***Sorting Algorithms and their complexity***

Different sorting algorithms have different time complexities, and in this document we look at what the complexities of 5 sorting algorithms are.

# Insertion Sort

Insertion sort, a simpler sorting algorithm, uses two for loops to iterate through the algorithm and as such the worst case scenario would be a quadratic complexity, O(n^2)

# Bubble Sort

Bubble sort, perhaps the most famous sorting algorithm out there, is similar to Insertion sort in the sense that it uses two repetitive loops. Henceforth, in a worst-case scenario the complexity would be quadratic,O(n^2). However, in non-worst case scenarios, the optimised variant of the algorithm can be faster then other quadratic sorts in special cases such as almost sorted lists.

# Merge Sort

Merge sort is a divide and conquer algorithm, which is an algorithm that continously breaks down a list(or other iterable objects) into sublists that brought back together sorted. Because of this, the sorting algorihm only checks some of the variables one with another, and so it has the logarithmic complexity, O(nlogn)

# QuickSort

Another divide and conquer algorithm, Quicksort splits an list into multiple sublists that are then brought back together. In most cases, Quicksort has a logarithmic complexity like Merge sort, ***however*** in a worst-case scenario(which is pretty uncommon) it would have a quadratic complexity O(n^2). This is because in a case in which, for example, the array is already sorted and it would just create a sublist for one element, and another one for the rest. This could be fixed by designating the pivot as the median element, but that would necessitate additional code.

# Selection Sort

Selection sort is another algorithm that uses two repetitive loops that makes it a quadratic algorithm 0(n^2)