Lab 11  Name:__________________________  Checked:______

Objectives:
More practice using arrays:
   1. Arrays of Strings and shuffling an array
   2. Arrays as parameters
   3. Collections

Preparation
1) An array to store a deck of cards: DeckOfCards.java
This program should have a structure similar to those for Lab 10, i.e., declare and instantiate an array and use two for-loops, one to initialize the array and one to print its contents. Through following steps you will be guided to initialize the array using the contents of two smaller arrays.

a. Use initializer lists to create the following arrays:
   - An array of 4 Strings to represent the suits: “hearts”, “spades”, “diamonds”, “clubs”
   - An array of 13 Strings to represent the ranks: “2”, “3”, …, “10”, “Jack”, “Queen”, “King”, “Ace”

b. Declare and instantiate a third array of 52 Strings to represent the deck of 52 playing cards.
   DO NOT use an initializer list to set its values (do not initialize at all yet).

c. Print the contents of all three arrays.
   What happens if you do not initialize the array’s values as is the case of the array deck?

   Answer:_________________________________________________________________________________________

d. Use the arrays for suits and ranks, above, to initialize the deck, so that it consists of Strings like “4 of hearts” or “King of clubs”, i.e., a rank, followed by the string “ of “ followed by a suit.

   Hints:
   - See example of nested for-each loops we did in class.
   - Use a nested for-each loop to combine each suit with each rank, for example “hearts” and “4”, concatenating them to make Strings such as “4 of hearts”
   - As the strings are combined, they should be stored as one of the 52 array elements representing the deck. Thus, you will need a counter to index into the deck array. You need to start that at zero and increment it each time an entry is added (in the inner loop).

   suits
   | 0 | 1 | 2 | 3 |
   ---|---|---|---|---|
   hearts | spades | diamonds | clubs |

   ranks
   | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
   ---|---|---|---|---|---|---|---|---|---|---|---|---|---|
   2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | jack | queen | king | ace |

   deck
   | 0 | 1 | 2 | 3 | 4 | ... | 50 | 51 |
   ---|---|---|---|---|---|---|---|---|
   2 of hearts | 3 of hearts | 4 of hearts | ... | king of clubs | ace of clubs |
2) Arrays as parameters: Shapes.java

a. Run the example Shapes.java. Note the use of arrays as parameters.

b. Change the numbers so that you get a nice blue V on a white triangle.

c. Write a method addX that adds x (a double) to the value of each element in an array of type double[]. (Note: this is one of the exercises we did in class)
Use addX() in the Shapes class to shift the blue V to the right.

d. Experiment: use the method addX to draw additional triangles. Note that with this method you have essentially created an alternative to the JavaFX Group translateX() or translateY() methods that we have been using.

   In what ways is this similar?

   In what ways is this different?

e. Finish by creating a design of your choice using your method three or more times, with different parameters.

------------------------------------
Submit DeckOfCards.java and Shapes.java through Blackboard by 8:00am the morning of the Lab.
Part A. – Shuffling in DeckOfCards.java
1. Compare your work with your partner’s. Verify that arrays are declared, instantiated, and initialized correctly.

Lab partner’s signature: ________________________________

2. Add code to shuffle the deck as follows:
Create a loop to repeat the following many, many times (how many do you think would be enough? ______________)
- Generate two random numbers a and b in the range 0..51
- Exchange the values (Strings) in deck[a] and deck[b]. For example, if the numbers generated were 2 and 50, then the “4 of hearts” and the “king of clubs” would switch places in the array.

After the deck is shuffled, print it again and behold the cards in random order!

Part B. – Shuffling the DVDCollection
1. Review the code for DVD.java, DVDCollection.java, and Movies.java (client). Modify the client code to add a few more movies so that you have at least 10 to work with.
2. Add a shuffle() method to the DVDCollection class. Use it in the client to shuffle the DVDs and print them again, now in some random order.

3. [Optional challenge] Add another method the DVDCollection class, to sort the DVD’s according to increasing price. Can you think of a way to do that?
One well known sorting method proceeds as follows:
- In a loop, go through the list examining adjacent elements and exchanging them if they are out of order. Obviously, going through the list once will NOT succeed in sorting the elements, but will at least get the largest one to move to the last position in the array. If you repeat this N times, where N is the length of the array, you are sure to get all of them sorted in order. This sorting algorithm is called “bubble sort”. It is not terribly efficient, but is interesting to implement and to analyze.

Part C. – Add some more polygons to Shapes.java
1. Compare your design with your partner’s. Sign and optionally comment on their design – take a selfie together with your V’s – Go Cats!

Lab partner’s comments: ________________________________

Lab partner’s signature: ________________________________

2. Add some more code to display a pentagon around the design (to make it look like it is inside a house maybe?).

3. Use the addX() method to display the pentagon in 3 or more positions, in different colors.
Lab 11 Comments

Comments on this lab, please:

What was the most valuable thing you learned in this lab?

What did you like best about this lab?

Was there any particular problem?

Do you have any suggestions for improving this lab as an effective learning experience?