**What is Big O Notation?**

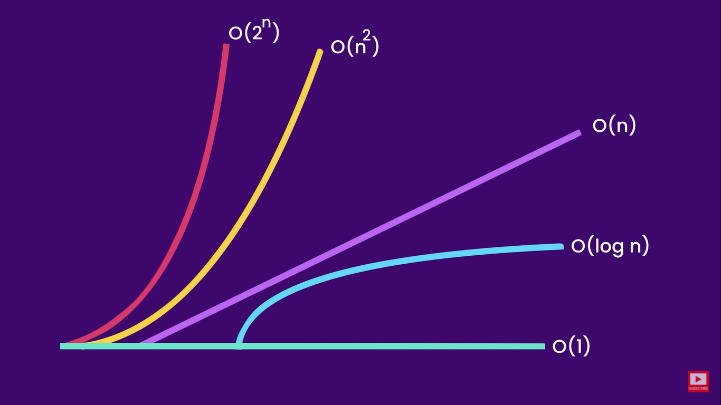
The big O notation is usually done to check the performance and scalability of an algorithm. We use this notation to check if the algorithm can hold the huge number of data or not. It is also known as the ***Big L***.

**What is Runtime Complexity?**

The runtime complexity of an algorithm is the time taken by the algorithm to execute in relation to the steps taken in the algorithm.

**Constant Complexity O(1)**

A constant task’s runtime will not change no matter whatever the input is. In these tasks, in most cases only single step is needed. For example, taking an array as an argument in a function, and the function prints it’s first index element, so no matter how long the array is, the algorithm has to print it’s [0] element.

**Linear Complexity O(n)**

This type of runtime complexity occurs when the length of the number of inputs varies. For example, if an algorithm traverses through an array, the size of the array can be 1 or 100, if the size is 1, the algorithm will take only 1 step, but if the size is 100, the algorithm will repeat 100 times, so the value of ***n*** grows linearly, this is why we call it linear complexity, denoted by O(n).

**Quadratic Complexity O(n2)**

In quadratic task, the number of steps is equal to the square of the input value(s). For example, we are using nested loops for accessing the elements of an array given in the user input, now it depends on the size of the array that how long it is, and we use nested loops for accessing them, as one loop is O(n), the nested will also be O(n), so we can write it as O(n2). The quadratic algorithm is less scalable than the linear algorithm.

**Logarithmic Complexity O(log n)**

The logarithmic approach allows you to divide the task in several useful parts and discard the useless items, this makes it work faster than the ordinary algorithms. This type of algorithm is also known as divide and conquer. The binary search is one of the best examples that use logarithmic approach.

**Exponential Complexity O(2n)**

The Exponential curve is the opposite of the logarithmic curve. This type of algorithm is much slower than the logarithmic algorithm. This type of algorithm has very less runtime scalability.

**The Space Complexity**

The space complexity also works on the big O notation but in this type of complexity we check how much space our algorithm is taking in the system. This is pretty much useful when we are dealing with the optimization of the application that we are developing.