Assignment 4

Due Sunday, 16th of November, 2014

Read the instructions below carefully. The instructions must be followed.

This assignment is worth 5% of your grade. No late assignments will be accepted.

For this assignment, you are allowed to use the concepts from the first 7 chapters of the textbook.

For this assignment a file called `a4.java` is provided for you. You should place all your code inside of this file in clearly indicated spaces and as detailed below. Place the file into a folder called A4_xxxxxx, where you should replace xxxxxx by your student number. Compress this folder into A4_xxxxxx.zip file. Submit your assignment, i.e., A4_xxxxxx.zip file, via Blackboard Learn. Submit your assignment via Blackboard Learn (as instructed in the first lab.)

Your code must compile, otherwise the assignment will be graded with mark 0. For that reason, if you run out of time and/or your program contains code that does not compile, then comment out that section of your code.

For this assignment, you should program a version of a *memory game* (to be described below). If you do not know what it is, google it or see here, for example: `http://www.brainmetrix.com/memory-game/`
Only your game will not use graphics libraries to display the board, but instead it will print the board with the tools that we have seen in class, like `System.out.print()`.

Your game should use a 2D Java array (i.e., a matrix) as the board. The player should be asked if she wants to play the game on a 2x2, 4x4 or 6x6 board. The board should be filled with the first `size/2` capital letters of the English alphabet such that each letter appears exactly twice in the 2D array (Here `size` is 4 in case of 2x2 array, 16 in case of 4x4 array and 36 in case of 6x6 array). This part is already done for you inside of `a4.java`.

Once such 2D array is created it needs to be shuffled before the game can be played (otherwise the game would be trivial as the locations of the letters would be predictable). To shuffle a 2D array, first copy it into an (1D) array. Then shuffle that 1D array. To see how to do that, look at the last page of this assignment (or alternatively Listing 2: `DeckOfCards.java` in chapter 6.4 of your book). The last page contains a program that shuffles a (1D) array of integers. Modify the part of the code labelled as "Shuffling the array" for your needs. Once you are done shuffling your 1D array, copy it back into your 2D array (i.e., back to your board). This should be done inside of the shuffleBoard method, as indicated in the `a4.java` file.

Your program, i.e., a4.java, has a playGame method that should play the whole game. You will have to program this method. It takes as input the shuffled board.

The game should be played as follows. When your program prints the board, the locations for which paring is not discovered yet, should display `*` and the locations for which the paring is discovered should display the letter that is on that location. In addition, the sides of the board should be labeled from 1 to $\sqrt{size}$ to help the user identify which locations they want opened next.

You may assume that the player will follow your instructions and will input integers when asked for location (rather than doubles or other type), but your program should test if the player entered integers in the required range and prompt her to repeat the entry until correct input is obtained. You should also fully test your program – for example, what does your program do if the player entered two locations that are already discovered, etc.

To summarize, as part of this assignment, you are provided with a file called a4.java. All your code must go inside this file in the clearly indicated spaces (and only in those spaces). You must not change any part of the code that is provided. In particular, the code for the main method is provided to you in its entirety. You cannot add nor delete anything inside of the main method.

Your task is to complete the program by completing the shuffleBoard method and playGame method. The shuffleBoard method takes as input the board (i.e., 2D array) and shuffles it as described above. The playGame method takes as input (the shuffled) board and plays the whole game.

Here is what a run of your program should look like. Study the below example carefully to understand what your program should do. Among other things, you will see that your program will need to "clear screen". It is ok to implement that by, for example, simply printing 30 or so new lines. To have your program wait for the player to press enter to continue, you should invoke (i.e., call) the waitForPlayer method. This method waitForPlayer is provided for you inside the a4.java file.

While your task is to program the shuffleBoard and playGame method only, you may, if you like, add extra methods of your own. For example, you may develop one method to print the current board, and another method to print the current board with two new locations (given by the user) revealed. These methods would be called/invoked from the playGame method.

Here is what a run of your program should look like:

```
Your program:
Welcome to Memory Game
For 2x2 board game press 2
For 4x4 board game press 4
For 6x6 board game press 6

Player: 1

Your program
Wrong input

For 2x2 board game press 2
For 4x4 board game press 4
For 6x6 board game press 6

Player: 2

Your program:

1 * *
2 * *
  1 2

Enter a pair of undiscovered distinct locations on the board that you want revealed.
i.e., a pair of integers in the range [1, 2]

Enter the first location

Player: 1 1
Your program: Enter the second location
Player: 2 1

Your program:

1 A *
2 B *
```

```
  1 2

Press enter to continue

Player: presses enter

Your program:  clears the screen and prints

1 * *
2 * *
  1 2

Enter a pair of undiscovered distinct locations on the board that you want revealed.
i.e., a pair of integers in the range [1, 2]

Enter the first location
Player: 1 1
Your program: Enter the second location
Player: 1 2

Your program:

1 A C
2 * *
  1 2

Press enter to continue

Player: presses enter

Your program:  clears the screen and prints

1 * *
2 * *
  1 2

Enter a pair of undiscovered distinct locations on the board that you want revealed.
i.e., a pair of integers in the range [1, 2]

Enter the first location
Player: 1 1
Your program: Enter the second location
Player: 2 2

1 A *
2 * A
  1 2

Press enter to continue

Your program:  clears the screen and prints

1 A *
2 * A
  1 2

Enter a pair of undiscovered distinct locations on the board that you want revealed.
i.e., a pair of integers in the range [1, 2]

Enter the first location
Player: 1 3

Your program:
The location is invalid. It is outside of the board.
```

```
Enter the first location
Player: 2 2

Your program:
The location is invalid. It is already discovered.

Enter the first location
Player: 1 2

Enter the second location
Player: 1 1

Your program:
The location is invalid. It is already discovered.

Enter the second location
Player: 1 2

Your program:
The location is invalid. The second location equal to the first.

Enter the second location
Player: 2 1

1 A B
2 B A
  1 2

Press enter to continue

Your program:  clears the screen and prints

1 A B
2 B A
  1 2

Congratulations! You completed the game.
```

```
public class Shuffle {
  public static void main(String[] args) {

 int[] deck = new int[52]; // 52 stand for 52 cards in the deck of cards

    // Initialize the array to numbers for 1 to 52
    for (int i = 0; i < deck.length; i++){
      deck[i] = i;
}

    // Shuffle the array
    for (int i = 0; i < deck.length; i++) {
      // Generate an index randomly
      int index = (int)(Math.random() * deck.length);
      int temp = deck[i];
      deck[i] = deck[index];
      deck[index] = temp;
    }
  }
}
```