Sudoku Assignment (Problem Solving for Computer Science)

Task 9

The original problem with the function ***'entriesToDel'*** was that it did not produce ***'n'*** (number) of coordinates as the coordinates would not all necessarily be unique. The for loop was traversing the array and returning random coordinates within the array, however, it had no way to check if it was outputting the same coordinates which led to coordinates being duplicated, which led to a number of inconsistent blank spaces.

The way I fixed the problem was to implement a method that checked to see if the coordinate had already been selected and if so to continue to search for a coordinate that was unique. I done this by creating two variables, ***'var runCheck'*** & ***'var duplication'***. The variable ***'runCheck'*** was an array that consisted of the random coordinates produced by ***'Math.round(3*Math.random())'*.** The other variable, ***'var duplication'*** was a Boolean (true/false) to check to see if there was any duplication within the array of coordinates. I used an embedded for loop to traverse both arrays (***'var array'*** & ***'runCheck'***) and an if statement to check if the arrays produced the same coordinates. The if statement conditions checked to see if the ***'var array'*** coordinates were the same as the ones found in ***'var runCheck'***. If the conditions were met, then ***'var duplication'*** would be set to 'TRUE' which exits the loop and runs another if statement that changes the original for loop to decrement opposed to increment until the conditions of the if statement are not met (***'n'*** amount of coordinates have been produced without any duplicates). Finally, once the conditions of the if statements are no longer met, the ***'runCheck'*** array is pushed into the original array before it is returned at the end of the function.

Task 10

In this assignment we used a Linear Search Algorithm. A linear search algorithm is a method for finding an element within a list, starting at the beginning of the data set, each item of data is examined until a match is made.

As the algorithm used in this assignment is a linear search algorithm, there are couple of limitations to this type of algorithm. Linear Search Algorithms are very slow in comparison to other types of algorithms, such as Binary Search Algorithms. For what we aim to achieve (creating a sudoku puzzle) this is not an issue as the data sets are small. However, if we needed to scale up the data set or use an un-sorted data set then the linear search algorithm would not be an efficient algorithm to use.

Sudokus can be generated and solved in a plethora of methods. An alternative algorithm that could have been used is a binary algorithm as it can handle large data sets and allows for being scaled up as it can efficiently manage large data sets.

A Backtracking algorithm (a type of brute force search) may have also been an interesting alternative as this algorithm works by trying a number and removing it (backtrack) if it doesn't work and trying the next number. This is better than generating all possible combinations of numbers and then trying every combination one by one, as it changes whenever it backtracks. A Backtrack Algorithm may have been a way to resolve the issue with the function *'entriesToDel'.* A random based algorithm (stochastic), such as genetic algorithm, is a way to solve and generate puzzles, however,

an algorithm that relies on randomness rather than pattern or reason isn't necessarily going to be more efficient.

Finally, from conducting some research I have come across an report that explains how a greedy algorithm can effectively generate sudoku puzzles. A greedy algorithm is an algorithm that makes the best choice at each small stage with the goal of this eventually leading to a overall solution. This means that the algorithm will always pick the best solution without considering consequence. As we are generating a puzzle, there is no consequence to the algorithm failing. The idea behind using this algorithm to generate sudoku puzzles can be found in an online report, 'A New Algorithm for Generating Unique-solution Sudoku'

Further Reading on Sudoku Generating Algorithms

In this paper we present a brand-new algorithm for generating unique-solution Sudoku puzzles. Its complexity is practicable, and it is very easy to expand. (Sun, Sun, Wu, Yin, & Yang, 2008)

https://www.researchgate.net/publication/251863893_A_New_Algorithm_for_Generating_Unique-Solution_Sudoku