Due Date:	By 11:59pm November 17, 2017
Evaluation:	4% of final mark (see marking rubric at the end of handout)
Late Submission:	none accepted
Purpose:	The purpose of this assignment is to help you learn primitive type arrays, strings arrays, 2-dimensional arrays and simple class design.
CEAB/CIPS Attributes:	Design/Problem analysis/Communication Skills

General Guidelines When Writing Programs:
- Please refer to the handout of Assignment #1.

Question 1 (One Dimensional Array)

Write a program that asks a user for names of 8 soccer teams and outputs a possible road map of 4 quarter finals, 2 semifinals, 1 final and the final winner of a tournament.

Your program should proceed as follows:

1. Ask the user for a name for the soccer tournament followed by the names of the 8 participating teams. (You can assume that each name, which can be more than 1 word, will be correctly entered on a separate line.) The participating teams are to be stored in a one-dimensional array.

2. Generate the output roadmap by choosing at random combinations of teams and a winner for each combination at each level to produce a final winner. Of course a team can only be in one quarter final game. Figure1 illustrates how teams are matched for the semi-finals and finals.
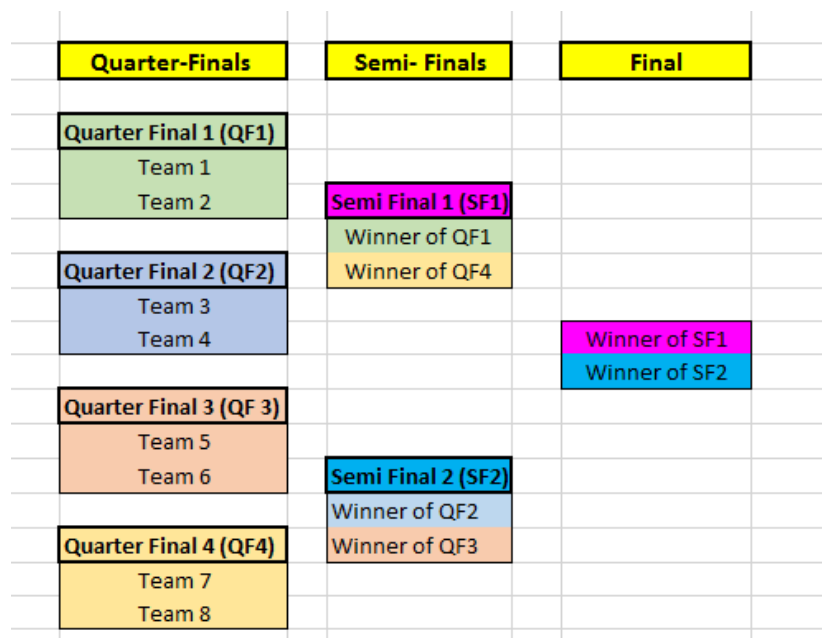

Figure 1- Elimination Tournament

Use the random number generator from the Math class or Random class to randomly pick the original combination of the 8 teams in the quarter finals and the winner of each game.

3. Your program output should be displayed in the format below.

4. The user should be able to request for a different outcome until he receives an output that he likes. This involves shuffling the teams in the array to determine who plays against whom in the quarter finals. (You can use `random.nextInt(bound)` in a loop to randomize the array- in other words to shuffle the array entries.)

5. When a user chooses to stop, your program should display a closing message like:
   `'Thank you for using the JAVA Tournament Winner Predictor Program.'`

Here are a few sample outputs to illustrate the expected behavior of your program.
<u>Note</u>: user input is highlighted in grey.

```
------------------------------------------------------
   Welcome to Tournament Outcome Predictor Program
------------------------------------------------------
Please enter a name for the soccer tournament
la liga

Please enter the 8 participating teams
Atlético Madrid
Barcelona
Getafe
Levante
Real Betis
Real Madrid
Sevilla
Valencia


        ----- la liga Outcome Predictions -----

Quater Final 1: Real Betis Vs Valencia
Valencia Wins !!!

Quater Final 2: Atlético Madrid Vs Sevilla
Atlético Madrid Wins !!!

Quater Final 3: Real Madrid Vs Barcelona
Barcelona Wins !!!

Quater Final 4: Getafe Vs Levante
Getafe Wins !!!

Semi-Final 1: Valencia Vs Getafe
Getafe Wins !!!

Semi-Final 2: Atlético Madrid Vs Barcelona
Atlético Madrid Wins !!!

Final: Getafe Vs Atlético Madrid
Getafe Wins !!!

Do you want a different outcome? y or n
y

----- la liga Outcome Predictions -----

Quater Final 1: Getafe Vs Sevilla
Getafe Wins !!!

Quater Final 2: Atlético Madrid Vs Real Betis
Real Betis Wins !!!

Quater Final 3: Valencia Vs Levante
Valencia Wins !!!

Quater Final 4: Barcelona Vs Real Madrid
Real Madrid Wins !!!

Semi-Final 1: Getafe Vs Real Madrid
Real Madrid Wins !!!

Semi-Final 2: Real Betis Vs Valencia
Valencia Wins !!!

Final: Real Madrid Vs Valencia
Real Madrid Wins !!!

Do you want a different outcome? y or n
n
Thank you for using the JAVA Tournament Winner Predictor Program
```

Question 2 (2-dimensional Arrays)

Write a program that implements a card magic game where you accurately predict a user's card from a randomly generated 4x4 card matrix. Your program should proceed as follows:

1. Generate a random 4x4 card matrix and print it on the screen.
   Tips: You could use a 1D index array for card numbers, 4x4 matrix for cards with one row each for 4 card types. You can randomize rows and columns independently using a loop to swap values at random or use the `randomizeArray()` method.

2. Ask a user to pick a card and to enter the corresponding column number for that card. (You have to make sure that the user enters a number between 1 and 4 and nothing else.)

3. Transpose the 4x4 card matrix (i.e. rows matrix becomes column matrix) and print it on the screen. Ask user for the new column number for his chosen card. (Simply swap loop variables.)
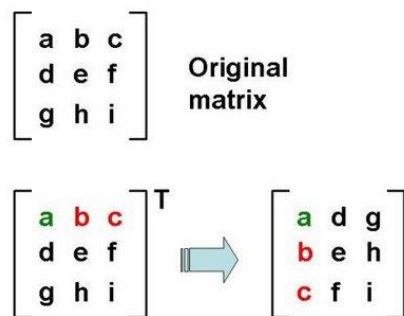
4. Make use of the two inputs from the user to predict his card. Your program output should be displayed in the format below. The user should be able to request for another try until he is satisfied that program correctly predicts the card every single time.

5. When a user chooses to stop, your program should display a closing message like:
   'Thank you for using the JAVA Magic 101 Program.'

Here is an example of the output to illustrate the expected behavior of your program.

Note: user input is highlighted in grey.

```
---------------------------------
   Welcome to Card Magic 101
---------------------------------
Nine of Diamonds     Ace of Diamonds     Ten of Diamonds      King of Diamonds
Nine of Hearts       Ace of Hearts       Ten of Hearts        King of Hearts
Nine of Clubs        Ace of Clubs        Ten of Clubs         King of Clubs
Nine of Spades       Ace of Spades       Ten of Spades        King of Spades
Please pick a card and enter the column number (1-4) where it appears
1
Nine of Diamonds     Nine of Hearts      Nine of Clubs        Nine of Spades
Ace of Diamonds      Ace of Hearts       Ace of Clubs         Ace of Spades
Ten of Diamonds      Ten of Hearts       Ten of Clubs         Ten of Spades
King of Diamonds     King of Hearts      King of Clubs        King of Spades
Please indicate which column number (1-4) it is in now
x
Please enter an integer value between 1 and 4
5
Please indicate which column number (1-4) it is in now
3
Your card is Nine of Clubs
Do you want to try one more time? y or n
y
Four of Diamonds     Six of Diamonds     Jack of Diamonds     Nine of Diamonds
Four of Clubs        Six of Clubs        Jack of Clubs        Nine of Clubs
Four of Spades       Six of Spades       Jack of Spades       Nine of Spades
Four of Hearts       Six of Hearts       Jack of Hearts       Nine of Hearts
Please pick a card and enter the column number (1-4) where it appears
3
Four of Diamonds     Four of Clubs       Four of Spades       Four of Hearts
Six of Diamonds      Six of Clubs        Six of Spades        Six of Hearts
Jack of Diamonds     Jack of Clubs       Jack of Spades       Jack of Hearts
Nine of Diamonds     Nine of Clubs       Nine of Spades       Nine of Hearts
Please indicate which column number (1-4) it is in now
4
Your card is Jack of Hearts
Do you want to try one more time? y or n
n
Thank you for using the JAVA Magic 101 Program
```

Question 3 (Simple Class Exercise)
Define a class named Car that stores information about a car. It should comprise of the following:

1. Private instance variables to store age of the car, its type (sedan or suv) and its cost.
2. 4 constructors:
   a. No argument (sets age to 0, type to sedan and cost to 32000).
   b. one argument constructor (sets cost to a value, age to 0 and type to sedan).
   c. two argument constructor (sets age to a value, cost to a value, and type to sedan).
   d. three argument constructor (sets age to a value, cost to a value, and type to value sedan or suv).

3. 3 Accessor methods: - methods to return age, type and cost respectively.
4. 5 Mutator methods: - 3 methods for setting the three values independently, a method to set all three values and a method to set only age and cost of the car.
5. A public method called `estimatePrice`() that returns the cost of a car based on type and age. A sedan costs $32000, depreciates 10% every year in first five years and 5% every year afterwards. An SUV costs $45000, depreciates 8% every year in the first five years and 4% every year afterwards.
6. A `toString()` method that returns the type of the car as well as it's age and cost.
7. An `equals()` method to test for equality of two objects of class car based on type and age.
8. `isLessThan()` and `isGreaterThan()` method to compare between the prices of two objects of class car.

b) Write a driver class

1. Which declares 4 car objects using 4 different constructors and outputs the description of the 4 cars.
2. Test your accessor methods.
3. Calculate the estimated price of the cars given the type and age (include 1 sedan and 1 SUV)
4. Test out all 5 mutator methods to modify car attributes.
5. Test methods `toString()`, `equals()`, `isLessThan()` and `isGreaterThan()` for different car objects.

Here is an example of the output to illustrate the expected behavior of your program.

```
Car C1: This car is type Sedan. Its age is 0 and costs $32000.0
Car C2: This car is type Sedan. Its age is 0 and costs $32000.0
Car C3: This car is type Sedan. Its age is 4 and costs $18000.0
Car C4: This car is type SUV. Its age is 2 and costs $36000.0

Accessor Method: The cartype for car C4 is SUV, its age is 2, and it costs $36000.0

The estimated price of car C3 is $24800.0
The estimated price of car C4 is $39240.0
Mutator Method: The new age for car C3 is 5
Mutator Method: The new cartype for car C3 is SUV
Mutator Method: The new cost for car C3 is 14000.0
Mutator Method: The new car age is 9 and its new cost is 9000.0
Mutator Method: The new cartype for car C3 is Sedan, its new age is 14and its cost is 5000.0
toString: This car is type Sedan. Its age is 14 and costs $5000.0

Cars C1 and C2 are equal is true
Cars C1 and C4 are equal is false
Car C4 is less than C3 is false
Car C1 is greater than C3 is true
```

Submitting Assignment 4

- Zip the source code (the .java file only please) of this assignment.
- Naming convention for zip file: Create one zip file, containing the source files for your assignment. The zip file should be called *a#_studentID*, where # is the number of the assignment and *studentID* is your student ID number.
  For example, for the fourth assignment, student 123456 would submit a zip file named a4_123456.zip
- Refer to your section's Moodle page for instructions on where to submit your assignment.

Evaluation Criteria for Assignment 4 (20 points)

Source Code		
Comments for all 3 questions (3 pts.)		
Description of the program (authors, date, purpose)	1	pts.
Description of variables and constants	1	pt.
Description of the algorithm	1	pts.
Programming Style for all 3 questions (2 pts.)		
Use of significant names for identifiers	1	pt.
Indentation and readability	0.5	pt.
Welcome Banner/Closing message	0.5	pt.
Question 1 (5 pts.)		
Prompting user/reading data	1	pt.
Producing random tournament results each time	3	pt.
Display results	1	pt.
Question 2 (5 pts.)		
Generating random card matrix every time	2.5	pts.
Prompting and validating user input	1	pt.
Display correct results and continue until stopped	1.5	pt.
Question 3 (5 pts.)		
Constructor-Accessor-Mutator	1.5	pts.
Comparison (=,<,>) , toString()	1	pt.
Other Methods	0.5	pts.
Test Code	2	pts.
TOTAL	**20**	**pts.**