# Complexity Analysis

There two metrics to measure the complexity of the algorithm

* Time
  + How fast program runs
* Space
  + How much memory program uses

Complexity analysis made through asymptotic analysis. There are three notations:

* Big O notation
* Theta Notation
* Omega Notation

## Tree

### Tree traversal:

All kind of binary tree

<https://towardsdatascience.com/5-types-of-binary-tree-with-cool-illustrations-9b335c430254>

Tree Traversal is basically going through every node(key) exactly once.

* BFS
  + Is like a reading a book
  + Nodes at the same level will be traverses first
  + It’s also called Level order traversal
* DFS
  + Go to one side → complete it → Go to other side
  + Recursive nature
  + 3 types or permutation. Left is visited always before Right
    - Inorder: Left → Root → Right
    - Preorder: Root → Left → Right
    - Postorder: Left → Right → Root

## Binary Search Tree:

In order to satisfy BST property:

1. Every node in BST must have a value that is strictly greater than all values in its left nodes and must have a value that is strictly less than or equal to all the values of its right nodes.
2. Left nodes and Right nodes must be BSTs themselves.

Validating BST algorithm trick:

That mean every node must be inside a range tree.value should not be less than minimum value and tree value should not be greater than or equal to maximum tree value

## Dynamic Programming:

It is an optimization over recursion. Reuse the solution when there are overlapping subproblems.

Two way to implement:

1. Memoization (Top Down)
2. Tabulation (Bottom Up)

## Traversal

<https://towardsdatascience.com/4-types-of-tree-traversal-algorithms-d56328450846>

**Resources:**

<https://www.algoexpert.io/product>

<https://practice.geeksforgeeks.org/batch/dsa-4/>

<https://www.youtube.com/channel/UC9fDC_eBh9e_bogw87DbGKQ/videos>

<https://www.youtube.com/channel/UCRPMAqdtSgd0Ipeef7iFsKw>