Course : OO Programming with Java

Date: October 27, 2017

Student Name:

Lab No. 4

Problem 1

(*Print distinct numbers*) Write a program that reads in ten numbers and displays the number of distinct numbers and the distinct numbers separated by exactly one space (i.e., if a number appears multiple times, it is displayed only once). (*Hint*: Read a number and store it to an array if it is new. If the number is already in the array, ignore it.) After the input, the array contains the distinct numbers. Here is the sample run of the program:

```
Enter ten numbers: 1 2 3 2 1 6 3 4 5 2
The number of distinct number is 6
The distinct numbers are: 1 2 3 6 4 5
```

Problem 2

Card Shuffling and Dealing) Modify the application of Fig. 7.9-11 to deal a five-card poker hand. Then modify class `DeckOfCards` of Fig. 7.10 to include methods that determine whether a hand contains

a) a pair

b) two pairs

c) three of a kind (e.g., three jacks)

d) four of a kind (e.g., four aces)

e) a flush (i.e., all five cards of the same suit)

f) a straight (i.e., five cards of consecutive face values)

g) a full house (i.e., two cards of one face value and three cards of another face value)

Sample output expected as follows

```
Left hand:          Right hand:
Queen of Hearts     Three of Hearts
Four of Diamonds    King of Spades
Three of Spades     Queen of Spades
Jack of Hearts      Eight of Clubs
Eight of Spades     Queen of Diamonds

Hand Values:
none                One Pair

Result: right hand is better
```

Problem 3

(*Pattern recognition: consecutive four equal numbers*) Write the following method that tests whether the array has four consecutive numbers with the same value.

```
public static boolean isConsecutiveFour(int[] values)
```

Write a test program that prompts the user to enter a series of integers and displays if the series contains four consecutive numbers with the same value. Your program should first prompt the user to enter the input size—i.e., the number of values in the series. Here are sample runs:

```
Enter the number of values: 8 Enter
Enter the values: 3 4 5 5 5 5 4 5 Enter
The list has consecutive fours
```

```
Enter the number of values: 9 Enter
Enter the values: 3 4 5 5 6 5 5 4 5 Enter
The list has no consecutive fours
```

(*Simulation: coupon collector's problem*) Coupon collector is a classic statistics problem with many practical applications. The problem is to pick objects from a set of objects repeatedly and find out how many picks are needed for all the objects to be picked at least once. A variation of the problem is to pick cards from a shuffled deck of 52 cards repeatedly and find out how many picks are needed before you see one of each suit. Assume a picked card is placed back in the deck before picking another. Write a program to simulate the number of picks needed to get four cards from each suit and display the four cards picked (it is possible a card may be picked twice). Here is a sample run of the program:

```
Queen of Spades
5 of Clubs
Queen of Hearts
4 of Diamonds
Number of picks: 12
```

Problem 5

(*Game: pick four cards*) Write a program that picks four cards from a deck of 52 cards and computes their sum. An Ace, King, Queen, and Jack represent 1, 13, 12, and 11, respectively. Your program should display the number of picks that yields the sum of 24.

Problem 6

(*Merge two sorted lists*) Write the following method that merges two sorted lists into a new sorted list.

```
public static int[] merge(int[] list1, int[] list2)
```

Implement the method in a way that takes at most **`list1.length + list2.length`** comparisons. Write a test program that prompts the user to enter two sorted lists and displays the merged list. Here is a sample run. Note that the first number in the input indicates the number of the elements in the list. This number is not part of the list

```
Enter list1: 5 1 5 16 61 111 Enter
Enter list2: 4 2 4 5 6 Enter
The merged list is 1 2 4 5 5 6 16 61 111
```

Problem 7

(*Algebra: multiply two matrices*) Write a method to multiply two matrices. The definition of the method is:

```
public static double[][] multiplyMatrix(double[][] a, double[][] b)
```

To multiply matrix `a` by matrix `b`, the number of columns in `a` must be the same as the number of rows in `b`, and the two matrices must have elements of the same or compatible types. Let `c` be the result of the multiplication. Assume the column size of matrix a is `n`. Each element *cij* is

$$a_{i1} \times b_{1j} + a_{i2} \times b_{2j} + \ldots + a_{in} \times b_{nj}.$$

For example, for two 3 * 3 matrices `a` and `b`, `c` is

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \times \begin{pmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{pmatrix} = \begin{pmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{pmatrix}$$

where $c_{ij} = a_{i1} \times b_{1j} + a_{i2} \times b_{2j} + a_{i3} \times b_{3j}$.

Write a test program that prompts the user to enter two 3 * 3 matrices and displays their product.

```
Enter matrix1: 1 2 3 4 5 6 7 8 9 Enter
Enter matrix2: 0 2 4 1 4.5 2.2 1.1 4.3 5.2 Enter
The multiplication of the matrices is
 1 2 3      0 2.0 4.0      5.3 23.9 24
 4 5 6   *  1 4.5 2.2  =   11.6 56.3 58.2
 7 8 9      1.1 4.3 5.2    17.9 88.7 92.4
```

Problem 8

(*Shuffle rows*) Write a method that shuffles the rows in a two-dimensional `int` array using the following header:

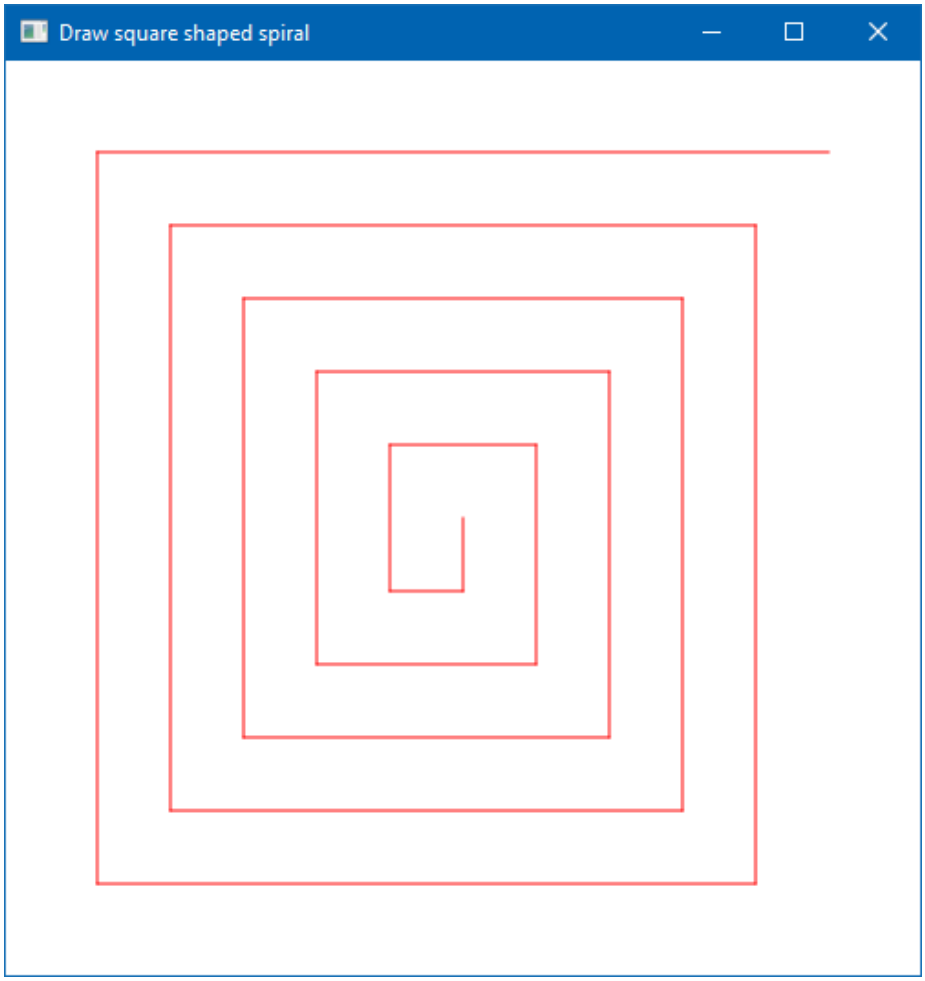
```
public static void shuffle(int[][] m)
```

Write a test program that shuffles the following matrix:

```
int[][] m = {{1, 2}, {3, 4}, {5, 6}, {7, 8}, {9, 10}};
```

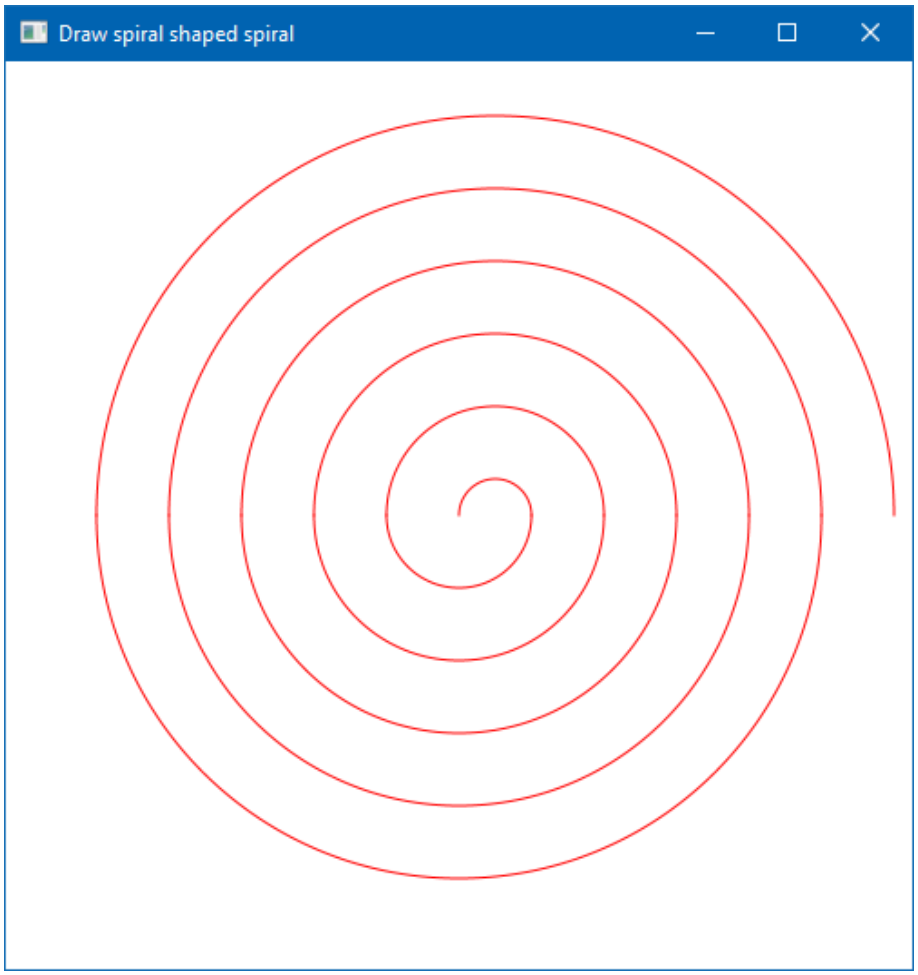
Problem No. 11.

Draw a square-shaped spiral (as in the screen capture given below), centered in the application window, using a JavaFX application. One technique is to use a loop that increases the line length after drawing every second line. The direction in which to draw the next line should follow a distinct pattern, such as down, left, up, right.



Problem No. 11.

Draw a circular spiral (as in the screen capture given below), using a JavaFX application to draw one semicircle at a time. Each successive semicircle should have a larger radius (as specified by the bounding rectangle's width) and should continue drawing where the previous semicircle finished


Draw spiral shaped spiral