CSC 502 PROJECT 3
CARSENTRATION, PART 1

PROFESSOR GODFREY C. MUGANDA
DEPT OF COMPUTER SCIENCE

1. Project Overview

We are going to write a version of the concentration game in two projects. In this first project, we are going to create the user interface and write a shell for a mouse-click handler.

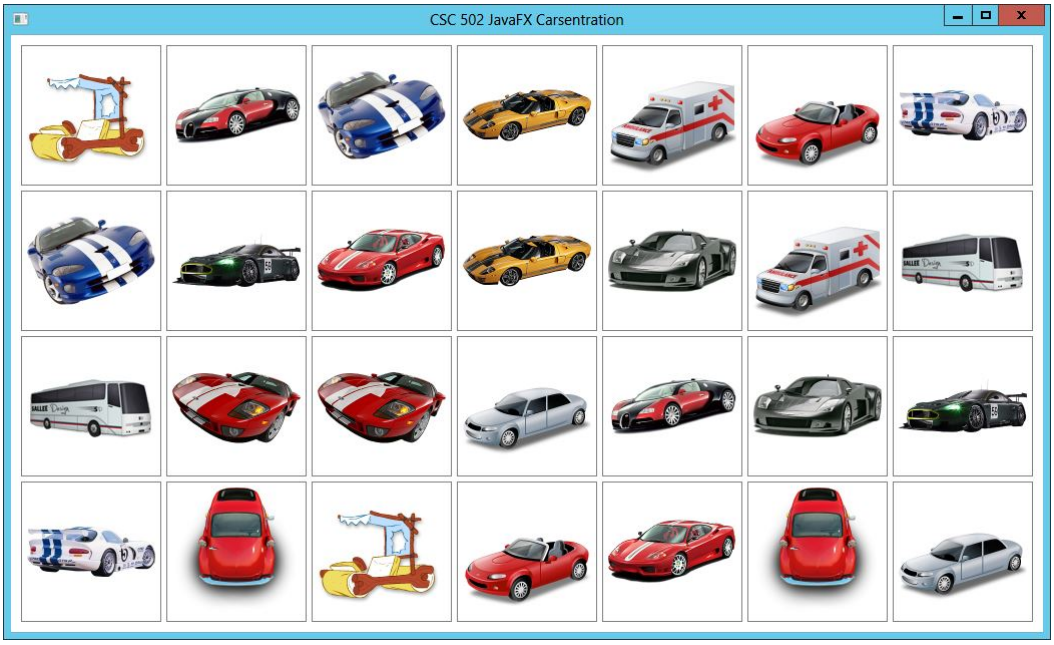
The game will involve the display of 14 cars in duplicate, (each car will be displayed twice) in a 4×7 grid. Each run of the program will display the same cars, but in random order.

A zipped folder of images of cars will be provided to you by the instructor. There will be 28 images in the folder, but you only need to use any 14 of them. The image files will be named

`car0.png  car1.png, ... , car27.png`

Here is a sample screen shot of a run of the program:

Here is another sample screen shot, from a separate run:



2. How to proceed

Create an array of 14 `Image` objects, and initialize the array so that kth image is the image from file `car"k".png`. For example, the image in position 5 of the array will come from the file `car5.img`.

You will want to create a list of the 28 numbers

$$0, 0, 1, 1, 2, 2, \ldots, 13, 13$$

(You can use an `ArrayList<Integer>` class). Use the `Collections.shuffle()` method to randomize the order. Finally, display the images in a 4×7 grid pane, by rows and then by columns, in the order that the indices appear in the shuffled list.

For example, if the shuffled list is

```
3, 8, 1, 2, 9, 12, 5, 4, 4, 1, 7, 6, 10, 10, 9, 13, 0, 3, 5, 7, 13, 2,
12, 6, 0, 8, 11, 11
```

then the sequence of images in the grid pane will be

```
3   8  1  2  9  12   5
4   4  1  7  6  10  10
9  13  0  3  5   7  13
2  12  6  0  8  11  11
```

Instead of using `ImageView` as is, create a subclass of `ImageView` that takes a constructor parameter k that is an index of an image, and sets the image in the image view to be the kth image in the array of images. Your subclass should have a

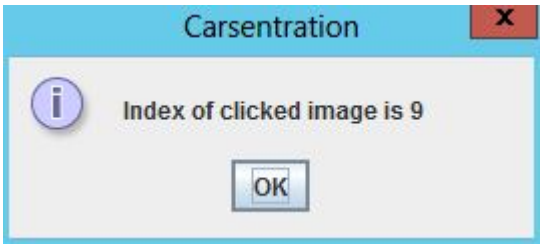
```
int getImageIndex()
```

method that will return the index of the image in the image view.

Finally, create a `MouseEvent` handler that is capable of handling mouse clicks. You will need a class that implements the

`EventHandler<MouseEvent>`

interface (you can also use a lambda expression). For this project, the event handler will just pop up a `JOptionPane` that uses the `showMessageDialog()` method to display the image index of the clicked image view.



This can be done with a code similar to this:

```
JOptionPane.showMessageDialog
                        (null,
                         "Index of clicked image is " + imView.getImageIndex(),
                         "Carsentration",
                         JOptionPane.INFORMATION_MESSAGE );
```

3. Due Date

This is due Monday of Week 6.