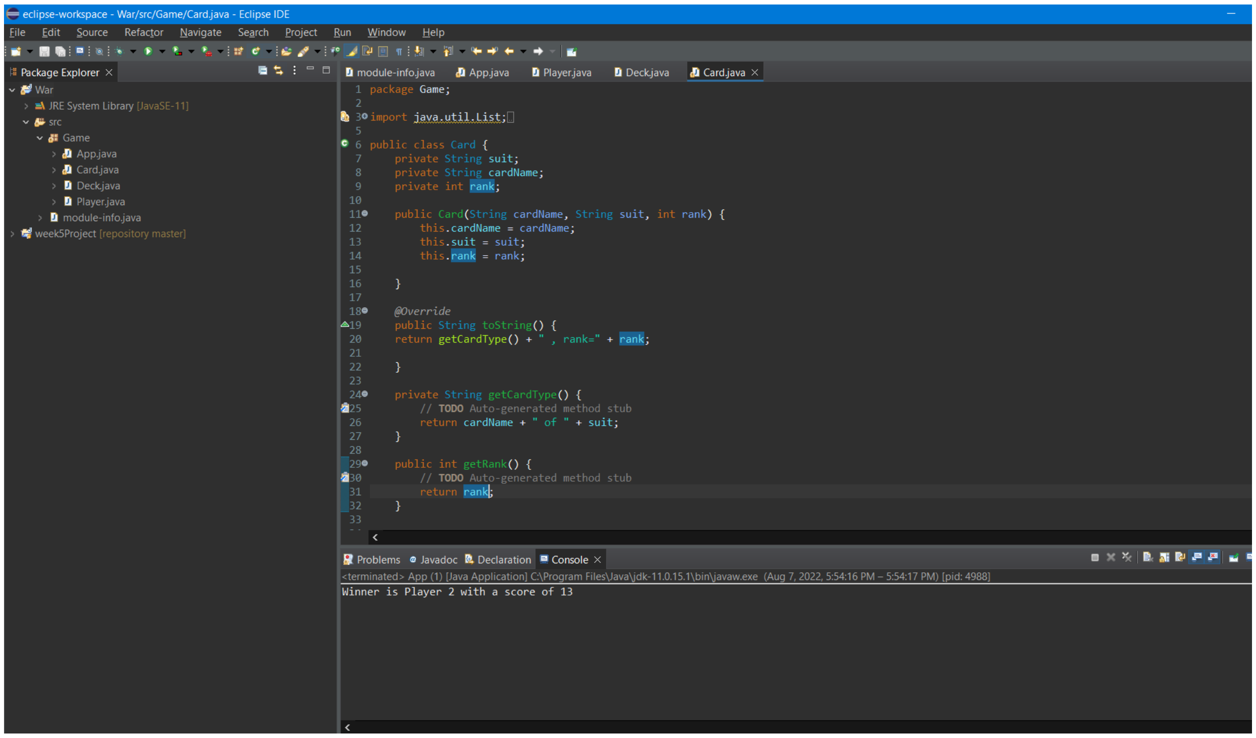


```java
@Override
public String toString() {
    StringBuilder b = new StringBuilder();

    b.append("\nCards in deck:\n");
    for (Card card : cards) {
        b.append("    ").append(card).append("\n");
    }
    return b.toString();

}

public List<?> getCards() {
    // TODO Auto-generated method stub
    return cards;
}

public int size() {
    // TODO Auto-generated method stub
    return cards.size();
}

public Card draw() {
    // TODO Auto-generated method stub
    return cards.remove(0);

}
```



```java
package Game;

import java.util.List;

public class Card {
    private String suit;
    private String cardName;
    private int rank;

    public Card(String cardName, String suit, int rank) {
        this.cardName = cardName;
        this.suit = suit;
        this.rank = rank;

    }

    @Override
    public String toString() {
    return getCardType() + " , rank=" + rank;

    }

    private String getCardType() {
        // TODO Auto-generated method stub
        return cardName + " of " + suit;
    }

    public int getRank() {
        // TODO Auto-generated method stub
        return rank;
    }

```

Screenshots of Running Application:


eclipse-workspace - War/src/Game/Card.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
War
JRE System Library [JavaSE-11]
src
Game
App.java
Card.java
Deck.java
Player.java
module-info.java
week5Project [repository master]
module-info.java App.java Player.java Deck.java Card.java
Problems Javadoc Declaration Console

```java
package Game;

import java.util.List;

public class Card {
    private String suit;
    private String cardName;
    private int rank;

    public Card(String cardName, String suit, int rank) {
        this.cardName = cardName;
        this.suit = suit;
        this.rank = rank;

    }

    @Override
    public String toString() {
    return getCardType() + " , rank=" + rank;

    }

    private String getCardType() {
        // TODO Auto-generated method stub
        return cardName + " of " + suit;
    }

    public int getRank() {
        // TODO Auto-generated method stub
        return rank;
    }

```

```
<terminated> App (1) [Java Application] C:\Program Files\Java\jdk-11.0.15.1\bin\javaw.exe  (Aug 7, 2022, 5:54:16 PM – 5:54:17 PM) [pid: 4988]
Winner is Player 2 with a score of 13
```

URL to GitHub Repository:

https://github.com/ddidonato93/week6War