Card Deck Shuffling in Java

Student's Name

Department, Institutional Affiliation

Course Number and Name

Instructor's Name

Due Date

Card Deck Shuffling in Java

Introduction

Simulating the shuffling and dealing of cards is something that can be achieved using a simple Java program. First of all, one must be familiar with a deck of cards and their constituents. For the sake of this project, I will list them:

- The number of cards in a deck is 52.
- The cards in the deck have 4 distinct suits (Clubs, Diamonds, Spades, and Hearts).
- Standard cards possess these identifiers: Ace, 2, 3, 4, 5, 6, 7, 8, 9, 10, Jack, Queen, and King.
- Let us demonstrate an operation performed on the cards by defining the above parameters so that the program contains all the cards in the deck, shuffles them, then picks the first 10 from the shuffled list and displays them (including suit and numbers).

Card Declaration

```
//Declaring the class
public class RandomCards {
    //creating a array of suits
    static String[] suits = {"Clubs", "Hearts", "Spades", "Diamonds"};

    // Array of strings for the numbers to words the index is converted
                        //  [0]     [1]   [2]       [3]  ect.....
    static String[] numberwords = {"Ace","Two", "Three", "Four", "Five", "Six", "Seven", "Eight", "Nine", "Ten", "Jack",
            "Queen", "King", };
    //creating the array of strings to contain the cards
   static String[] deck = new String[52];
```

Card Generation

Once the cards have been defined and declared, a code to generate them and place them in an array is implemented.

```
static void GenerateDeck() {
      //Generates the deck of cards with
    for(int i = 0; i< deck.length; i++) {
    //creates the deck      remainder of each index
        deck[i] = numberwords[i%13] + " of " + suits[i/13];
        }
}
```

Card Shuffling

The best way to shuffle cards in programming is by using the random functions; the cards are randomized by the computer itself so that the order changes every time the **random ()** function is called. The random () function is a generator for uniformly distributed sequences (Schildt, 2014).

```
//This shuffles the cards
static void Draw() {
    Random rand = new Random();
    //Shuffling the deck aka randomizing it
    for(int i = 0; i<deck.length; i++) {
        //random number index
        int index = rand.nextInt(52);
        //creating temporary place holder for the card
        String a = deck[i];
        //moving the placeholder card to the random index
        deck[i] = deck[index];
        // making the random number the next placeholder.
        deck[index] = a;
    }
}
```

In the code above, the cards were automatically randomized, as no specific order was defined. Now, we select the first 5 cards from the deck and assign them to the computer player, then the next 5 cards in the pile are selected and assigned to the human player. Here is the code for assigning the cards to both the player and the computer.

```
static void Deal() {
    //Shuffling the cards
    Draw();
    //printing computers cards
    System.out.println("*****Computer*****");
    // picking the first 5 cards off of the top of the deck
    for(int i = 0; i<5; i++) {
        System.out.println(deck[i]);
    }
    //printing players cards
    System.out.println("*****Player*****");
    //picking the next 5 cards from the top of the deck
    for(int i = 5; i<10; i++) {
        System.out.println(deck[i]);
    }
}
```

Main Function

The main function just calls the two functions which run the program, that is, the **GenerateDeck ()** and the **Deal ()** functions declared above.

```
public static void main(String[] args) {
    // Making the deck
    GenerateDeck();
    //Shuffling (Draw) Then Dealing
    Deal();
}
}
```

Conclusion

The program will give a different outcome every time the main function is run because the values are randomized. It displays the five random cards that are assigned to the computer and then assigns the next five to the human user.

Reference

Schildt, H. (2014). *Java: The Complete Reference.* Oracle Press.