Big O Notation (9618)

- Is a mathematical notation used to describe the performance or complexity of an algorithm in relation to the time taken or memory used for the task.

Big O Order of Time Complexity

- O (1) : Describes an algorithm that always takes the same time to perform the task. e.g: an algorithm deciding if a number is +ve or -ve

- O (N) : Describes an algorithm where time increases linearly in relation to the number of items (N) e.g: linear search

- O (N^2) : Describes an algorithm where the time to perform the task will increase linearly to the square of number of items. (N) e.g: Bubble sort, insertion sort.

- $O(2^n)$: Describes an algorithm where the time to perform the tasks double every time the algorithm uses an extra item of data. e.g: calculation of fibonacci numbers using recursion

- $O(Log(N))$: Describes an algorithm where the time increases logarithmically. e.g: Binary searching

Big O Order of Space Complexity

- $O(1)$: Describes an algorithm that always uses the same space to perform the same task.

  e.g: any algorithm that just uses the variables, $d \leftarrow a+b+c$

- $O(N)$: Describes an algorithm where the space to perform the task increases linearly to the number of data items. e.g: Any algorithm that uses arrays