

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:

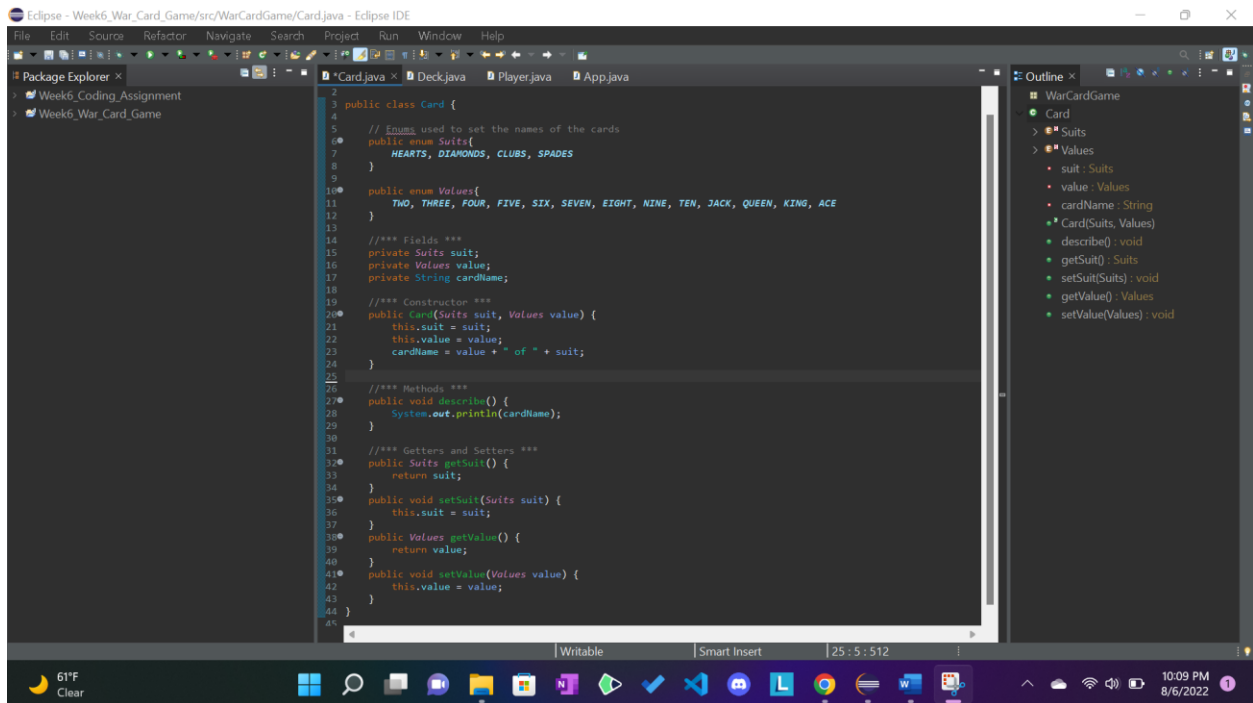
For the final project you will be creating an automated version of the classic card game *WAR*.

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 Class:

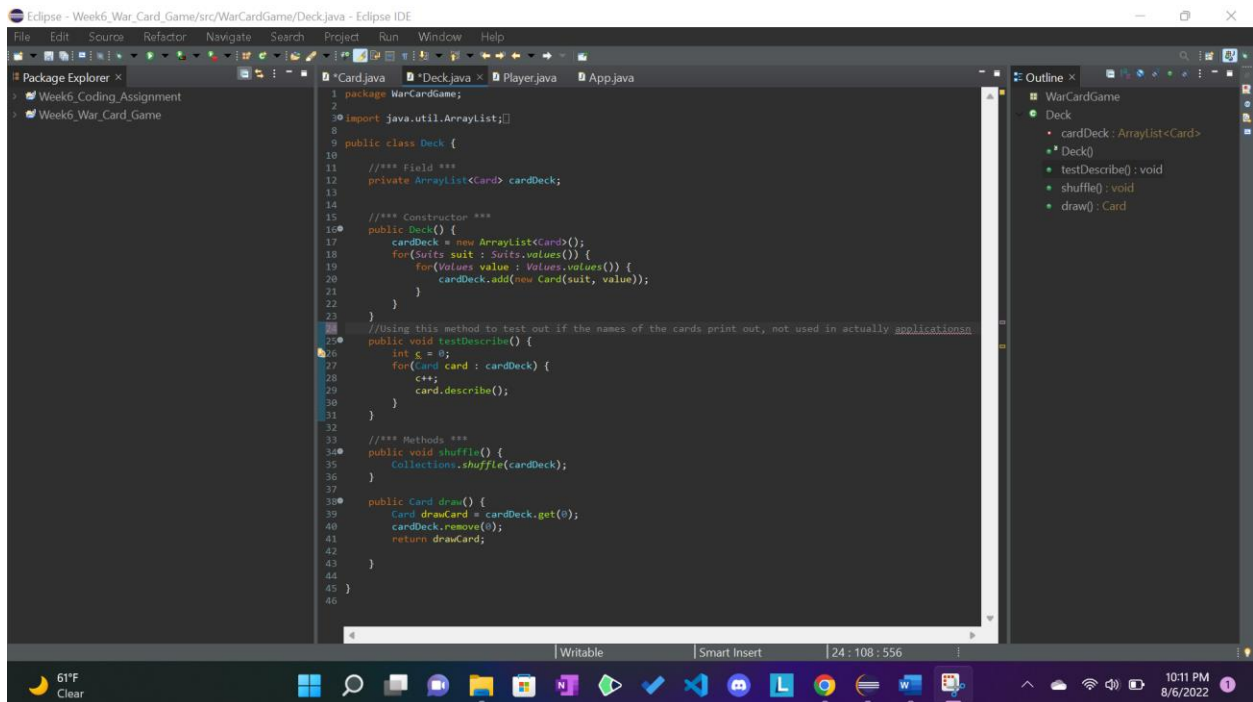


```
1 package WarCardGame;
2
3 public class Card {
4
5     // Enums used to set the names of the cards
6     public enum Suits {
7         HEARTS, DIAMONDS, CLUBS, SPADES
8     }
9
10    public enum Values {
11        TWO, THREE, FOUR, FIVE, SIX, SEVEN, EIGHT, NINE, TEN, JACK, QUEEN, KING, ACE
12    }
13
14    /** Fields */
15    private Suits suit;
16    private Values value;
17    private String cardName;
18
19    /** Constructor */
20    public Card(Suits suit, Values value) {
21        this.suit = suit;
22        this.value = value;
23        cardName = value + " of " + suit;
24    }
25
26    /** Methods */
27    public void describe() {
28        System.out.println(cardName);
29    }
30
31    /** Getters and Setters */
32    public Suits getSuit() {
33        return suit;
34    }
35    public void setSuit(Suits suit) {
36        this.suit = suit;
37    }
38    public Values getValue() {
39        return value;
40    }
41    public void setValue(Values value) {
42        this.value = value;
43    }
44 }
45
```

Outline:

- WarCardGame
 - Card
 - Suits
 - Values
 - suit : Suits
 - value : Values
 - cardName : String
 - Card(Suits, Values)
 - describe() : void
 - getSuit() : Suits
 - setSuit(Suits) : void
 - getValue() : Values
 - setValue(Values) : void

Deck Class:

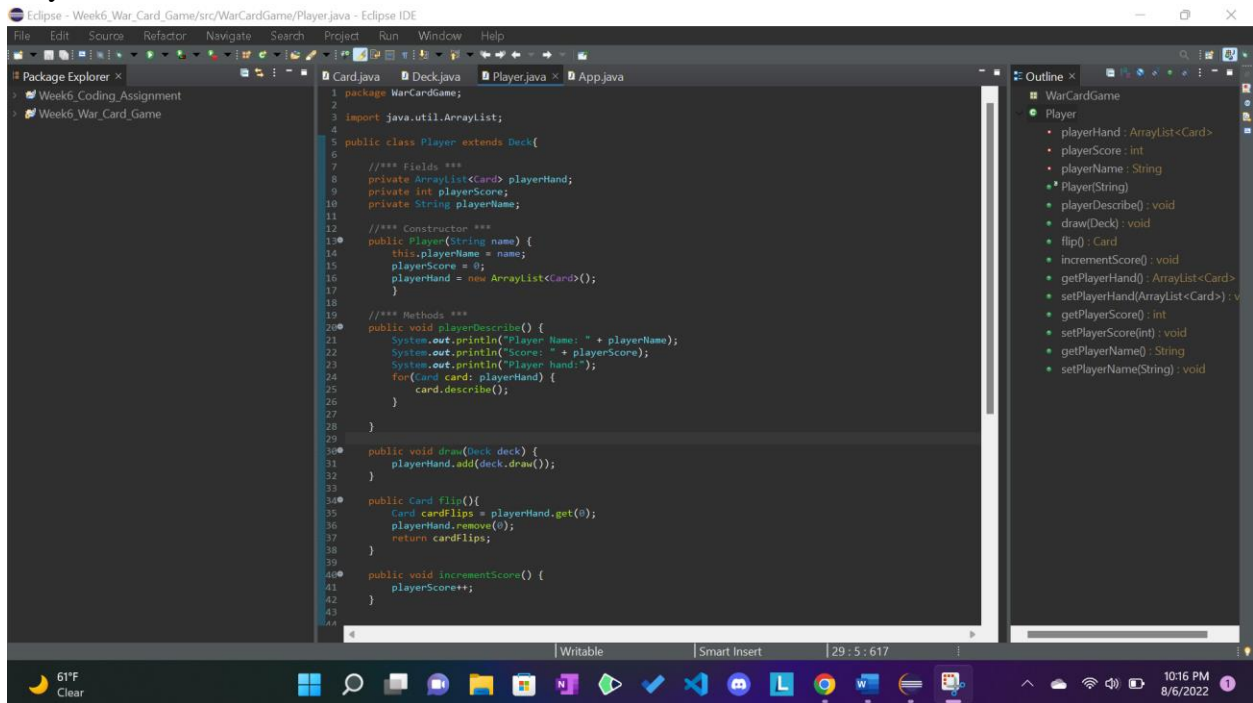


```
1 package WarCardGame;
2
3 import java.util.ArrayList;
4
5 public class Deck {
6
7     /** Field */
8     private ArrayList<Card> cardDeck;
9
10    /** Constructor */
11    public Deck() {
12        cardDeck = new ArrayList<Card>();
13        for (Suits suit : Suits.values()) {
14            for (Values value : Values.values()) {
15                cardDeck.add(new Card(suit, value));
16            }
17        }
18    }
19
20    /** Using this method to test out if the names of the cards print out, not used in actually applications */
21    public void testDescribe() {
22        int i = 0;
23        for (Card card : cardDeck) {
24            i++;
25            card.describe();
26        }
27    }
28
29    /** Methods */
30    public void shuffle() {
31        Collections.shuffle(cardDeck);
32    }
33
34    public Card draw() {
35        Card drawCard = cardDeck.get(0);
36        cardDeck.remove(0);
37        return drawCard;
38    }
39 }
40
```

Outline:

- WarCardGame
 - Deck
 - cardDeck : ArrayList<Card>
 - Deck()
 - testDescribe() : void
 - shuffle() : void
 - draw() : Card

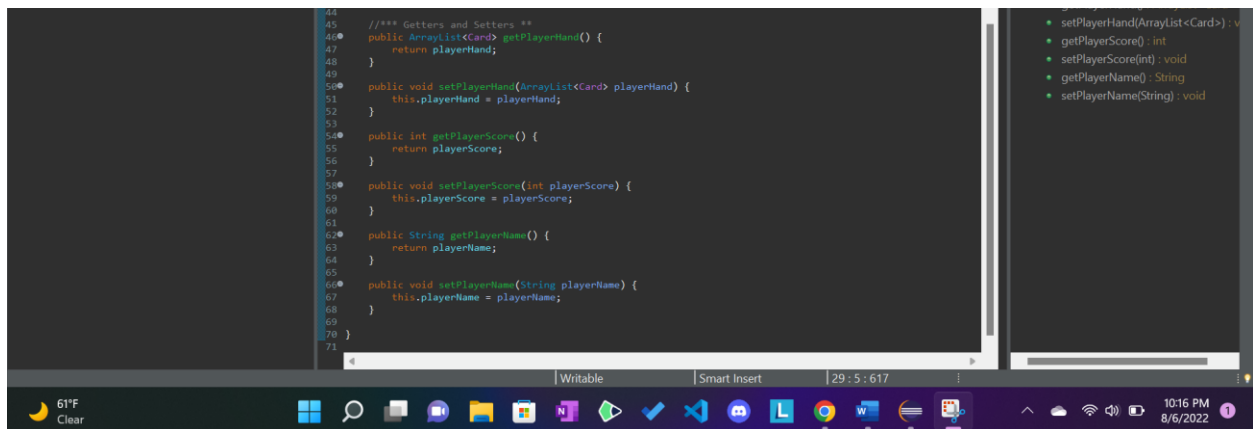
Player Class:



```
1 package WarCardGame;
2
3 import java.util.ArrayList;
4
5 public class Player extends Deck{
6
7     /** Fields **
8     private ArrayList<Card> playerHand;
9     private int playerScore;
10    private String playerName;
11
12    /** Constructor **
13    public Player(String name) {
14        this.playerName = name;
15        playerScore = 0;
16        playerHand = new ArrayList<Card>();
17    }
18
19    /** Methods **
20    public void playerDescribe() {
21        System.out.println("Player Name: " + playerName);
22        System.out.println("Score: " + playerScore);
23        System.out.println("Player hand:");
24        for(Card card: playerHand) {
25            card.describe();
26        }
27    }
28
29
30    public void draw(Deck deck) {
31        playerHand.add(deck.draw());
32    }
33
34    public Card flip(){
35        Card cardFlips = playerHand.get(0);
36        playerHand.remove(0);
37        return cardFlips;
38    }
39
40    public void incrementScore() {
41        playerScore++;
42    }
43
44 }
```

Outline:

- WarCardGame
 - Player
 - playerHand : ArrayList<Card>
 - playerScore : int
 - playerName : String
 - Player(String)
 - playerDescribe() : void
 - draw(Deck) : void
 - flip() : Card
 - incrementScore() : void
 - getPlayerHand() : ArrayList<Card>
 - setPlayerHand(ArrayList<Card>) : void
 - getPlayerScore() : int
 - setPlayerScore(int) : void
 - getPlayerName() : String
 - setPlayerName(String) : void

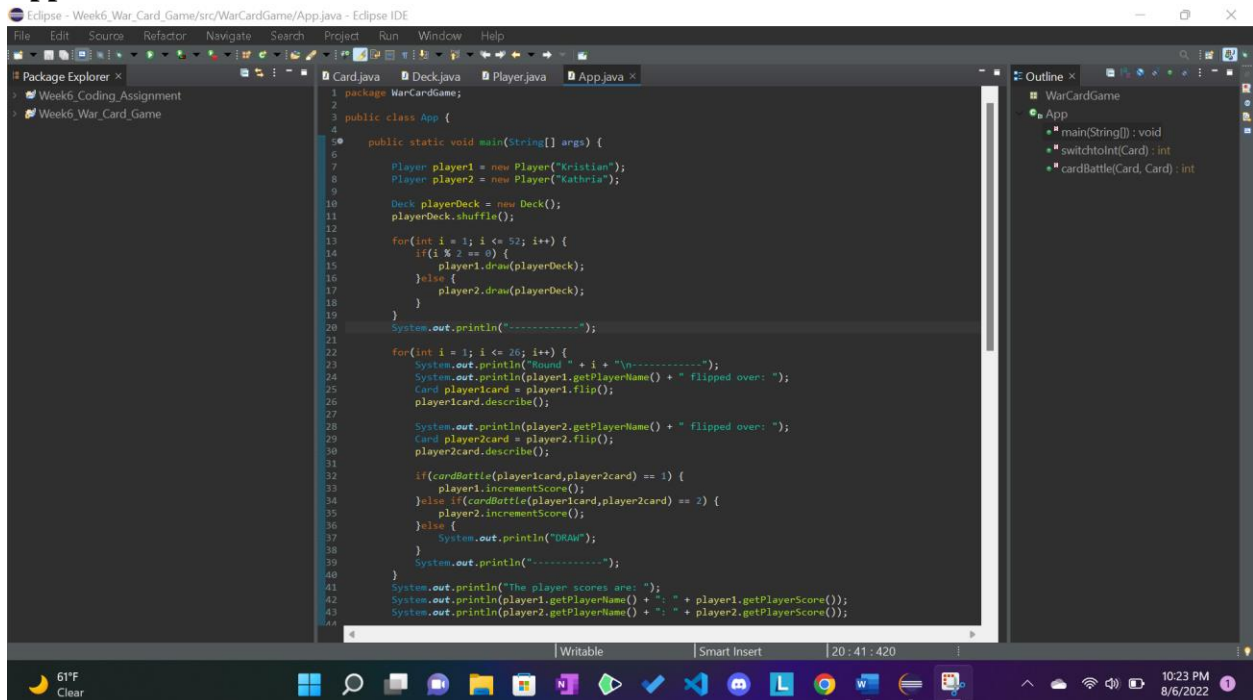


```
44
45 /** Getters and Setters **
46 public ArrayList<Card> getPlayerHand() {
47     return playerHand;
48 }
49
50 public void setPlayerHand(ArrayList<Card> playerHand) {
51     this.playerHand = playerHand;
52 }
53
54 public int getPlayerScore() {
55     return playerScore;
56 }
57
58 public void setPlayerScore(int playerScore) {
59     this.playerScore = playerScore;
60 }
61
62 public String getPlayerName() {
63     return playerName;
64 }
65
66 public void setPlayerName(String playerName) {
67     this.playerName = playerName;
68 }
69
70 }
71 }
```

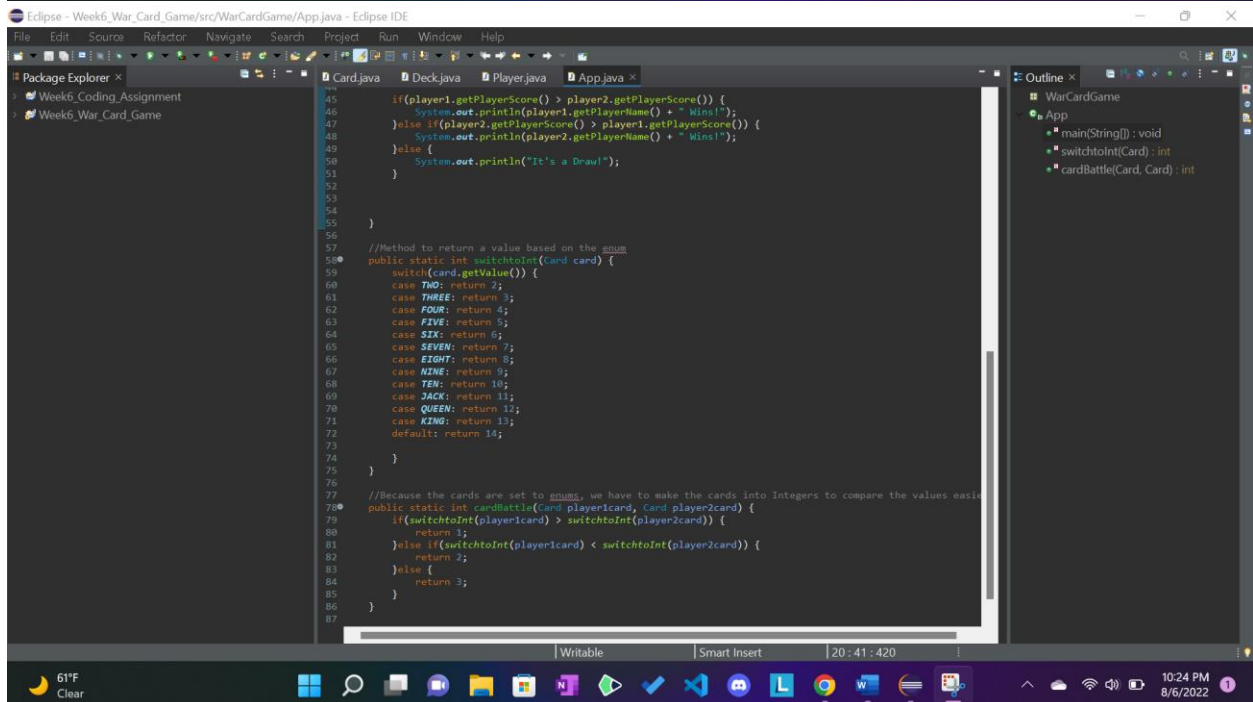
Outline:

- WarCardGame
 - Player
 - playerHand : ArrayList<Card>
 - playerScore : int
 - playerName : String
 - Player(String)
 - playerDescribe() : void
 - draw(Deck) : void
 - flip() : Card
 - incrementScore() : void
 - getPlayerHand() : ArrayList<Card>
 - setPlayerHand(ArrayList<Card>) : void
 - getPlayerScore() : int
 - setPlayerScore(int) : void
 - getPlayerName() : String
 - setPlayerName(String) : void

App Class:

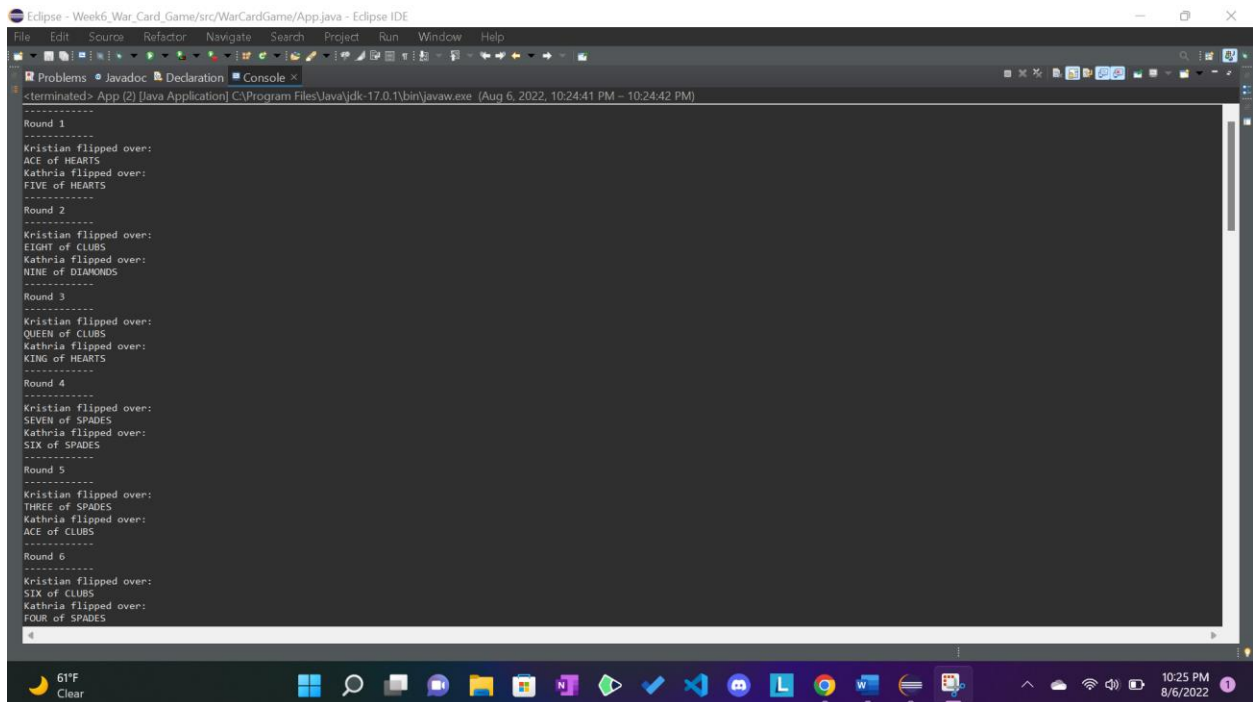


```
1 package WarCardGame;
2
3 public class App {
4
5     public static void main(String[] args) {
6
7         Player player1 = new Player("Kristian");
8         Player player2 = new Player("Kathria");
9
10        Deck playerDeck = new Deck();
11        playerDeck.shuffle();
12
13        for(int i = 1; i <= 52; i++) {
14            if(i % 2 == 0) {
15                player1.draw(playerDeck);
16            } else {
17                player2.draw(playerDeck);
18            }
19        }
20        System.out.println("-----");
21
22        for(int i = 1; i <= 26; i++) {
23            System.out.println("Round " + i + " =====");
24            System.out.println(player1.getPlayerName() + " flipped over: ");
25            Card player1card = player1.flip();
26            player1card.describe();
27
28            System.out.println(player2.getPlayerName() + " flipped over: ");
29            Card player2card = player2.flip();
30            player2card.describe();
31
32            if(cardBattle(player1card, player2card) == 1) {
33                player1.incrementScore();
34            } else if(cardBattle(player1card, player2card) == 2) {
35                player2.incrementScore();
36            } else {
37                System.out.println("DRAW");
38            }
39            System.out.println("-----");
40
41            System.out.println("The player scores are: ");
42            System.out.println(player1.getPlayerName() + ": " + player1.getPlayerScore());
43            System.out.println(player2.getPlayerName() + ": " + player2.getPlayerScore());
44        }
45    }
46}
```



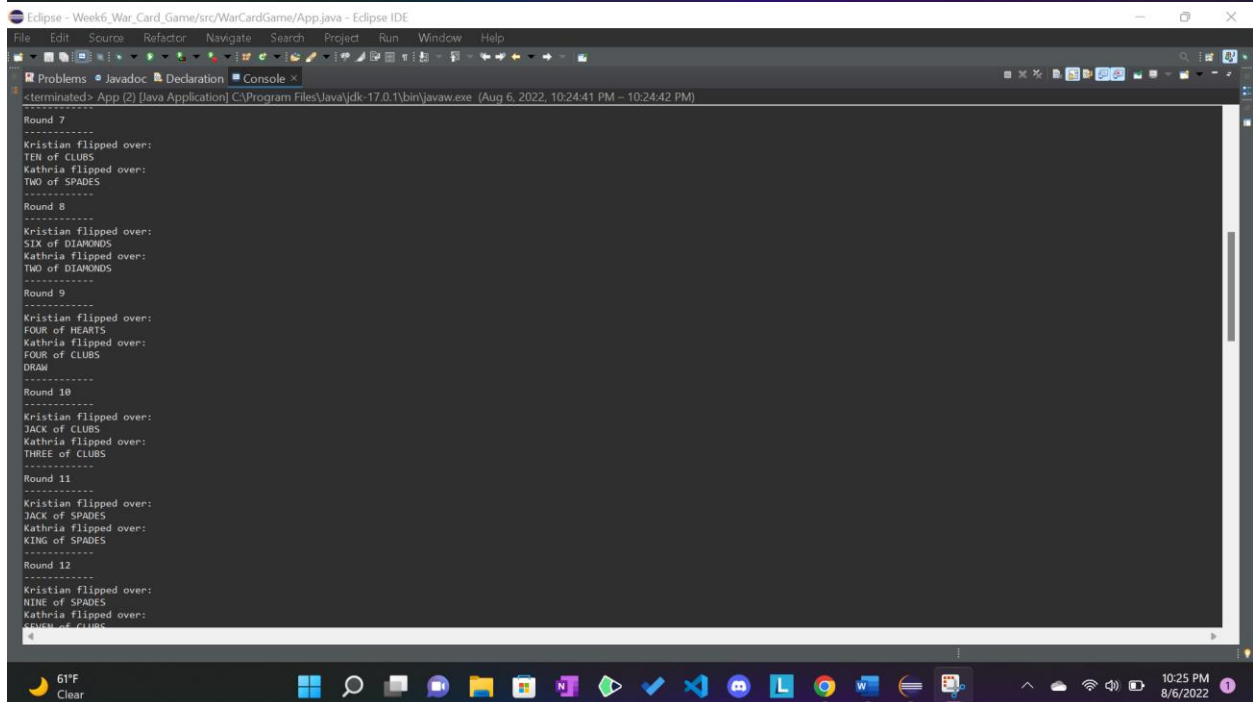
```
45
46    if(player1.getPlayerScore() > player2.getPlayerScore()) {
47        System.out.println(player1.getPlayerName() + " Wins!");
48    } else if(player2.getPlayerScore() > player1.getPlayerScore()) {
49        System.out.println(player2.getPlayerName() + " Wins!");
50    } else {
51        System.out.println("It's a Draw!");
52    }
53
54    }
55
56    //Method to return a value based on the enum
57    public static int switchtoInt(Card card) {
58        switch(card.getValue()) {
59            case TWO: return 2;
60            case THREE: return 3;
61            case FOUR: return 4;
62            case FIVE: return 5;
63            case SIX: return 6;
64            case SEVEN: return 7;
65            case EIGHT: return 8;
66            case NINE: return 9;
67            case TEN: return 10;
68            case JACK: return 11;
69            case QUEEN: return 12;
70            case KING: return 13;
71            default: return 14;
72        }
73    }
74
75    //Because the cards are set to enums, we have to make the cards into Integers to compare the values easily
76    public static int cardBattle(Card player1card, Card player2card) {
77        if(switchtoInt(player1card) > switchtoInt(player2card)) {
78            return 1;
79        } else if(switchtoInt(player1card) < switchtoInt(player2card)) {
80            return 2;
81        } else {
82            return 3;
83        }
84    }
85
86    }
87}
```

Screenshots of Running Application:



The screenshot shows the Eclipse IDE interface with the console window open. The console displays the output of a Java application running a card game. The output shows rounds 1 through 6, with each round having two lines of output: "Kristian flipped over:" followed by a card name, and "Kathria flipped over:" followed by a card name. The cards are: Round 1: ACE of HEARTS, FIVE of HEARTS; Round 2: EIGHT of CLUBS, NINE of DIAMONDS; Round 3: QUEEN of CLUBS, KING of HEARTS; Round 4: SEVEN of SPADES, SIX of SPADES; Round 5: THREE of SPADES, ACE of CLUBS; Round 6: SIX of CLUBS, FOUR of SPADES. The console window title is "Console x" and the application title is "App (2) [Java Application] C:\Program Files\Java\jdk-17.0.1\bin\javaw.exe (Aug 6, 2022, 10:24:41 PM - 10:24:42 PM)". The taskbar at the bottom shows the system clock as 10:25 PM 8/6/2022 and the temperature as 61°F Clear.

```
<terminated> App (2) [Java Application] C:\Program Files\Java\jdk-17.0.1\bin\javaw.exe (Aug 6, 2022, 10:24:41 PM - 10:24:42 PM)
-----
Round 1
-----
Kristian flipped over:
ACE of HEARTS
Kathria flipped over:
FIVE of HEARTS
-----
Round 2
-----
Kristian flipped over:
EIGHT of CLUBS
Kathria flipped over:
NINE of DIAMONDS
-----
Round 3
-----
Kristian flipped over:
QUEEN of CLUBS
Kathria flipped over:
KING of HEARTS
-----
Round 4
-----
Kristian flipped over:
SEVEN of SPADES
Kathria flipped over:
SIX of SPADES
-----
Round 5
-----
Kristian flipped over:
THREE of SPADES
Kathria flipped over:
ACE of CLUBS
-----
Round 6
-----
Kristian flipped over:
SIX of CLUBS
Kathria flipped over:
FOUR of SPADES
-----
```



The screenshot shows the Eclipse IDE interface with the console window open. The console displays the output of a Java application running a card game. The output shows rounds 7 through 12, with each round having two lines of output: "Kristian flipped over:" followed by a card name, and "Kathria flipped over:" followed by a card name. The cards are: Round 7: TEN of CLUBS, TWO of SPADES; Round 8: SIX of DIAMONDS, TWO of DIAMONDS; Round 9: FOUR of HEARTS, FOUR of CLUBS; Round 10: JACK of CLUBS, THREE of CLUBS; Round 11: JACK of SPADES, KING of SPADES; Round 12: NINE of SPADES, SEVEN of CLUBS. The console window title is "Console x" and the application title is "App (2) [Java Application] C:\Program Files\Java\jdk-17.0.1\bin\javaw.exe (Aug 6, 2022, 10:24:41 PM - 10:24:42 PM)". The taskbar at the bottom shows the system clock as 10:25 PM 8/6/2022 and the temperature as 61°F Clear.

```
<terminated> App (2) [Java Application] C:\Program Files\Java\jdk-17.0.1\bin\javaw.exe (Aug 6, 2022, 10:24:41 PM - 10:24:42 PM)
-----
Round 7
-----
Kristian flipped over:
TEN of CLUBS
Kathria flipped over:
TWO of SPADES
-----
Round 8
-----
Kristian flipped over:
SIX of DIAMONDS
Kathria flipped over:
TWO of DIAMONDS
-----
Round 9
-----
Kristian flipped over:
FOUR of HEARTS
Kathria flipped over:
FOUR of CLUBS
DRAW
-----
Round 10
-----
Kristian flipped over:
JACK of CLUBS
Kathria flipped over:
THREE of CLUBS
-----
Round 11
-----
Kristian flipped over:
JACK of SPADES
Kathria flipped over:
KING of SPADES
-----
Round 12
-----
Kristian flipped over:
NINE of SPADES
Kathria flipped over:
SEVEN of CLUBS
-----
```

Eclipse - Week6_War_Card_Game/src/WarCardGame/App.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Problems Javadoc Declaration Console x

<terminated> App (2) [Java Application] C:\Program Files\Java\jdk-17.0.1\bin\javaw.exe (Aug 6, 2022, 10:24:41 PM - 10:24:42 PM)

```
-----
Round 13
-----
Kristian flipped over:
THREE of DIAMONDS
Kathria flipped over:
TWO of HEARTS
-----
Round 14
-----
Kristian flipped over:
SEVEN of HEARTS
Kathria flipped over:
KING of DIAMONDS
-----
Round 15
-----
Kristian flipped over:
SEVEN of DIAMONDS
Kathria flipped over:
KING of CLUBS
-----
Round 16
-----
Kristian flipped over:
QUEEN of SPADES
Kathria flipped over:
THREE of HEARTS
-----
Round 17
-----
Kristian flipped over:
FIVE of CLUBS
Kathria flipped over:
JACK of HEARTS
-----
Round 18
-----
Kristian flipped over:
EIGHT of SPADES
Kathria flipped over:
NINE of CLUBS
-----
```

61°F Clear 10:25 PM 8/6/2022

Eclipse - Week6_War_Card_Game/src/WarCardGame/App.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Problems Javadoc Declaration Console x

<terminated> App (2) [Java Application] C:\Program Files\Java\jdk-17.0.1\bin\javaw.exe (Aug 6, 2022, 10:24:41 PM - 10:24:42 PM)

```
-----
Round 19
-----
Kristian flipped over:
FIVE of DIAMONDS
Kathria flipped over:
TEN of HEARTS
-----
Round 20
-----
Kristian flipped over:
EIGHT of DIAMONDS
Kathria flipped over:
SIX of HEARTS
-----
Round 21
-----
Kristian flipped over:
QUEEN of DIAMONDS
Kathria flipped over:
ACE of SPADES
-----
Round 22
-----
Kristian flipped over:
EIGHT of HEARTS
Kathria flipped over:
JACK of DIAMONDS
-----
Round 23
-----
Kristian flipped over:
NINE of HEARTS
Kathria flipped over:
TEN of DIAMONDS
-----
Round 24
-----
Kristian flipped over:
TWO of CLUBS
Kathria flipped over:
QUEEN of HEARTS
-----
```

61°F Clear 10:26 PM 8/6/2022

```
-----
Round 25
-----
Kristian flipped over:
FOUR of DIAMONDS
Kathria flipped over:
FIVE of SPADES
-----
Round 26
-----
Kristian flipped over:
ACE of DIAMONDS
Kathria flipped over:
TEN of SPADES
-----
The player scores are:
Kristian: 11
Kathria: 14
Kathria Wins!
```

URL to GitHub Repository:

https://github.com/KristianPador/War_Card_Game