```
package cardgame;


public class Card
      //Use Enum to enable variables
{
    enum Rank
   {
    Ace,
    Two,
    Three,
    Four,
    Five,
    Six,
    Seven,
    Eight,
    Nine,
    Ten,
    Jack,
    Queen,
    King;
   }
    enum Suit
   {
   Hearts,
   Clubs,
   Diamonds,
   Spades;
   }
     //Make variables final as they won't be changing throughout the course
     //of the game
   public final Suit suit;
```

```
    public final Rank rank;

    Card(Suit suit, Rank rank) {
        this.suit = suit;
        this.rank = rank;
    }
    //creating methods to pull these variables
    Rank getRank()
    {
        return rank;
    }

    Suit getSuit()
    {
        return suit;
    }

    @Override
    public String toString()
    {
        return rank + " of " + suit;
    }
}

package cardgame;

import cardgame.Card.Rank;
import cardgame.Card.Suit;
import java.util.Random;

public class Deck {
```

```
public static final int SIZE = 52;
private final Card[] cards = new Card[SIZE];
//here im setting the deck size to 52
//making deck count start at 0, working its way up adding new
//suits and values to each card up to 52
Deck() {
   int currentCardIndex = 0;

   for (Suit suit : Suit.values()) {
      for (Rank rank : Rank.values()) {
         cards[currentCardIndex++] = new Card(suit, rank);
      }
   }
}

Card[] getCards()
{
   return cards;
}

Card getCard(int index)
{
   return cards[index];
}
//all cards are returned and the deck is shuffled
void shuffleDeck()
{
   Random rand = new Random();

   for (int i = 0; i < SIZE; i++)
   {
```

```
            int j = rand.nextInt(SIZE);
            swapCards(i, j);
        }
    }

    void swapCards(int i, int j)
    {
        Card temp = cards[i];
        cards[i] = cards[j];
        cards[j] = temp;
    }

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

        stringBuilder.append("Current Deck:\n");

        for (int i = 0; i < Deck.SIZE; i++)
        {
            stringBuilder.append("Card #" + (i + 1) + ": " + getCard(i) + "\n");
        }
        return stringBuilder.toString();
    }

}
```

```
package cardgame;

import java.util.ArrayList;
```

```
import java.util.List;


public class Player
{
   private String name;
   private List<Card> cards = new ArrayList<>();
   // array list, grows to accomodate new elements and shrinks when
   // others are removed
   Player(String name)
   {
      this.name = name;
   }
   // getting the users name
   void giveCard(Card card)
   {
      cards.add(card);
   }
   //give the user their set of cards
   List<Card> getCards()
   {
      return cards;
   }
   // creating a string that presents each players 13 hand
   String printPlayerCards()
   {
      StringBuilder stringBuilder = new StringBuilder();
      stringBuilder.append(name + " has the following cards:\n");


      for (Card card : cards)
      {
         stringBuilder.append(card + "\n");
```

```
        }
            return stringBuilder.toString();
    }

    @Override
    public String toString()
    {
        return name;
    }
}

package cardgame;

import java.util.Scanner;

public class Game
{
private static final int NO_OF_PLAYERS = 4;
private static final Player[] players = new Player[NO_OF_PLAYERS];
private static final Deck deck = new Deck();
//making the game a 4 player game
public static void main(String[] args)
{
    Game game = new Game();
    // adding prompts for the output
    System.out.println("Welcome to Bridges Card Game\n");
    System.out.println("Enter the four players' names");

    Scanner scan = new Scanner(System.in);
    for (int i = 0; i < NO_OF_PLAYERS; i++)
    {
```

```
        Game.players[i] = new Player(scan.next());
    }
    // scanner easiest way to read input of users
    Game.deck.shuffleDeck();
    // calling methods
    System.out.println(game.deck);

    Game.dealCards();

    Game.displayCardsForAllPlayers();
}
// cards are dealt finally
private static void dealCards()
{
    for (int i = 0; i < Deck.SIZE; i++)
    {
        players[i % NO_OF_PLAYERS].giveCard(deck.getCard(i));
    }
} // all cards are displayed and who has what
private static void displayCardsForAllPlayers()
{
    for (int i = 0; i < NO_OF_PLAYERS; i++)
    {
        System.out.println(players[i].printPlayerCards());
    }
}
}
```

```
package cardgame;

public class Interface {
```

```
    public interface Card{
        int getRank();
        int getSuit();
        int shuffleDeck();
        int dealCards();
        int displayCardsForAllPlayers();
    }
}
// interface containing methods
```