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*.

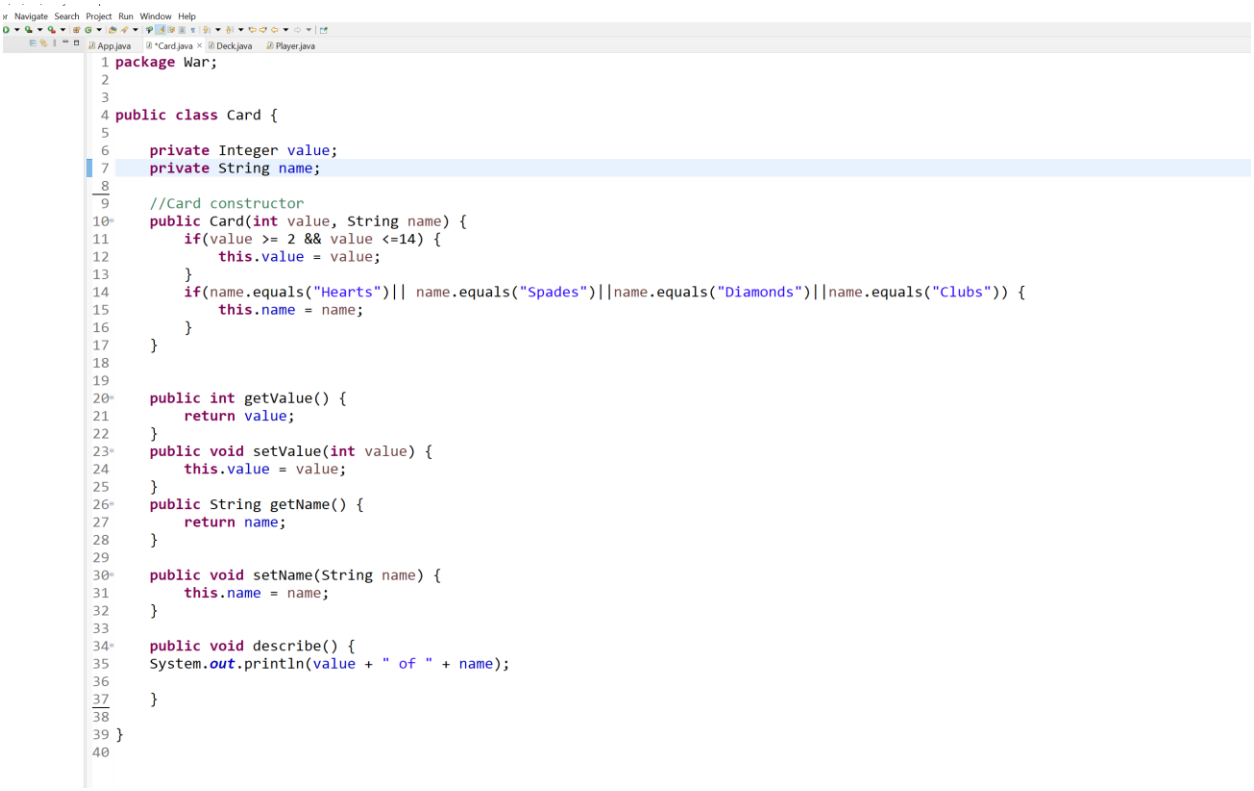
1. Create the following classes.
    a. Card
        i. Fields
            1. **value** (contains a value from 2-14 representing cards 2-Ace)
            2. **name** (e.g. Ace of Diamonds, or Two of Hearts)
        ii. Methods
            1. Getters and Setters
            2. **describe** (prints out 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:

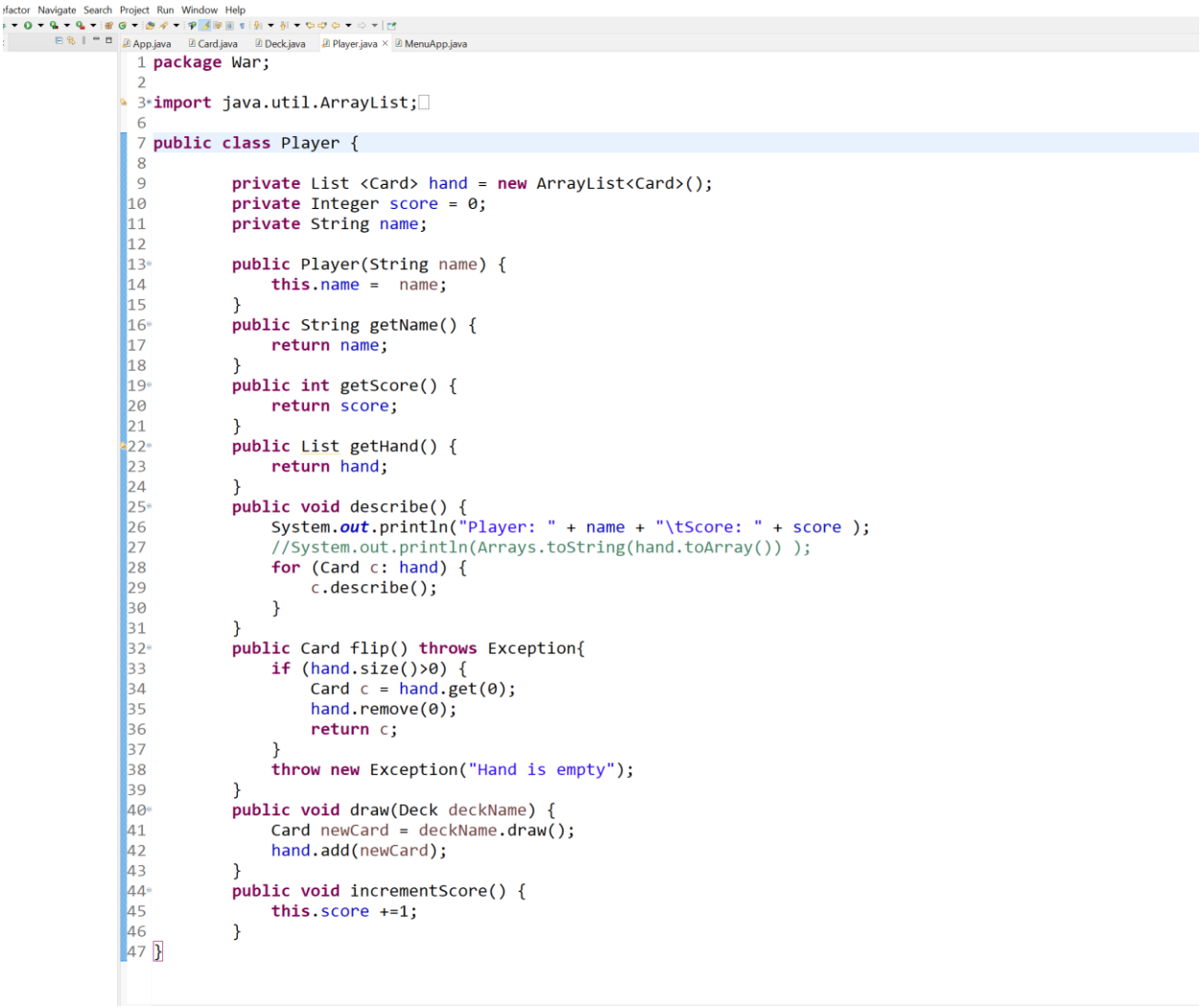



```java
package War;


public class Card {

    private Integer value;
    private String name;

    //Card constructor
    public Card(int value, String name) {
        if(value >= 2 && value <=14) {
            this.value = value;
        }
        if(name.equals("Hearts")|| name.equals("Spades")||name.equals("Diamonds")||name.equals("Clubs")) {
            this.name = name;
        }
    }


    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(value + " of " + name);

    }

}

```

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

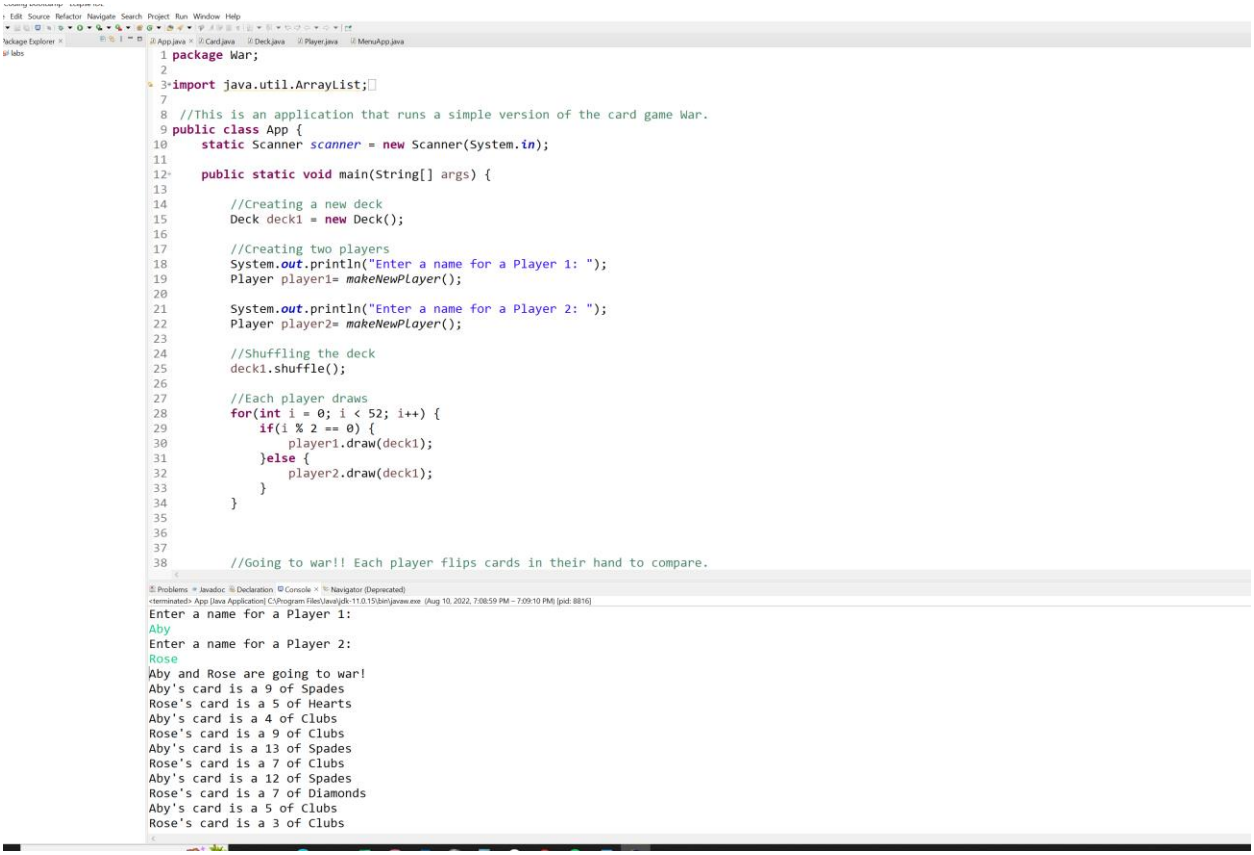
```java
package War;

import java.util.ArrayList;

public class Deck {
    private List <Card> cards = new ArrayList <Card>();
    private String name;

    //Deck constructor
    public Deck() {

        for(int i = 2; i<=14; i ++) {
          for(int j = 0; j<= 4; j++) {
            if (j==0) {
                cards.add(new Card(i,"Hearts"));
            }else if(j==1) {
                cards.add(new Card (i,"Spades"));
            }else if(j==2) {
                cards.add(new Card(i,"Clubs"));
            }else if (j==3){
                cards.add(new Card(i,"Diamonds"));
            }
          }
        }
}
    //method to check if deck is actually 52 cards, if needed
    public void checkDeckSize() {
        System.out.println(cards.size());
    }
    public void shuffle() {
        Collections.shuffle(cards, new Random());
    }
    public Card draw() {

        if (cards.size()>0) {
            Card c = cards.get(0);
            cards.remove(0);
            return c;
        }
        return null;
    }
    public void describeCards() {
        for (Card c: cards) {
            c.describe();
        }
    }
    //This describe method is redundant, but nostalgic
```

Problems | Javadoc | Declaration | Console | Navigator (Deprecated)



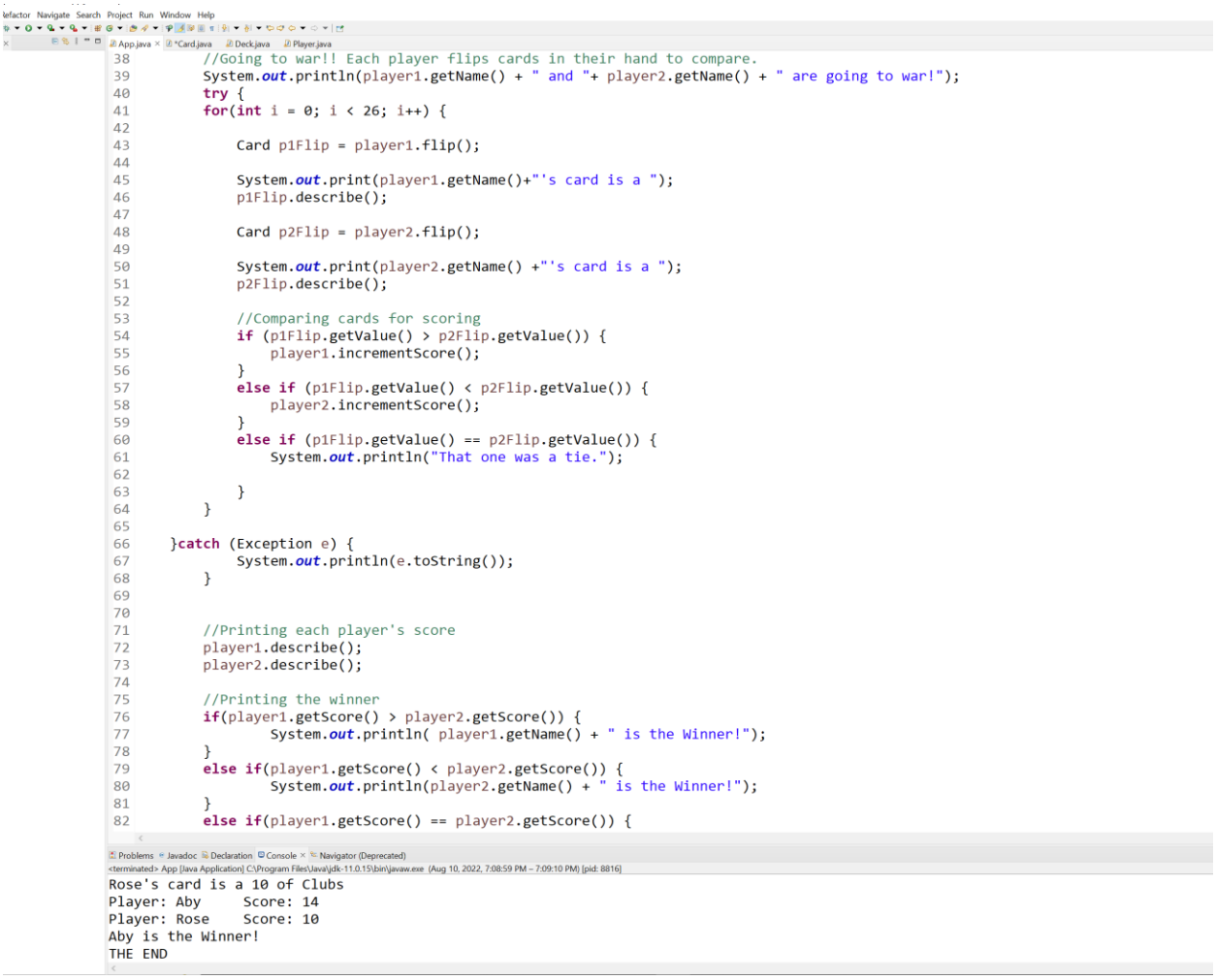

```java
package War;

import java.util.ArrayList;

public class Player {

        private List <Card> hand = new ArrayList<Card>();
        private Integer score = 0;
        private String name;

        public Player(String name) {
            this.name =  name;
        }
        public String getName() {
            return name;
        }
        public int getScore() {
            return score;
        }
        public List getHand() {
            return hand;
        }
        public void describe() {
            System.out.println("Player: " + name + "\tScore: " + score );
            //System.out.println(Arrays.toString(hand.toArray()) );
            for (Card c: hand) {
                c.describe();
            }
        }
        public Card flip() throws Exception{
            if (hand.size()>0) {
                Card c = hand.get(0);
                hand.remove(0);
                return c;
            }
            throw new Exception("Hand is empty");
        }
        public void draw(Deck deckName) {
            Card newCard = deckName.draw();
            hand.add(newCard);
        }
        public void incrementScore() {
            this.score +=1;
        }
}
```

Screenshots of Running Application:



```java
package War;

import java.util.ArrayList;

//This is an application that runs a simple version of the card game War.
public class App {
    static Scanner scanner = new Scanner(System.in);

    public static void main(String[] args) {

        //Creating a new deck
        Deck deck1 = new Deck();

        //Creating two players
        System.out.println("Enter a name for a Player 1: ");
        Player player1= makeNewPlayer();

        System.out.println("Enter a name for a Player 2: ");
        Player player2= makeNewPlayer();

        //Shuffling the deck
        deck1.shuffle();

        //Each player draws
        for(int i = 0; i < 52; i++) {
            if(i % 2 == 0) {
                player1.draw(deck1);
            }else {
                player2.draw(deck1);
            }
        }



        //Going to war!! Each player flips cards in their hand to compare.
```

```
Enter a name for a Player 1:
Aby
Enter a name for a Player 2:
Rose
Aby and Rose are going to war!
Aby's card is a 9 of Spades
Rose's card is a 5 of Hearts
Aby's card is a 4 of Clubs
Rose's card is a 9 of Clubs
Aby's card is a 13 of Spades
Rose's card is a 7 of Clubs
Aby's card is a 12 of Spades
Rose's card is a 7 of Diamonds
Aby's card is a 5 of Clubs
Rose's card is a 3 of Clubs
```



```
        //Going to war!! Each player flips cards in their hand to compare.
        System.out.println(player1.getName() + " and "+ player2.getName() + " are going to war!");
        try {
        for(int i = 0; i < 26; i++) {

            Card p1Flip = player1.flip();

            System.out.print(player1.getName()+"'s card is a ");
            p1Flip.describe();

            Card p2Flip = player2.flip();

            System.out.print(player2.getName() +"'s card is a ");
            p2Flip.describe();

            //Comparing cards for scoring
            if (p1Flip.getValue() > p2Flip.getValue()) {
                player1.incrementScore();
            }
            else if (p1Flip.getValue() < p2Flip.getValue()) {
                player2.incrementScore();
            }
            else if (p1Flip.getValue() == p2Flip.getValue()) {
                System.out.println("That one was a tie.");

            }
        }

    }catch (Exception e) {
            System.out.println(e.toString());
        }


        //Printing each player's score
        player1.describe();
        player2.describe();

        //Printing the winner
        if(player1.getScore() > player2.getScore()) {
                System.out.println( player1.getName() + " is the Winner!");
        }
        else if(player1.getScore() < player2.getScore()) {
                System.out.println(player2.getName() + " is the Winner!");
        }
        else if(player1.getScore() == player2.getScore()) {
```

```
Rose's card is a 10 of Clubs
Player: Aby     Score: 14
Player: Rose    Score: 10
Aby is the Winner!
THE END
```

```
66      }catch (Exception e) {
67              System.out.println(e.toString());
68          }
69
70
71          //Printing each player's score
72          player1.describe();
73          player2.describe();
74
75          //Printing the winner
76          if(player1.getScore() > player2.getScore()) {
77                  System.out.println( player1.getName() + " is the Winner!");
78          }
79          else if(player1.getScore() < player2.getScore()) {
80                  System.out.println(player2.getName() + " is the Winner!");
81          }
82          else if(player1.getScore() == player2.getScore()) {
83                  System.out.println("That was a draw.");
84          }
85          System.out.println("THE END");
86
87  }
88      //Method that takes scanner input for player name
89      public static Player makeNewPlayer() {
90      String playerName = scanner.nextLine();
91      Player player = new Player(playerName);
92      return player;
93      }
94
95
96  }
97
```

Problems Javadoc Declaration Console Navigator (Deprecated)

<terminated> App [Java Application] C:\Program Files\Java\jdk-11.0.15\bin\javaw.exe (Aug 10, 2022, 7:08:59 PM – 7:09:10 PM) [pid: 8816]

```
Rose's card is a 10 of Clubs
Player: Aby     Score: 14
Player: Rose    Score: 10
```

URL to GitHub Repository:

https://github.com/Hughes405/Week-6-Project.git