Memory Game Project

**Project Objective:**

To create a project that will be a memory game as explained in the slides and demoed in class.

**Project Specs:**

View:

1. You will create a board that looks similar to that in the slide.
2. 4X5 view of buttons (4 columns and 5 rows) al lot like the calculator
3. You will have a label that keeps track of moves left
4. You will have a label that keeps track of moves made
5. You will have a hidden label that will display if the user wins
6. The view will have an image for the background, to simulate the cards back and use Special Characters for the items that are revealed. When the Characters are revealed, the background of the button is set to blank.
7. The user cannot have more than two buttons revealing their special character. When a match occurs, the revealed matches are then allowed to remain revealed for the rest of the game.

The View will look like the following (Of course you can use your own images)

Controller:

1. The controller will only work with either the view or the model. None of the game play should be in the controller (i.e. 2D array of the board).
2. The controller will have one action for all the buttons,
3. The Controller will have an outlet for the labels
4. The controller can have other functions for flipping the image and Emoji symbols
5. You may also use other variables to keep track of previously press buttons like an array. (knowledge of the previously pressed button is important to check for a win).

Model:

1. Will contain a struct of type node:
   1. Will have a string for the emoji symbol
   2. Will have a Boolean type that indicate if it has matched.
2. Will have a class for the gameplay
   1. Will have a 2D array of type Node (used to contain the random emoji symbols for the board)
   2. Will randomly initialize the board with emoji symbols.
   3. Will have a row of the board that will help build it
   4. Should have a search function that uniquely identifies at most 2 of the same items in the 2D Array.
   5. Should have a print function in aiding in debugging your class.
   6. Must have a function that returns an emoji, given a tag number i.e.

getEmoji(tag)->emoji

* 1. Should have a function that translates a tag number to a row and column number
  2. Must have a function that will indicate if two items, given their tag numbers, match. This function may look like

IsMathc(tag1, tag2)->Boolean

* + 1. Checks to see if the two items match
    2. If they match, set their values in the struct to true and return true
    3. If they do not match return false
  1. Must have a function that can check the board for a win.
     1. A win is if all the nodes are true
     2. This function will return true if they are all true
     3. This function will return false if they are not all true

Extra credit:

1. Extra credit if you use the timer object to automatically flip the button
2. If you provide a popup menu to change the difficulty and to reset the game.
3. Save the highest score.