Exposure Java	Lab 11b
The <Deck> Class Program	80 & 100 Point Versions

Assignment Purpose:

This assignment is meant to demonstrate how to use a static one-dimensional array in a class, including assigning values and displaying the array.

Chapter IX introduced the **Card** class. This class was used to explain encapsulation. The **Card** class is also a fundamental class in the *Elevens AP® Lab*. Nothing was mentioned about the Card Game program that is a form of solitaire, called *Elevens.* In this chapter the previously introduced **Card** class, is now used to create a **Deck** class. In a future chapter both the **Card** class and the **Deck** class will be used in the Elevens solitaire card game.

Chapter XI showed how the **cards** array is used as an attribute in the **Deck** class to store **Card** objects. This lab assignment is meant to improve the **Deck** class. The Student Starting version, shown below, shows a minimal **Deck** class.

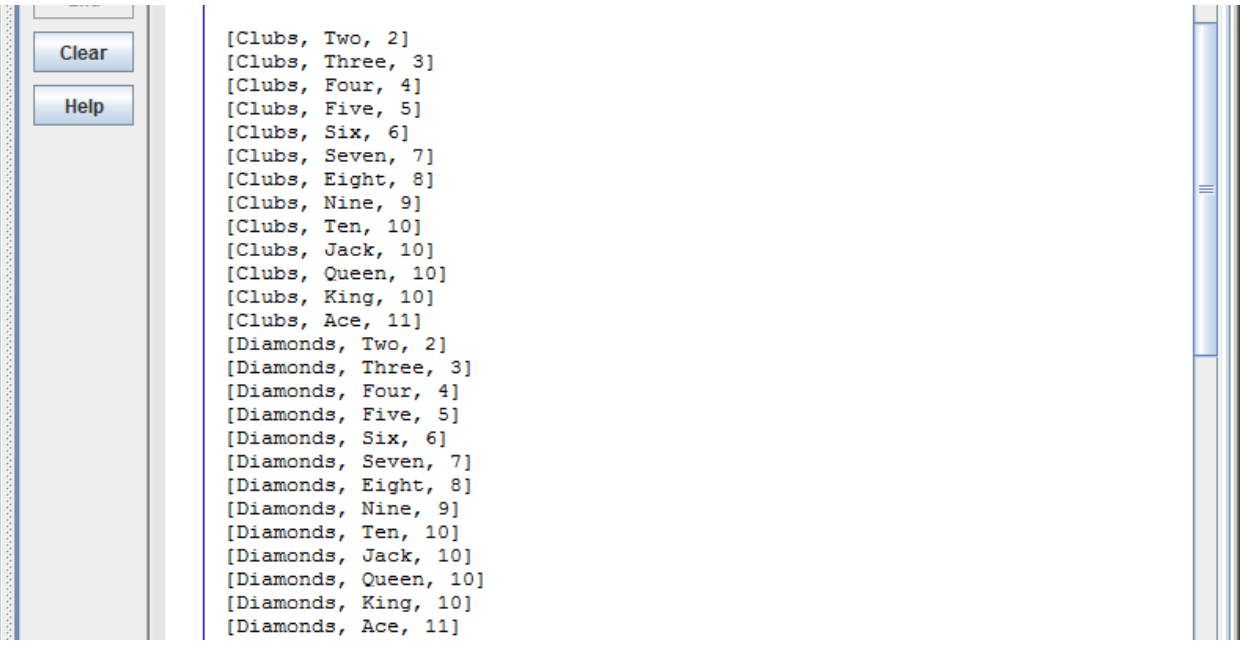
Lab11bvst Student Version	Do not copy this file, which is provided.

```
// Lab11bvst.java
// This is the Student starting version for the <Deck> class lab 11b assignment.


public class Lab11bvst
{
   public static void main(String[] args)
   {
      Deck deck = new Deck();
      System.out.println(deck);
   }
}


class Deck
{
   private Card[] cards;
   private int size;

   public Deck()
   {
      size = 52;
      cards = new Card[size];
   }
 }
```

80 Point Version Specifics

For the 80-point version you need to rewrite the constructor so that all 52 cards of a normal card deck are assigned to the **cards** array. Keep in mind that card information needs to be stored inside the **Deck** class and is not passed by parameter. Additionally, you need to re-define the **toString** method for the **Deck** class so that it can be used to display the attribute values in a convenient manner. Make sure to take advantage of the **toString** method that already exists in the **Card** class.

80 Point Version Output



```
[Hearts, Two, 2]
[Hearts, Three, 3]
[Hearts, Four, 4]
[Hearts, Five, 5]
[Hearts, Six, 6]
[Hearts, Seven, 7]
[Hearts, Eight, 8]
[Hearts, Nine, 9]
[Hearts, Ten, 10]
[Hearts, Jack, 10]
[Hearts, Queen, 10]
[Hearts, King, 10]
[Hearts, Ace, 11]
[Spades, Two, 2]
[Spades, Three, 3]
[Spades, Four, 4]
[Spades, Five, 5]
[Spades, Six, 6]
[Spades, Seven, 7]
[Spades, Eight, 8]
[Spades, Nine, 9]
[Spades, Ten, 10]
[Spades, Jack, 10]
[Spades, Queen, 10]
[Spades, King, 10]
[Spades, Ace, 11]
```

Hint:

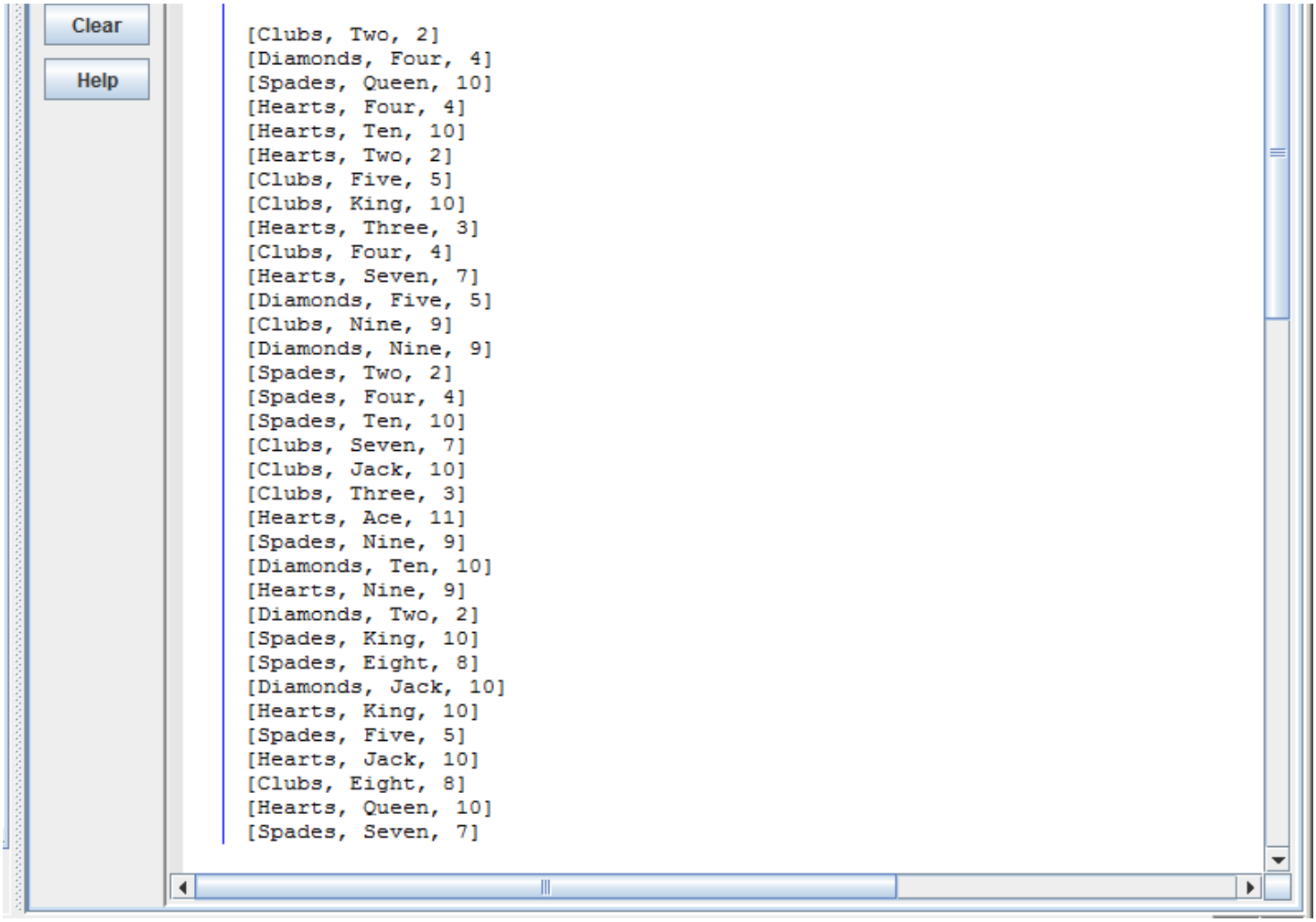
You can store attributes in the **Deck** class that have an initializer list as shown below:

private String[ ] suits = {"Clubs","Diamonds","Hearts","Spades"};

100 Point Version Specifics

For the 100-point version you need to add a **shuffle** method, which is called from the constructor. The **shuffle** method is a **private** *helper method* in the **Deck** class. For this version you need to *shuffle* the deck by swapping the cards. Generate two random numbers in the **[0..51]** number range that will represent the indexes of the **cards** array and swap the cards. Make 1000 swaps and then display the cards. Use **Math.random** to generate random numbers.

100 Point Version Partial Output


Clear
Help
[Clubs, Two, 2]
[Diamonds, Four, 4]
[Spades, Queen, 10]
[Hearts, Four, 4]
[Hearts, Ten, 10]
[Hearts, Two, 2]
[Clubs, Five, 5]
[Clubs, King, 10]
[Hearts, Three, 3]
[Clubs, Four, 4]
[Hearts, Seven, 7]
[Diamonds, Five, 5]
[Clubs, Nine, 9]
[Diamonds, Nine, 9]
[Spades, Two, 2]
[Spades, Four, 4]
[Spades, Ten, 10]
[Clubs, Seven, 7]
[Clubs, Jack, 10]
[Clubs, Three, 3]
[Hearts, Ace, 11]
[Spades, Nine, 9]
[Diamonds, Ten, 10]
[Hearts, Nine, 9]
[Diamonds, Two, 2]
[Spades, King, 10]
[Spades, Eight, 8]
[Diamonds, Jack, 10]
[Hearts, King, 10]
[Spades, Five, 5]
[Hearts, Jack, 10]
[Clubs, Eight, 8]
[Hearts, Queen, 10]
[Spades, Seven, 7]