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.

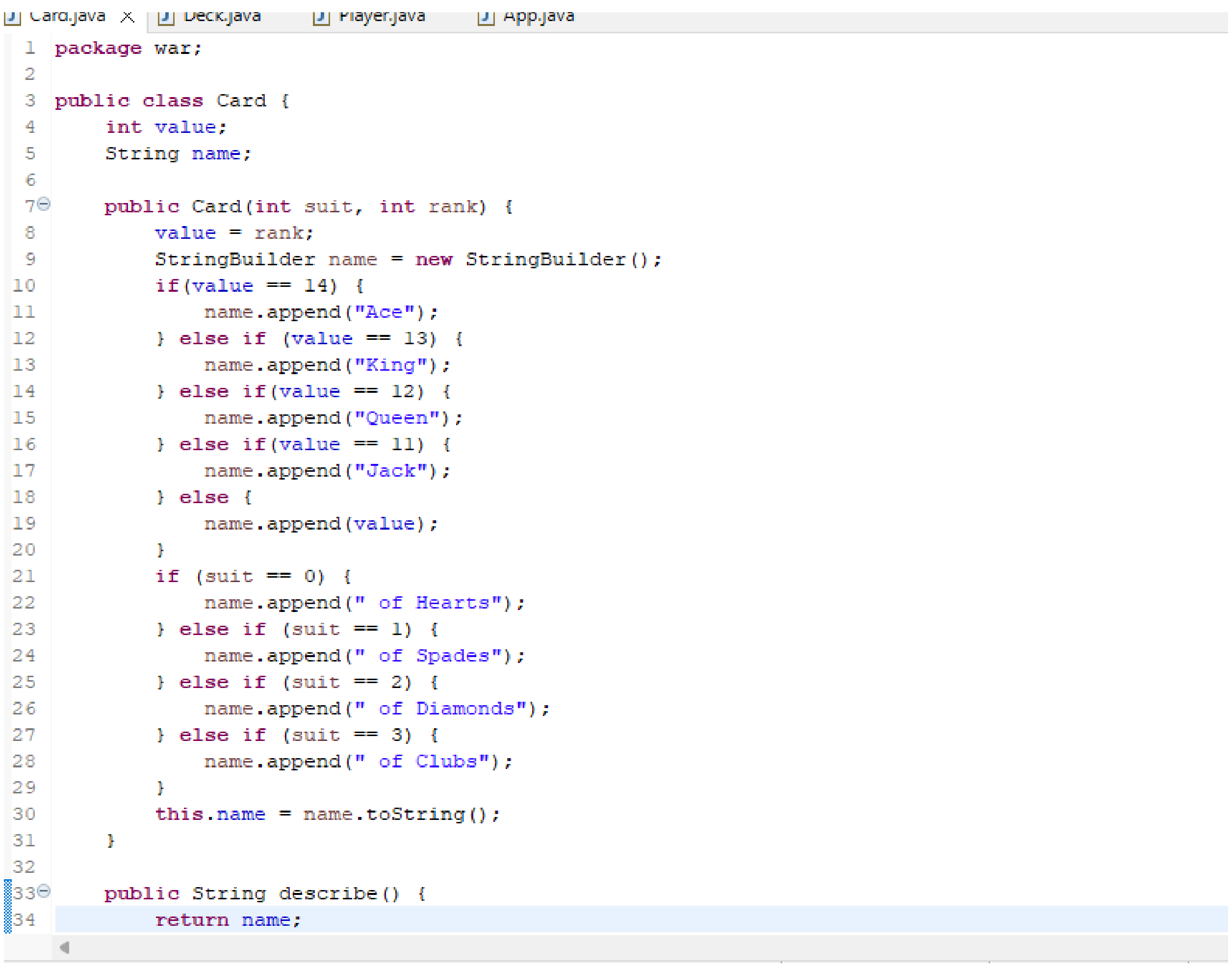
Coding Steps:

For the final project you will be creating an automated version of the classic card game *WAR*.

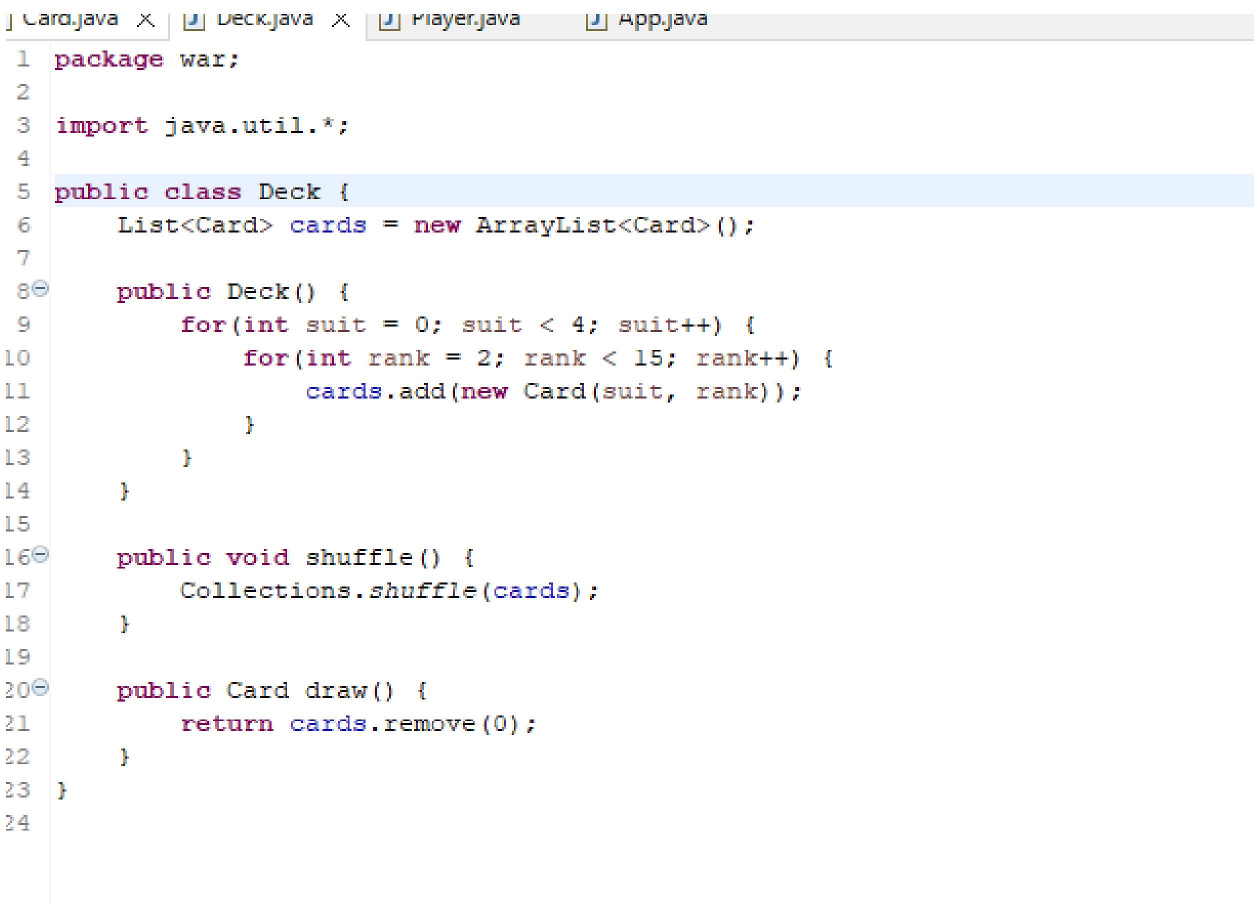
1. Create the following classes.
    a. Card
        i. Fields
            1. **value** (contains a value from 2-14 representing cards 2-Ace)
            2. **name** (e.g. Ace of Diamonds, or Two of Hearts)
        ii. Methods
            1. Getters and Setters
            2. **describe** (prints out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

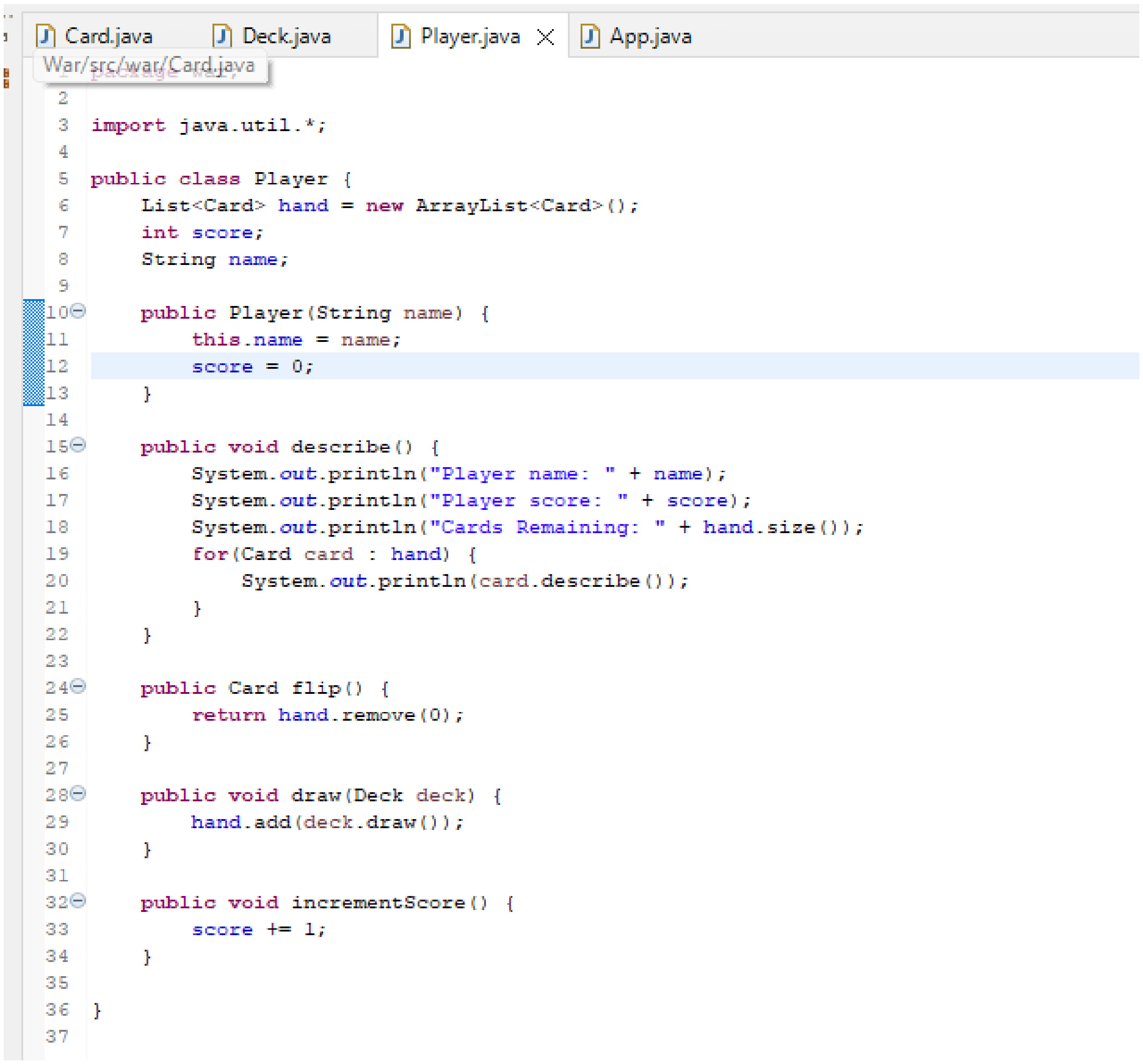
Screenshots of Code:



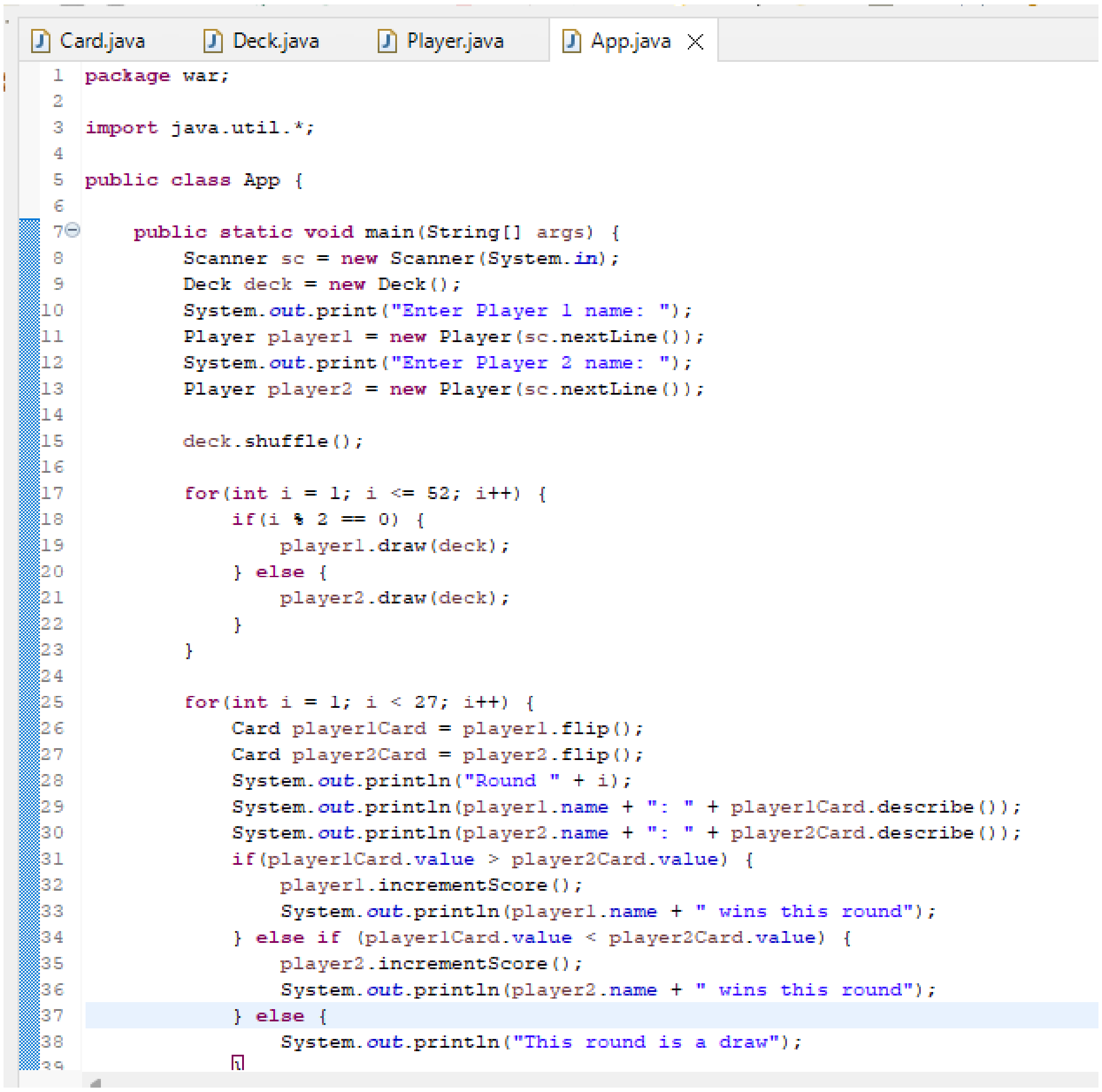
```java
package war;

public class Card {
    int value;
    String name;

    public Card(int suit, int rank) {
        value = rank;
        StringBuilder name = new StringBuilder();
        if(value == 14) {
            name.append("Ace");
        } else if (value == 13) {
            name.append("King");
        } else if(value == 12) {
            name.append("Queen");
        } else if(value == 11) {
            name.append("Jack");
        } else {
            name.append(value);
        }
        if (suit == 0) {
            name.append(" of Hearts");
        } else if (suit == 1) {
            name.append(" of Spades");
        } else if (suit == 2) {
            name.append(" of Diamonds");
        } else if (suit == 3) {
            name.append(" of Clubs");
        }
        this.name = name.toString();
    }

    public String describe() {
        return name;
```



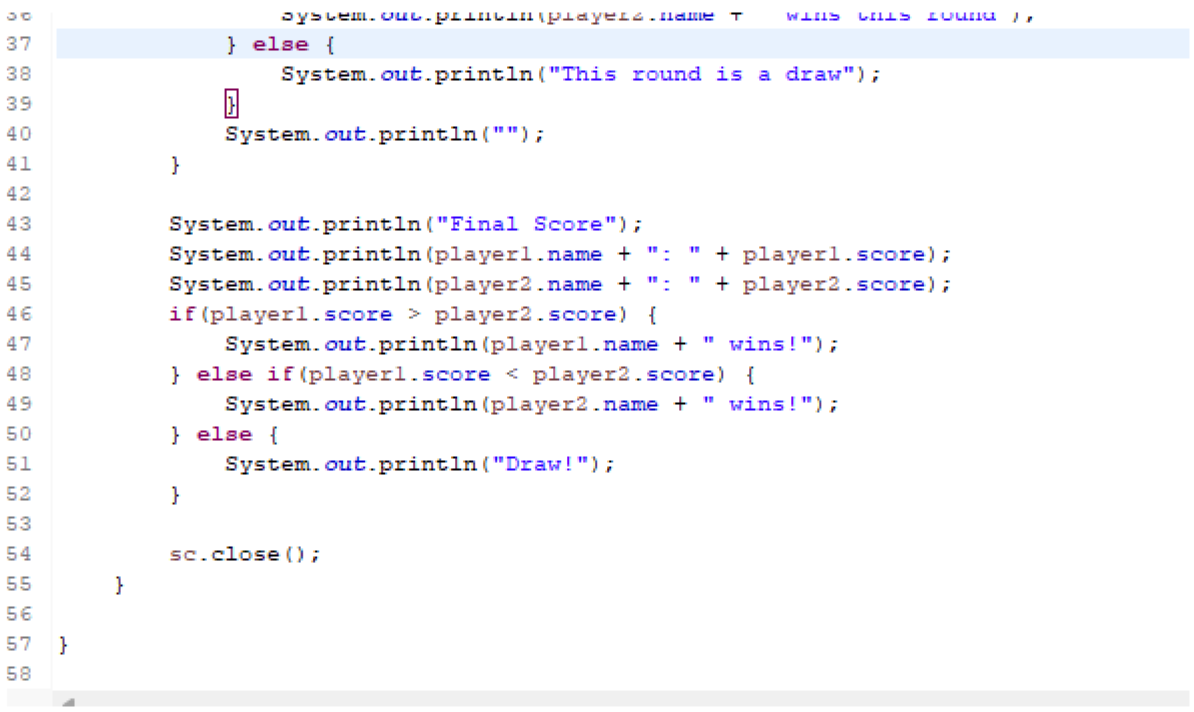
```java
package war;

import java.util.*;

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

    public Deck() {
        for(int suit = 0; suit < 4; suit++) {
            for(int rank = 2; rank < 15; rank++) {
                cards.add(new Card(suit, rank));
            }
        }
    }

    public void shuffle() {
        Collections.shuffle(cards);
    }

    public Card draw() {
        return cards.remove(0);
    }
}

```



```java
package war;

import java.util.*;

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

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

    public void describe() {
        System.out.println("Player name: " + name);
        System.out.println("Player score: " + score);
        System.out.println("Cards Remaining: " + hand.size());
        for(Card card : hand) {
            System.out.println(card.describe());
        }
    }

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

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

    public void incrementScore() {
        score += 1;
    }

}
```

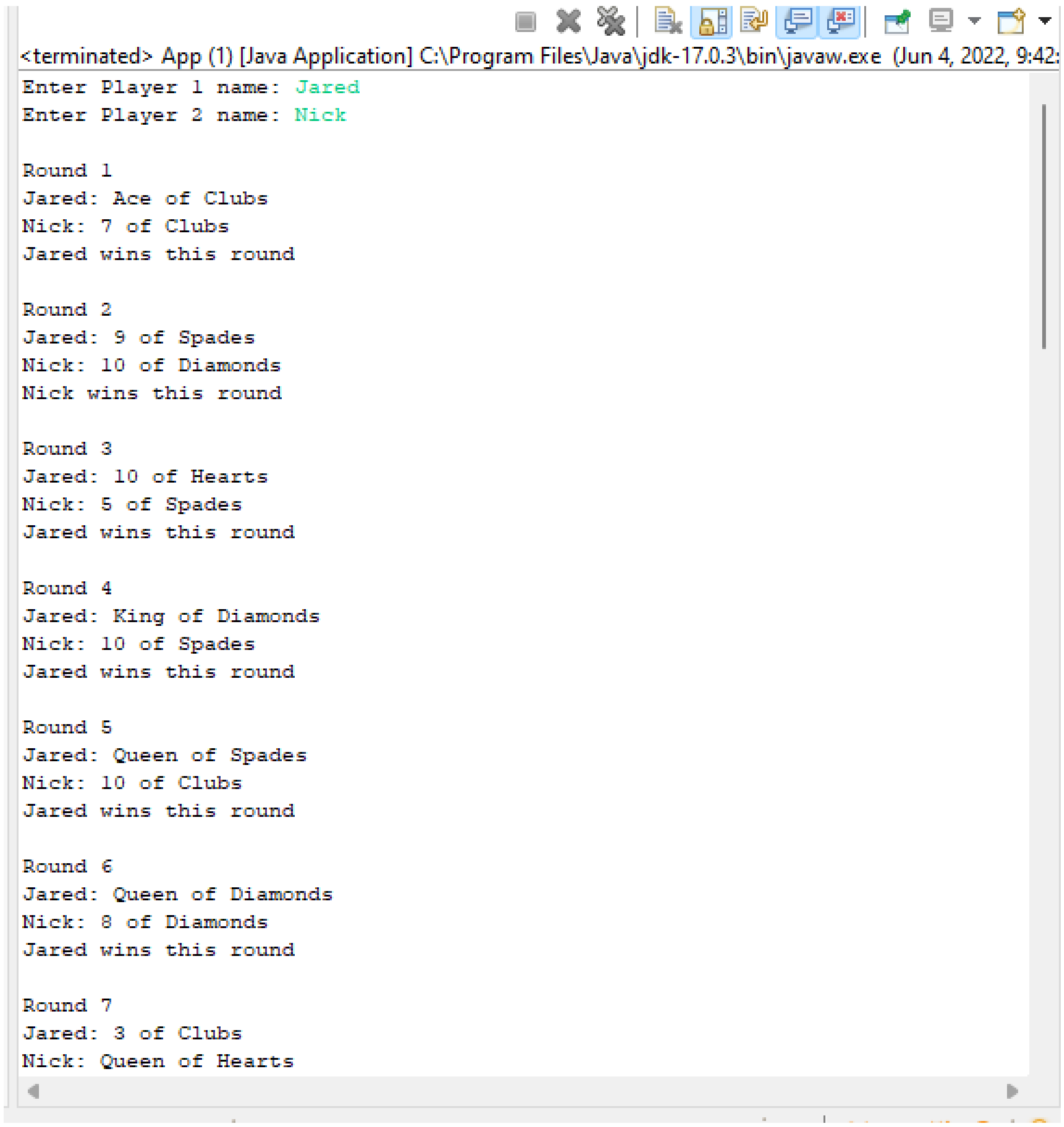


```java
package war;

import java.util.*;

public class App {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        Deck deck = new Deck();
        System.out.print("Enter Player 1 name: ");
        Player player1 = new Player(sc.nextLine());
        System.out.print("Enter Player 2 name: ");
        Player player2 = new Player(sc.nextLine());

        deck.shuffle();

        for(int i = 1; i <= 52; i++) {
            if(i % 2 == 0) {
                player1.draw(deck);
            } else {
                player2.draw(deck);
            }
        }

        for(int i = 1; i < 27; i++) {
            Card player1Card = player1.flip();
            Card player2Card = player2.flip();
            System.out.println("Round " + i);
            System.out.println(player1.name + ": " + player1Card.describe());
            System.out.println(player2.name + ": " + player2Card.describe());
            if(player1Card.value > player2Card.value) {
                player1.incrementScore();
                System.out.println(player1.name + " wins this round");
            } else if (player1Card.value < player2Card.value) {
                player2.incrementScore();
                System.out.println(player2.name + " wins this round");
            } else {
                System.out.println("This round is a draw");
```

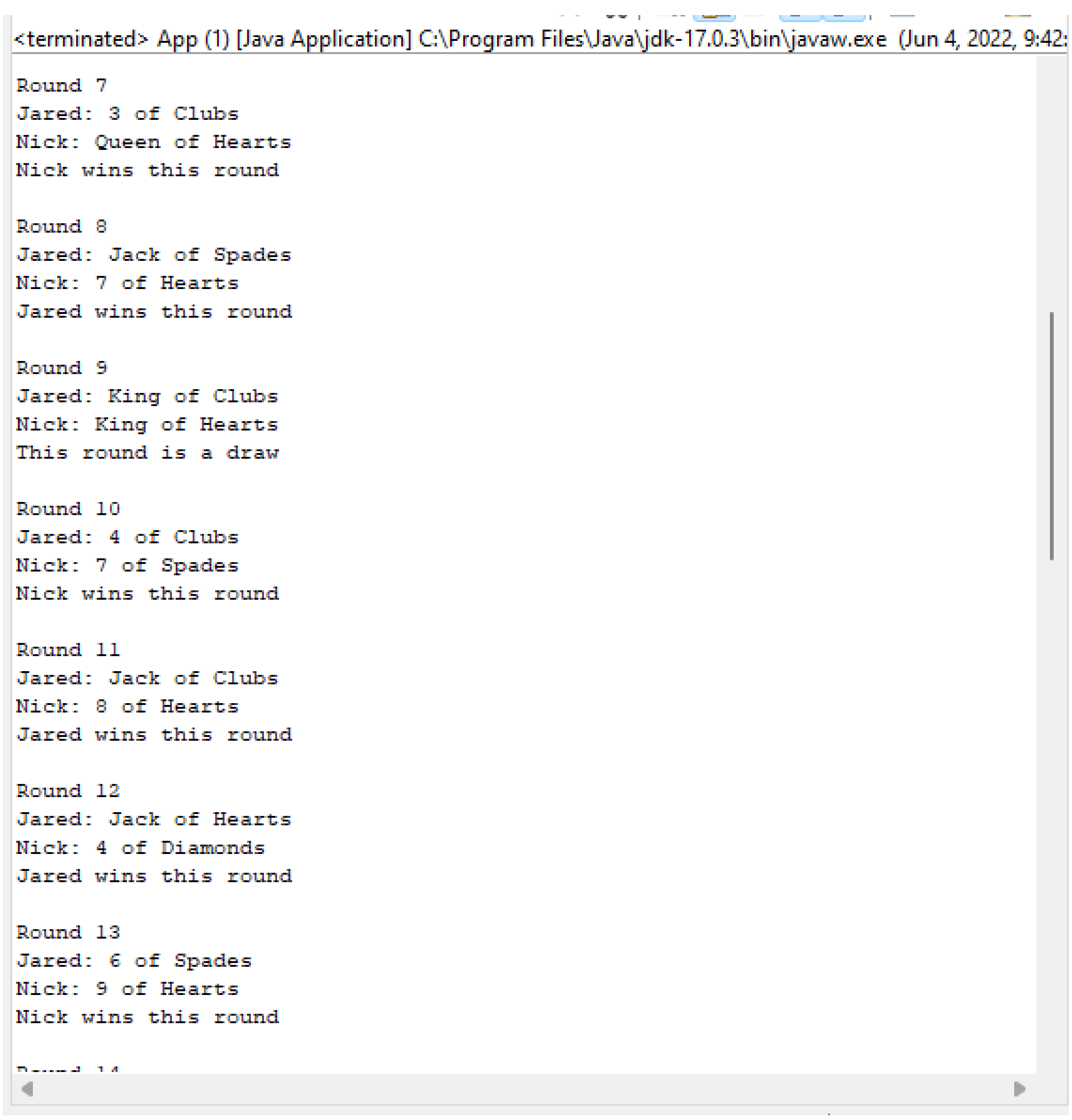


```
                System.out.println(player2.name + " wins this round");
            } else {
                System.out.println("This round is a draw");
            }
            System.out.println("");
        }

        System.out.println("Final Score");
        System.out.println(player1.name + ": " + player1.score);
        System.out.println(player2.name + ": " + player2.score);
        if(player1.score > player2.score) {
            System.out.println(player1.name + " wins!");
        } else if(player1.score < player2.score) {
            System.out.println(player2.name + " wins!");
        } else {
            System.out.println("Draw!");
        }

        sc.close();
    }

}
```

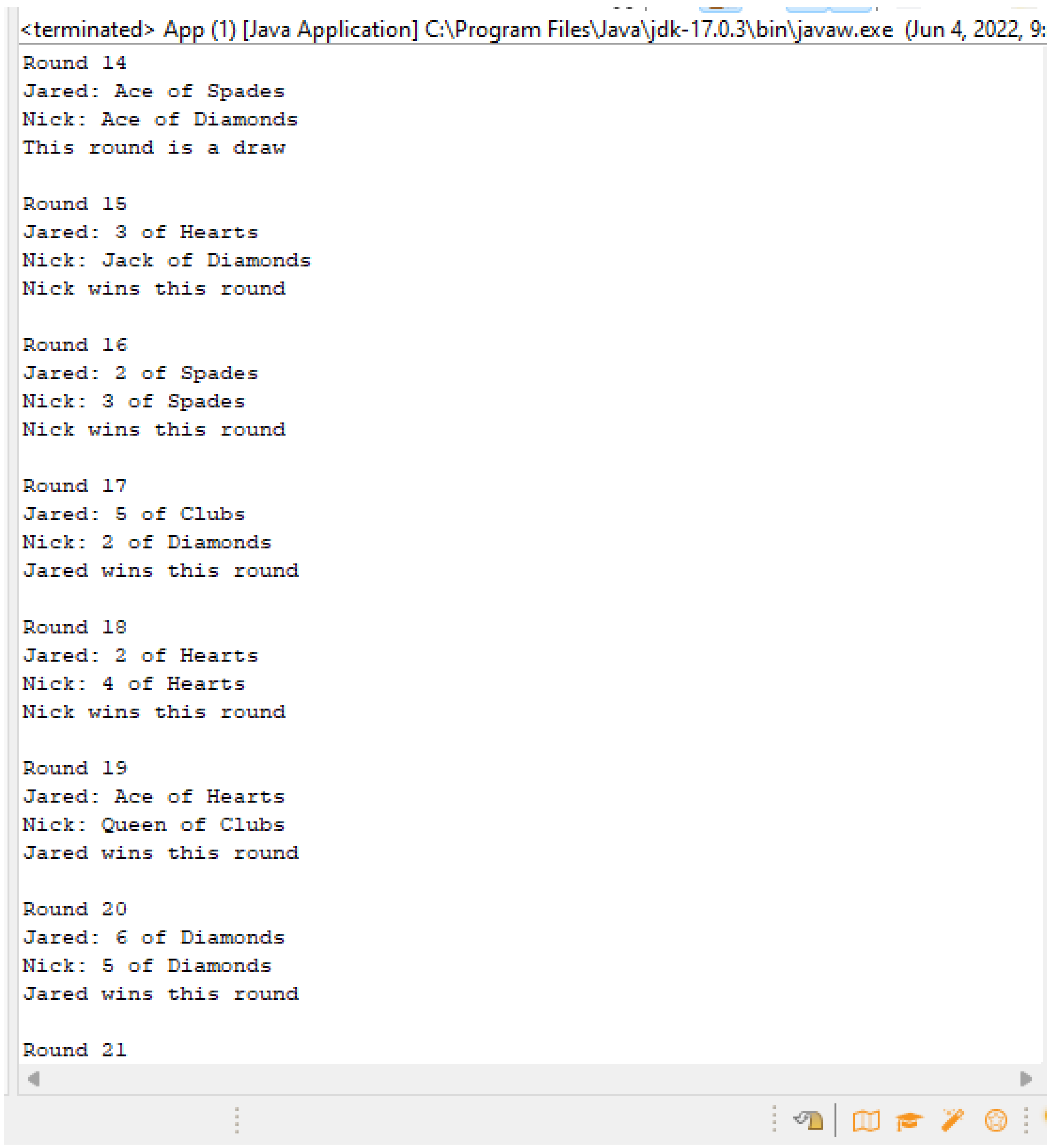
Screenshots of Running Application:



```
<terminated> App (1) [Java Application] C:\Program Files\Java\jdk-17.0.3\bin\javaw.exe (Jun 4, 2022, 9:42:
Enter Player 1 name: Jared
Enter Player 2 name: Nick

Round 1
Jared: Ace of Clubs
Nick: 7 of Clubs
Jared wins this round

Round 2
Jared: 9 of Spades
Nick: 10 of Diamonds
Nick wins this round

Round 3
Jared: 10 of Hearts
Nick: 5 of Spades
Jared wins this round

Round 4
Jared: King of Diamonds
Nick: 10 of Spades
Jared wins this round

Round 5
Jared: Queen of Spades
Nick: 10 of Clubs
Jared wins this round

Round 6
Jared: Queen of Diamonds
Nick: 8 of Diamonds
Jared wins this round

Round 7
Jared: 3 of Clubs
Nick: Queen of Hearts
```

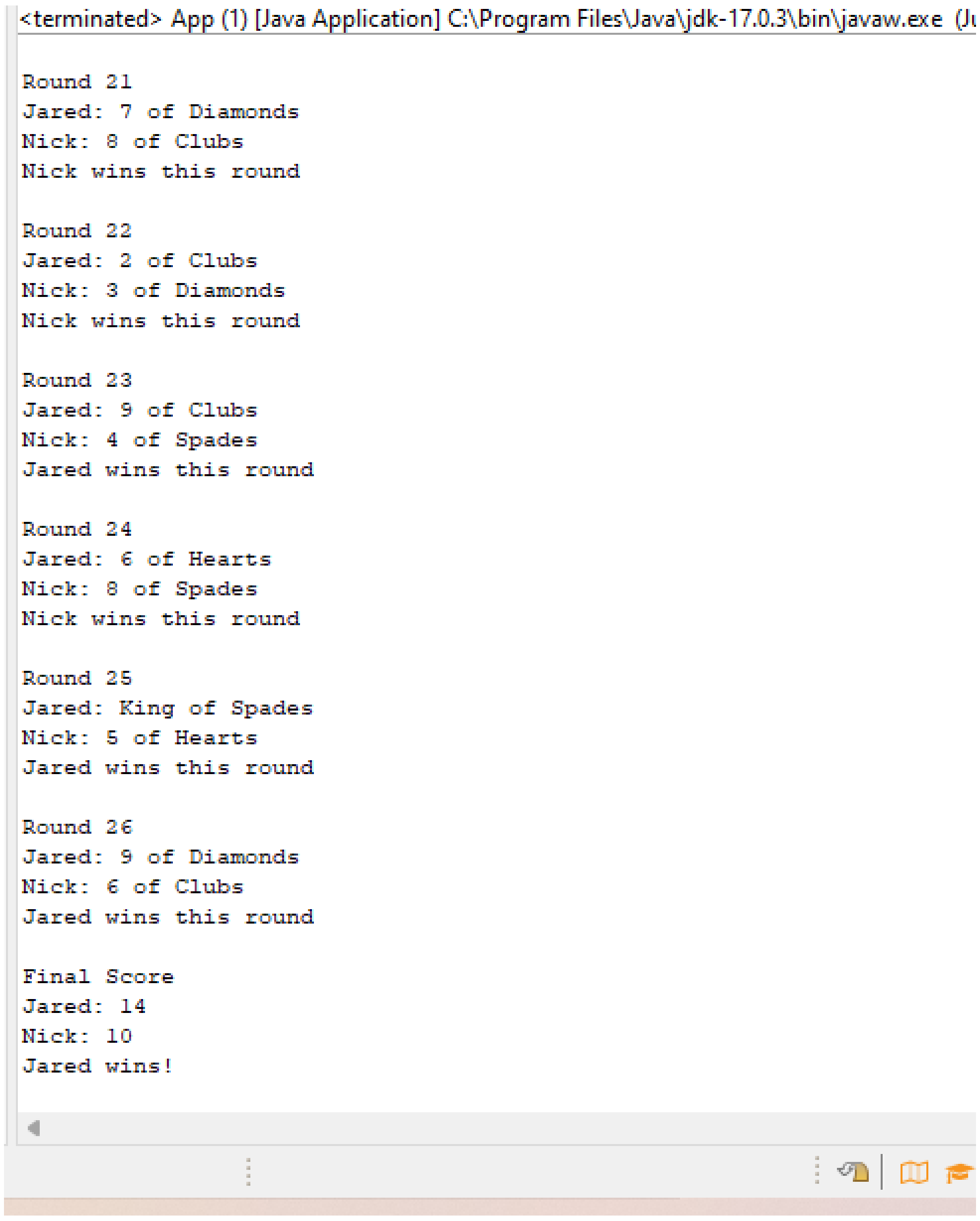

<terminated> App (1) [Java Application] C:\Program Files\Java\jdk-17.0.3\bin\javaw.exe (Jun 4, 2022, 9:42:

```
Round 7
Jared: 3 of Clubs
Nick: Queen of Hearts
Nick wins this round

Round 8
Jared: Jack of Spades
Nick: 7 of Hearts
Jared wins this round

Round 9
Jared: King of Clubs
Nick: King of Hearts
This round is a draw

Round 10
Jared: 4 of Clubs
Nick: 7 of Spades
Nick wins this round

Round 11
Jared: Jack of Clubs
Nick: 8 of Hearts
Jared wins this round

Round 12
Jared: Jack of Hearts
Nick: 4 of Diamonds
Jared wins this round

Round 13
Jared: 6 of Spades
Nick: 9 of Hearts
Nick wins this round
```



<terminated> App (1) [Java Application] C:\Program Files\Java\jdk-17.0.3\bin\javaw.exe (Jun 4, 2022, 9:

```
Round 14
Jared: Ace of Spades
Nick: Ace of Diamonds
This round is a draw

Round 15
Jared: 3 of Hearts
Nick: Jack of Diamonds
Nick wins this round

Round 16
Jared: 2 of Spades
Nick: 3 of Spades
Nick wins this round

Round 17
Jared: 5 of Clubs
Nick: 2 of Diamonds
Jared wins this round

Round 18
Jared: 2 of Hearts
Nick: 4 of Hearts
Nick wins this round

Round 19
Jared: Ace of Hearts
Nick: Queen of Clubs
Jared wins this round

Round 20
Jared: 6 of Diamonds
Nick: 5 of Diamonds
Jared wins this round

Round 21
```



```
<terminated> App (1) [Java Application] C:\Program Files\Java\jdk-17.0.3\bin\javaw.exe (J

Round 21
Jared: 7 of Diamonds
Nick: 8 of Clubs
Nick wins this round

Round 22
Jared: 2 of Clubs
Nick: 3 of Diamonds
Nick wins this round

Round 23
Jared: 9 of Clubs
Nick: 4 of Spades
Jared wins this round

Round 24
Jared: 6 of Hearts
Nick: 8 of Spades
Nick wins this round

Round 25
Jared: King of Spades
Nick: 5 of Hearts
Jared wins this round

Round 26
Jared: 9 of Diamonds
Nick: 6 of Clubs
Jared wins this round

Final Score
Jared: 14
Nick: 10
Jared wins!
```

URL to GitHub Repository:

https://github.com/JaredBears/War