I designed a memory game where players are required to remember a random sequence of colours and subsequently repeat it to win points. The primary motivation for picking a game amongst many other possible forms of presentation was due to a failure in a game assignment given to us earlier in the school term. During that assignment I faced issues trying to stop the counter from increasing counts even when if the player did not click on any button. It led to my decision to take on a gaming programme for the final assignment to challenge myself (which I succeeded in!). In addition, other secondary reasons why I selected this was because it encompasses almost every skill I have learnt throughout this module, such as the use of loops, arrays, timeout functions, styling, attributes, adding music, conditionals, counters and even math RandInt. In short, this assignment allowed me to combine content I have learnt from every single lesson in this module.

The essence of this game is such that the boxes are arranged in an array and are initialised to randomly turn black in sequence. There are several levels to this game to increase the challenge when programming the code. But on a lighter note, a specific sound file was added to spike the tension levels for the user. The music component was most effortless given the very guided practice we already went through in tutorial.

I started off coding with the existing code I had for my clicking Game assignment earlier in the term. While I had to remove most parts of it due to irrelevance to my current project, it provided me with a backbone and layout to start with, such as creating a new Raphael paper and setting the background of my game with an image. I picked a wooden floor themed as my background because the imagery of wood helps to ease the tension of the game with its soothing colour that allows for my brightly-coloured boxes to stand out.

Then I began to create my counter and buttons. Perhaps the most challenging aspect for me due to the freshness of this concept, I took a lot of effort to set up an array of buttons, each with their unique set of colour attributes. These are buttons that will appear as boxes that the player will click on during the game, so I deliberately styled them to be vibrantly coloured to add dynamism to the game. These buttons are all looped as it is a more efficient way to run the code repeatedly with a different sequence. I am starting to appreciate the usefulness of loops and arrays.

I decided to utilise the innerHTML property for my points system which I was always reluctant to use because it was mandatory to return the HTML content of the element. As it was a property hardly used during tutorials and assignments, I referred to http://www.w3schools.com/jsref/prop_html_innerhtml.asp for guidance.

Perhaps the most number of conditionals I have ever used in a single assignment, my game required heavy usage of “if” and “else if” statements for the points system. This was necessary to increase the points if the right button/ sequence was selected. This was manageable because as opposed to “for” loops, “if” and “else if” are more in tandem with the English language and less a computer language.

The number of “if” and “else if” statements increased further after I decided to increase the number of levels of the game so it is more complex. Using a combination of arrays and conditionals, the number of buttons appearing will increase as the level of the game increases!

Interestingly, I adopted a new function called the EventTarget.removeEventListener() which was easily understandable as it is the opposite of EventTarget.getEventListener() we learnt in the module. It basically removes an event type previously registered and I picked it up from https://developer.mozilla.org/en-US/docs/Web/API/EventTarget/removeEventListener. For example, I used it to remove the wrong button function in order for the right button to appear. I found this an extremely effective and easy way to remove my events temporarily.

After browsing through the aforementioned site for more guidance and inspiration, I also decided to adopt the parseInt() function that parses a string argument (in this case, total number of correct clicks) and returns an integer (increasing the score for right clicks). I sought help from some of my friends and they recommended this as a most efficient way to produce the effect I wanted.

Similarly, it is also the parseInt() function that I felt can be further improved in this game. I really had no idea how to produce the effect I wanted and hence had to seek advice from several friends and even to this point I believe there might be an alternative set of functions I can initialise to reach that outcome.

Tiong Jia En
A0129564X