Space & Time Complexity

NOTE : In some videos, you will hear about words like Space & Time Complexity of Algorithms, big Oh etc.
The concept of Space & Time Complexity is covered in the later part of the course(as an individual chapter).
To be able to understand it before that, please read this reading material.

SCENARIO

Imagine you give your lucky pen to one of your friends. Now your exams are approaching, so you would like your pen back.

SPACE COMPLEXITY

Space Complexity is represented as a function that portrays the amount of space necessary for an algorithm to run until complete.

In our scenario we are looking at you can think of space complexity as the number of rooms you need to figure out who has the pen. However, in computer science, this typically means how much memory does the processes and data structures in our codebase / functions take up to achieve their goal.

TIME COMPLEXITY

Time Complexity is represented as a function that portrays the amount of time necessary for an algorithm to run until complete.

In our scenario we are looking at time complexity that can be represented in the approach you take in finding out who has the pen. However, in computer science, this typically means how much time the processes and data structures in our codebase / functions take up to achieve their goal.

Time complexity however is an umbrella term for the different types of time complexities that we can calculate. From fastest to slowest they are:

Worst Case Time Complexity: The absolute most number of times an operation needs to be done before completed

Average Case Time Complexity: The average number of times it will take for the algorithm / code to complete

Amortized Running Time: Similar to average, it is the number of times the operation will take when run a sufficient amount of time consecutively

Best Case Time Complexity: The fastest number of times an operation needs to complete

NOTE - Read O(n) as Big Oh of n.

Example of Complexity Function

- $O(n^2)$: You ask one friend if they have the pen. You also ask them if the other 99 friends have the pen. Afterwards move on to the next friend and repeat the process.
- O(n): You ask each friend one by one if they have the pen.

- O(log n): You divide the friends into two rooms and ask if the person with the pen is in room 1 or room 2. Depending on which group has the pen, split them up into the two rooms and repeat until there is one person in the right room with the pen.
- O(1): You remember who has the pen and you go to them directly.

You Should Know : Something that is important to note is that the Time / Space Complexity of algorithm/code is not in fact actual time or space that is required to execute a particular code but the number of times a statement executes. Meaning the function is not a measure of time / or space but a measure of what the code is actually doing.

SUMMARY

There are multiple solutions that can be written for the same problem. We need to learn how to compare the performance of different algorithms and choose the best one to solve a particular problem. While analyzing an algorithm, we mostly consider **time** complexity and **space** complexity.

Time complexity is the time taken by the algorithm to execute each set of instructions. It is always better to select the most efficient algorithm when a simple problem can be solved with different methods.

Similarly, **Space complexity** of an algorithm is the amount of space or memory taken by an algorithm to run as a function of the length of the input.

Time and space complexity depends on lots of things like hardware, operating system, processors, etc. However, we don't consider any of these factors while analyzing the algorithm. We will only consider the **execution time** of an algorithm.