Binary Search Tree

Binary Tree

A **binary tree** is made of nodes, where each node contains a left reference, a right reference, and a data element.

- The topmost node in the tree is called the **root**.
- Every node (excluding a root) in a tree is connected by a directed edge from exactly one other node. This node is called a **parent**.
- Each node can be connected to atmost two nodes, called **children**.
- Nodes with no children are called **leaves**, or **external nodes**.
- Nodes which are not leaves are called **internal nodes**.
- Nodes with the same parent are called **siblings**.

Binary Search Tree

A **binary search tree** is a rooted binary tree, whose internal nodes each store a key and each have two distinguished sub-trees, commonly denoted left and right. The tree additionally satisfies the **binary search tree property**, which states that the key in each node must be greater than all keys stored in the left sub-tree, and smaller than all keys in right sub-tree.

Program

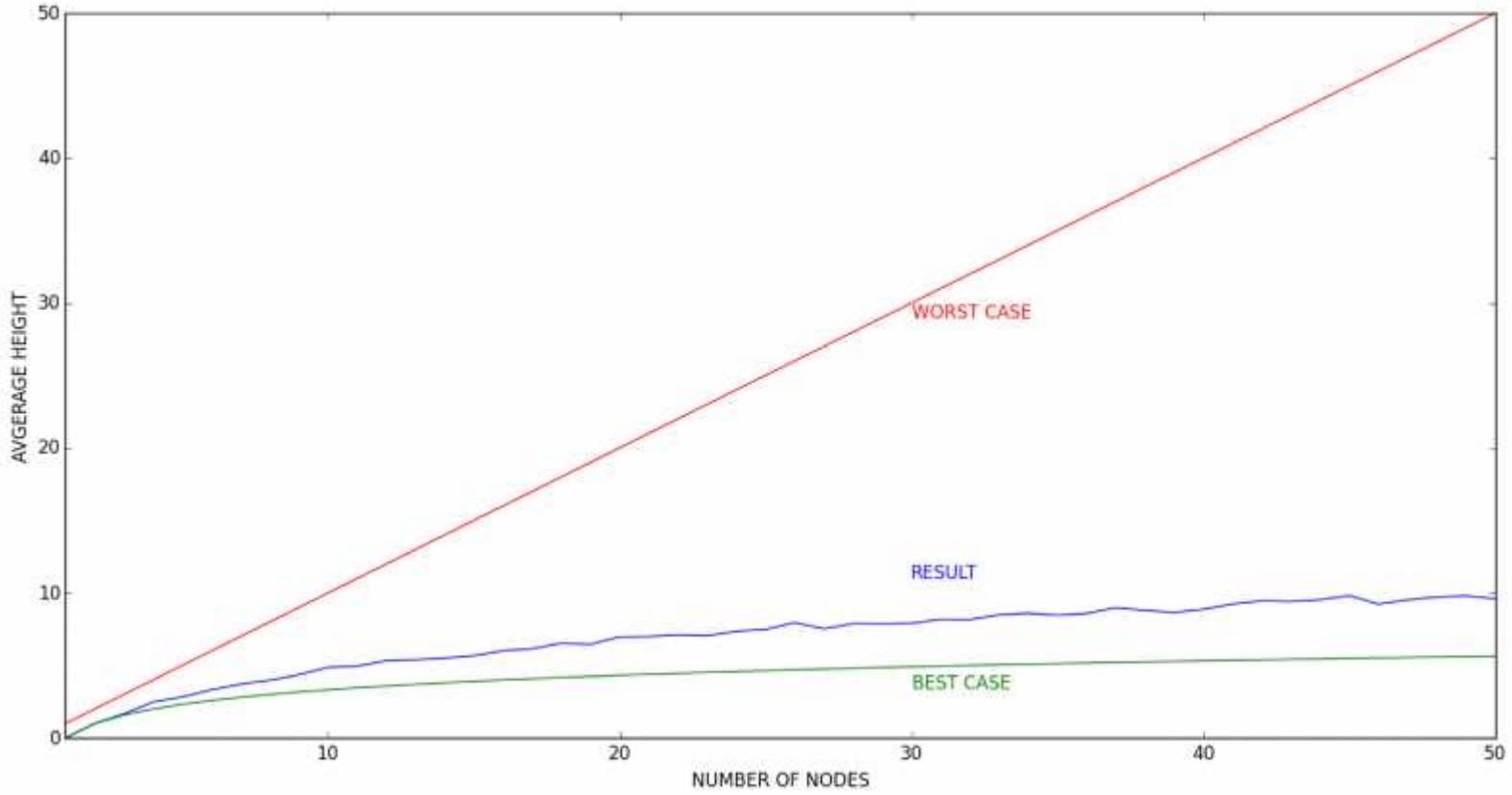
In my program I have created a class called Binary having

- **attributes**: val, right and left
- **methods**: add() and height().

The program generates 100 binary search trees having n nodes for each $1 \leq n \leq 50$ and calculates the average height of these binary trees.

I have used matplotlib module to plot the graph that indicates the result of the program along with the best and worst case of binary search tree.

Output


AVGERAGE HEIGHT
NUMBER OF NODES
WORST CASE
RESULT
BEST CASE
0
10
20
30
40
50