# Intro to Java Week 6 Coding Assignment

Tyler Livermore 4/2/2022

**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:**

```
1 package week6FinalProject;
    import java.util.Scanner;
    public class App {
         public static void main(String[] args) {
  7⊝
            Deck deck = new Deck();
  8
  9
            Player player1 = new Player();
 10
            Player player2 = new Player();
 11
 12
             //Receive and set user input for player names
 13
             Scanner enterName = new Scanner(System.in);
            System.out.print("Please Enter Player 1 Name: ");
 14
 15
            String playerName = enterName.next();
 16
            player1.setPlayer(playerName);
 17
 18
            System.out.print("Please Enter Player 2 Name: ");
 19
            playerName = enterName.next();
            player2.setPlayer(playerName);
 20
 21
            enterName.close();
 22
            //scanner closed to avoid leak. Must be after all inputs completed or else exception on the 2nd input b/c the System.in is closed
 23
 24
            deck.shuffle();
 25
 26
             //divvy out the players' hands
 27
             for (int i = 0; i < 52; i++) {
                if (i % 2 == 0) { //using modulus to alternate draws as even/odd turns
 28
 29
                    player1.draw(deck);
 30
                } else if (i % 2 != 0) {
                    player2.draw(deck);
 31
 32
 33
            //run each round of War, increment score for each round winner +1 for (int j = 0; j < 26; j++) {
 35
 36
 37
                 int player1Card = player1.flip();
 38
                int player2Card = player2.flip();
 39
 40
                if (player1Card > player2Card) {
 41
                    player1.incrementScore();
 42
                } else if (player2Card > player1Card) {
 43
 44
                    player2.incrementScore();
 45
            }
 46
 47
 48
            player1.getScore();
 49
             player2.getScore();
             System.out.println(player1.getPlayerName() + ": " + player1.getScore());
 50
             System.out.println(player2.getPlayerName() + ": " + player2.getScore());
 51
52
53
               //compare final scores
54
               if (player1.getScore() > player2.getScore()) {
55
                    System.out.println(player1.getPlayerName() + " wins!");
               } else if (player2.getScore() > player1.getScore()) {
56
57
                    System.out.println(player2.getPlayerName() +
                                                                             " wins!");
58
               } else {
59
                    System.out.println("The game ends in a draw.");
60
61
          }
62
     }
```

```
package week6FinalProject;
 3 public class Card {
 4
        private int value;
                               //2-14 (2-Ace)
 5
        private String name;
                            //Ace of Diamonds, etc.
 6
        public void setCard(String name, int value) {
 7⊝
 8
           this.name = name;
 9
            this.value = value;
10
11
        public int getCardValue() {
12⊖
            int cardValue = this.value;
13
14
            return cardValue;
15
16
        public String getCardName() {
17⊝
            String cardName = this.name;
18
19
            return cardName;
20
        }
21
        public void describe() {
22⊖
23
            System.out.println("(" + getCardValue() + ") " + getCardName());
24
25 }
```

```
    Deck.java 
    X

                                                                                   Card card22 = new Card();
  package week6FinalProject;
                                                                                   card22.setCard("Ten of Spades", 10);
                                                                   58
  3⊕ import java.util.ArrayList; ...
                                                                   59
                                                                                   Card card23 = new Card();
                                                                                   card23.setCard("Jack of Spades", 11);
                                                                                   Card card24 = new Card();
                                                                   61
     public class Deck {
  8
                                                                                   card24.setCard("Queen of Spades", 12);
         private Map <String, Integer> cards;
                                                                   62
                                                                                   Card card25 = new Card();
 10
         private List <String> shuffledCards;
                                                                                   card25.setCard("King of Spades", 13);
 11
                                                                                   Card card26 = new Card();
 129
             public Deck() { //create the 52 standard cards
                                                                   65
                 //Hearts
                                                                                   card26.setCard("Ace of Spades", 14);
                 Card card1 = new Card();
                                                                   67
                                                                                   //Diamonds
 14
                                                                                   Card card27 = new Card();
                 card1.setCard("Two of Hearts", 2);
 15
                                                                   68
                 Card card2 = new Card();
                                                                   69
                                                                                   card27.setCard("Two of Diamonds", 2);
                 card2.setCard("Three of Hearts", 3);
                                                                                   Card card28 = new Card();
 17
                                                                   70
                                                                                   card28.setCard("Three of Diamonds", 3);
                 Card card3 = new Card();
                                                                   71
 18
 19
                 card3.setCard("Four of Hearts", 4);
                                                                   72
                                                                                   Card card29 = new Card();
                 Card card4 = new Card();
                                                                                   card29.setCard("Four of Diamonds", 4);
 20
                                                                   73
                 card4.setCard("Five of Hearts", 5);
                                                                   74
                                                                                   Card card30 = new Card();
 21
                                                                                   card30.setCard("Five of Diamonds", 5);
 22
                 Card card5 = new Card();
                                                                   75
                                                                   76
                                                                                   Card card31 = new Card();
 23
                 card5.setCard("Six of Hearts", 6);
                 Card card6 = new Card();
                                                                                   card31.setCard("Six of Diamonds", 6);
 24
                                                                                   Card card32 = new Card();
                                                                   78
 25
                 card6.setCard("Seven of Hearts", 7);
                 Card card7 = new Card();
                                                                   79
                                                                                   card32.setCard("Seven of Diamonds", 7);
 26
 27
                 card7.setCard("Eight of Hearts", 8);
                                                                                   Card card33 = new Card();
                                                                                   card33.setCard("Eight of Diamonds", 8);
                 Card card8 = new Card();
                                                                   81
 28
 29
                 card8.setCard("Nine of Hearts", 9);
                                                                   82
                                                                                   Card card34 = new Card();
                                                                                   card34.setCard("Nine of Diamonds", 9);
 30
                 Card card9 = new Card();
                 card9.setCard("Ten of Hearts", 10);
                                                                                   Card card35 = new Card();
                                                                   84
 31
                                                                                   card35.setCard("Ten of Diamonds", 10);
 32
                 Card card10 = new Card();
                                                                   85
                 card10.setCard("Jack of Hearts", 11);
                                                                                   Card card36 = new Card();
 33
                                                                   86
                 Card card11 = new Card();
                                                                                   card36.setCard("Jack of Diamonds", 11);
 34
                                                                                   Card card37 = new Card();
                 card11.setCard("Queen of Hearts", 12);
 35
                                                                   88
                                                                                   card37.setCard("Queen of Diamonds", 12);
                 Card card12 = new Card();
                                                                   89
                 card12.setCard("King of Hearts", 13);
 37
                                                                   90
                                                                                   Card card38 = new Card();
                                                                   91
                                                                                   card38.setCard("King of Diamonds", 13);
 38
                 Card card13 = new Card();
 39
                 card13.setCard("Ace of Hearts", 14);
                                                                   92
                                                                                   Card card39 = new Card();
                                                                                   card39.setCard("Ace of Diamonds", 14);
 40
                 //Spades
                                                                   93
                 Card card14 = new Card();
                                                                                   //Clubs
 41
                                                                                   Card card40 = new Card();
 42
                 card14.setCard("Two of Spades", 2);
                                                                   95
                                                                                   card40.setCard("Two of Clubs", 2);
 43
                 Card card15 = new Card();
                                                                   96
                                                                                   Card card41 = new Card();
                 card15.setCard("Three of Spades", 3);
 44
                                                                                   card41.setCard("Three of Clubs", 3);
                                                                   98
 45
                 Card card16 = new Card();
                                                                                   Card card42 = new Card();
                 card16.setCard("Four of Spades", 4);
                                                                   99
 46
                 Card card17 = new Card();
                                                                                   card42.setCard("Four of Clubs", 4);
 47
                                                                                   Card card43 = new Card();
                 card17.setCard("Five of Spades", 5);
                                                                  L01
 48
                 Card card18 = new Card();
                                                                                   card43.setCard("Five of Clubs", 5);
 49
                                                                  102
                                                                                   Card card44 = new Card();
 50
                 card18.setCard("Six of Spades", 6);
                                                                  L03
                                                                                   card44.setCard("Six of Clubs", 6);
                 Card card19 = new Card();
                                                                  104
 51
                                                                                   Card card45 = new Card();
 52
                 card19.setCard("Seven of Spades", 7);
                                                                  105
 53
                 Card card20 = new Card();
                                                                  L06
                                                                                   card45.setCard("Seven of Clubs", 7);
                                                                                   Card card46 = new Card();
card46.setCard("Eight of Clubs", 8);
                 card20.setCard("Eight of Spades", 8);
                                                                  107
 54
 55
                 Card card21 = new Card();
                                                                  108
                                                                  L09
                 card21.setCard("Nine of Spades", 9);
                                                                                   Card card47 = new Card();
                     card47.setCard("Nine of Clubs", 9);
110
111
                     Card card48 = new Card();
                     card48.setCard("Ten of Clubs", 10);
112
113
                     Card card49 = new Card();
                     card49.setCard("Jack of Clubs", 11);
114
115
                     Card card50 = new Card();
                     card50.setCard("Queen of Clubs", 12);
116
117
                     Card card51 = new Card();
118
                     card51.setCard("King of Clubs", 13);
119
                     Card card52 = new Card();
                     card52.setCard("Ace of Clubs", 14); //52 = have all the cards
120
```

121

```
122
                 cards = Map.ofEntries(
123
                     /*populate a map of the 52 cards, value linked to name(key)
124
                     *Map use allows lookup of values by methods later in the program
125
                     name is the key b/c of uniqueness vs. duplicate values*/
126
                     Map.entry(card1.getCardName(), card1.getCardValue()),
127
                     Map.entry(card2.getCardName(), card2.getCardValue()),
128
                     Map.entry(card3.getCardName(), card3.getCardValue()),
129
                     Map.entry(card4.getCardName(), card4.getCardValue()),
130
                     Map.entry(card5.getCardName(), card5.getCardValue()),
131
                     Map.entry(card6.getCardName(), card6.getCardValue()),
132
                     Map.entry(card7.getCardName(), card7.getCardValue()),
133
                     Map.entry(card8.getCardName(), card8.getCardValue()),
134
                     Map.entry(card9.getCardName(), card9.getCardValue()),
135
                     Map.entry(card10.getCardName(), card10.getCardValue()),
                     Map.entry(card11.getCardName(), card11.getCardValue()),
136
137
                     Map.entry(card12.getCardName(), card12.getCardValue()),
138
                     Map.entry(card13.getCardName(), card13.getCardValue()),
139
                     Map.entry(card14.getCardName(), card14.getCardValue()),
                     Map.entry(card15.getCardName(), card15.getCardValue()),
140
141
                     Map.entry(card16.getCardName(), card16.getCardValue()),
142
                     Map.entry(card17.getCardName(), card17.getCardValue()),
143
                     Map.entry(card18.getCardName(), card18.getCardValue()),
144
                     Map.entry(card19.getCardName(), card19.getCardValue()),
145
                     Map.entry(card20.getCardName(), card20.getCardValue()),
146
                     Map.entry(card21.getCardName(), card21.getCardValue()),
147
                     Map.entry(card22.getCardName(), card22.getCardValue()),
148
                     Map.entry(card23.getCardName(), card23.getCardValue()),
149
                     Map.entry(card24.getCardName(), card24.getCardValue()),
150
                     Map.entry(card25.getCardName(), card25.getCardValue()),
151
                     Map.entry(card26.getCardName(), card26.getCardValue()),
152
                     Map.entry(card27.getCardName(), card27.getCardValue()),
153
                     Map.entry(card28.getCardName(), card28.getCardValue()),
154
                     Map.entry(card29.getCardName(), card29.getCardValue()),
155
                     Map.entry(card30.getCardName(), card30.getCardValue()),
156
                     Map.entry(card31.getCardName(), card31.getCardValue()),
157
                     Map.entry(card32.getCardName(), card32.getCardValue()),
158
                     Map.entry(card33.getCardName(), card33.getCardValue()),
159
                     Map.entry(card34.getCardName(), card34.getCardValue()),
160
                     Map.entry(card35.getCardName(), card35.getCardValue()),
161
                     Map.entry(card36.getCardName(), card36.getCardValue()),
162
                     Map.entry(card37.getCardName(), card37.getCardValue()),
163
                     Map.entry(card38.getCardName(), card38.getCardValue()),
164
                     Map.entry(card39.getCardName(), card39.getCardValue()),
165
                     Map.entry(card40.getCardName(), card40.getCardValue()),
166
                     Map.entry(card41.getCardName(), card41.getCardValue()),
167
                     Map.entry(card42.getCardName(), card42.getCardValue()),
168
                     Map.entry(card43.getCardName(), card43.getCardValue()),
169
                     Map.entry(card44.getCardName(), card44.getCardValue()),
170
                     Map.entry(card45.getCardName(), card45.getCardValue()),
171
                     Map.entry(card46.getCardName(), card46.getCardValue()),
172
                     Map.entry(card47.getCardName(), card47.getCardValue()),
173
                     Map.entry(card48.getCardName(), card48.getCardValue()),
174
                     Map.entry(card49.getCardName(), card49.getCardValue()),
175
                     Map.entry(card50.getCardName(), card50.getCardValue()),
176
                     Map.entry(card51.getCardName(), card51.getCardValue()),
177
                     Map.entry(card52.getCardName(), card52.getCardValue())
178
                 );
179
             }
180
```

```
181⊖
        public void shuffle() {
            //Card names from Map moved into a List for calling indexes, values called later from Map by name
182
            List<String> shuffledCards = new ArrayList<>(cards.keySet());
183
184
            this.shuffledCards = shuffledCards;
            Collections.shuffle(shuffledCards); //shuffle method used from imported Collections library
185
186
        //select top card of shuffled deck and then remove
187
188⊖
        public String draw() {
            String drawnCard = shuffledCards.get(0);
189
190
            shuffledCards.remove(0);
191
            return drawnCard;
192
         //lookup card value in Map table
193
        public Integer retrieveValue(String cardName) {
194⊖
195
            return cards.get(cardName);
196
197 }
```

```
Player.java X
 package week6FinalProject;
 3⊕ import java.util.ArrayList;
 6 public class Player {
 7
         private String name;
 8
         private List <String> hand;
 9
         private int score = 0;
 10
         public Player() {
 11⊖
 12
             List<String> hand = new ArrayList<>();
 13
             this.hand = hand;
 14
         }
 15
 16⊖
         public void describe() {
 17
             System.out.println(name);
             System.out.println(score);
 18
 19
         }
 20
 21
 22⊖
         public void setPlayer(String name) {
 23
             this.name = name;
 24
 25
 26⊖
         public String getPlayerName() {
 27
             return this.name;
 28
 29
 30⊕
         public int getScore() {
 31
             return score;
 32
 33
 34
         //build up the player's cards with loop in Main
 35⊕
         public void draw(Deck deck) {
 36
             hand.add(deck.draw());
 37
 38
 39⊕
         public Integer flip() {
             //select top card of hand and remove after
 40
 41
             String cardFromHand = hand.get(0);
 42
             hand.remove(0);
 43
             Deck deck = new Deck();
 44
             //look up card value by name
 45
             return deck.retrieveValue(cardFromHand);
 46
         }
 47
48⊝
         public void incrementScore() {
 49
             this.score++;
50
51 }
```

# **Screenshots of Running Application:**

Confirming population of Map deck (was testing with an abbreviated card list):

# Shuffling the Map deck (as a List):

Looking up value of "Two of Hearts" correctly returns the card's value of 2:

© Console X № Problems ② Declaration

<terminated> App (1) [Java Application] D:\Software Development\Tools\Eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32x86\_64\_17.0.1.v20211116-165

{Two of Hearts=2, Queen of Hearts=12, Jack of Hearts=11, Ace of Hearts=14, Five of Hearts=5, Seven of Hearts=7, King of Hearts=13, Ten of Hearts=10, Nine of Hearts=9, Eight of Hearts=8, Six of Hearts=6, Four of Hearts=4, Three of Hearts=3}

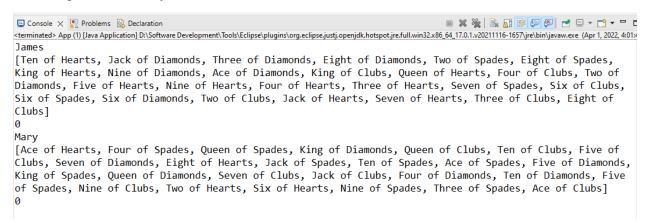
[Ace of Hearts, Six of Hearts, Ten of Hearts, Eight of Hearts, Two of Hearts, Three of Hearts, Seven of Hearts, Jack of Hearts, Queen of Hearts, Five of Hearts, Nine of Hearts, King of Hearts, Four of Hearts]

2

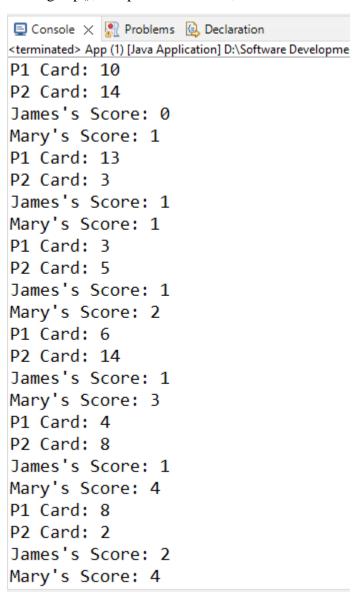
#### Draw top card and check value:

#### Full deck check:

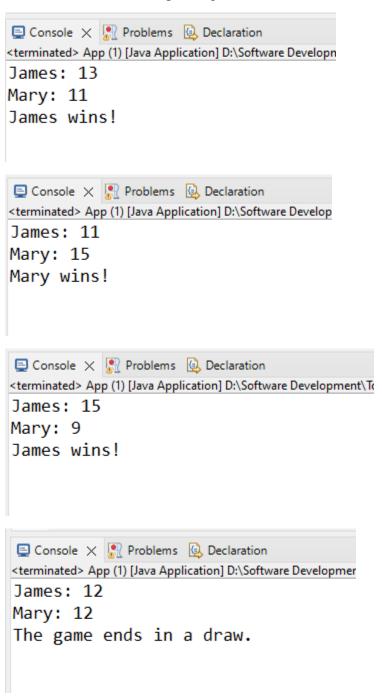
## Checking that each Player's hand was filled with their 26 cards:



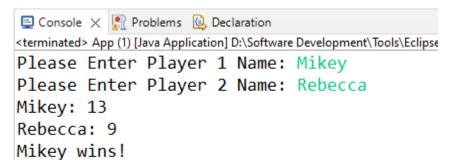
Testing flip(), comparison of result, and incrementScore():



Some outcomes of a completed game:



User input (scanner in) for player names:



# **URL** to GitHub Repository:

https://github.com/tylerjlivermore/Week6\_CardsWar