CSC 415/515 PROJECT 3
JAVASCRIPT CONCENTRATION GAME

PROF. GODFREY MUGANDA
DEPT OF COMPUTER SCIENCE

1. Introduction

Using JavaScript, write a game that will help people work on their concentration and memory skills.

Think of a game played by two people, a person wanting to improve their memory and concentration skills, and a memory trainer. The trainer has a deck of memory cards. All cards have the same image of Figure 1 on one side (the back of the card). However, each card has a image of motor vehicle on its second side (the front face of the card). There are 28 cards altogether, but only 14 different images appear on the faces of the cards, with each image appearing of the front face of two different cards.

To begin, the trainer shuffles the 28 cards and lays them face-down down in a 4×7 grid as shown in Figure 2, which is a screen shot of what the game you write should like when play begins.

To play, the player points to a card, which the trainer then turns face up. The player then points to a different card, which the trainer also turns face up. If the two face-up cards are displaying the same image, the trainer removes them from the table and replaces both of them with cards that display a outline of a car (see Figure 3). The removed cards are said to have been *cleared*.

If, on the other hand, the two face-up cards display different images, the trainer turns them face down again. The player is now supposed to remember the location of each those two images in the grid. The player continues to select pairs of cards at a time to be turned face up. Each time, the trainer removes the two cards and replaces them with *cleared* cards if the two cards match, but turns them face-down if they do not match. The game ends when all cards have been cleared.

Figure 1. Card that hides an image

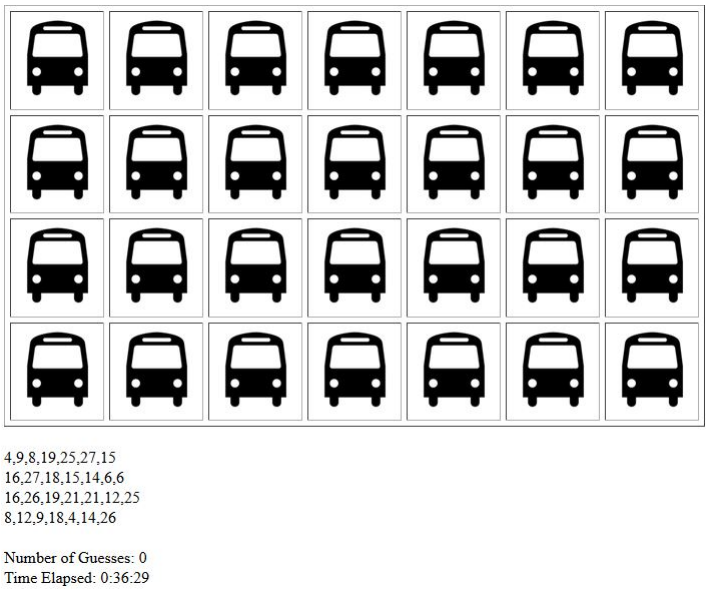

FIGURE 2. The Start of a Game

FIGURE 3. Card indicating a cleared image

2. Overall Design

A zipped-up folder of 28 images of cars will be provided to you. The names of the images files are `car0.png` through `car27.png`. There also two image files, `hidden.png` and `cleared.png`.

For your user interface, use a table with 4 rows of images, where each row holds seven images. This is shown below, with the contents of the three of the rows deleted to save space.

```
<!DOCTYPE html>
<html>
    <head>
        <title>JavaScript Concentration Game</title>
        <meta http-equiv="Content-Type"
             content="text/html; charset=UTF-8">
        <script type="text/javascript"
             src="Carsentration.js">
        </script>
    </head>
    <body>
        <table border="1" cellspacing="5" id="table">
            <tbody>
```

```
                <tr>
                    <td><img id="0" src="images/hidden.png"
                        width="90" height="90" /></td>
                    <td><img id="1" src="images/hidden.png"
                        width="90" height="90" /></td>
                    <td><img id="2" src="images/hidden.png"
                        width="90" height="90" /></td>
                    <td><img id="3" src="images/hidden.png"
                        width="90" height="90" /></td>
                    <td><img id="4" src="images/hidden.png"
                        width="90" height="90" /></td>
                    <td><img id="5" src="images/hidden.png"
                        width="90" height="90" /></td>
                    <td><img id="6" src="images/hidden.png"
                        width="90" height="90" /></td>
                </tr>

                <tr>
                </tr>

                <tr>
                </tr>

                <tr>
                </tr>
            </tbody>
        </table>
        <br/>
        <span id ="cheat"></span>
        <br/>
        Number of Guesses: <span id="numberOfGuesses">0</span>
        <br/>
        Time Elapsed: <span id ="timeElapsed">0:0:0</span>
        <br/>
    </body>
</html>
```

Notice that the `id` attributes of the image elements are the numbers $0 \ldots 17$. This will allow you to use loops to work with the images.

Notice also, the three span elements at the bottom of the HTML page. These are used to display a cheat sheet (revealing the indices of the images hidden in each cell of the table); the current number of guesses, and the amount of time elapsed since the start of the game. The time displays in hours, minutes and seconds.

3. How the game should work

The behavior of the game is best understood by viewing it as being in one of three states at any given time: 0 car images showing, 1 car image showing, and 2 car images showing. Let us call these states 0, 1, and 2. Initially, the game is in state 0.

The game responds to mouse clicks on the image cells as follows:

(1) If the game is in state 0, the clicked image is shown, and the game transitions to state 1.
(2) if the game is in state 1, then one image is already showing. If the clicked image is the same as the one showing, nothing happens. if the clicked image is already cleared, nothing happens. Otherwise, the clicked image is shown, the game sets a time out interval of 2 seconds to allow the user time to memorize the positions of the exposed images, and transitions to state 2.
(3) if the game is in state 2, nothing happens.

The game responds to a time-out event as follows. First a check is made to see whether the two exposed cells are displaying the same image. If so, the two exposed cells are cleared and the game transitions to state 0. Otherwise, the two exposes cells are covered up again, and the game transitions to state 0.

Of course, the game should keep track of the number of cleared cells. The game ends when all cells have been cleared. At that point, a JavaScript alert box pops congratulating the user on successful completion.

4. Implementation Hints

You will need a number of functions, as well as "global" variables accessible to the different functions. The global variables would keep track of number of cells already cleared, the state of the game, the index of the cells whose images are currently showing, etc.

A `reset()` function will be useful to initialize and reinitialize the global variables to start and restart the game if the user wants to play again. (Asking the user if they want to play again at the end of the game is optional).

A `handleClick(ImageId)` function is useful. You can call this to process a click on a particular image.

You can put an event listener for click events on the images on the table, and set it the event listener to handle click events on the images in the capture face.

You could also have a `timeElapsedFunction` to update the time elapsed.

Part of the code will look like this:

```
/**
 * Set up event handlers and call reset
 * @returns {undefined}
 */
function init()
{
   var tables = document.getElementsByTagName("table") ;

   tables[0].addEventListener("click",
               function(event)
               {
                  var clickedImageId =  event.target.getAttribute("id");
                  handleClick(clickedImageId);
               } );
   reset();
}
```

```
window.onload = init;
window.setInterval(timeElapsedFunction, 1000);
```

The `init()` function is set as the event handler for the *load* event on the browser window. That event fires as soon the HTML document is done loading into the browser window. The `init()` function will then set the click event listener of the table, and the game will be ready to go.

Additional hints will be given in lecture on Monday of Week 7, if needed.

5. For JavaScript Newbies

If you have never programmed in JavaScript, be sure to read up on array, and also on how to use the Random object in JavaScript to shuffle numbers

6. Due Date

This is due Monday of Week 8.