**Binary Tree [CO2]**

Instructions for students:

● Complete the following methods on Tree.

● You may use any language to complete the tasks.

● All your methods must be written in one single .java or .py or .pynb file. DO NOT CREATE separate files for each task.

● If you are using JAVA, you must include the main method as well which should test your other methods and print the outputs according to the tasks.

● If you are using PYTHON, then follow the coding templates shared in this folder.

NOTE:

**● YOU CANNOT USE ANY BUILT-IN FUNCTION EXCEPT** len **IN PYTHON. [negative indexing, append is prohibited]**

**● YOU HAVE TO MENTION SIZE OF ARRAY WHILE INITIALIZATION**

**● YOUR CODE SHOULD WORK FOR ALL RELEVANT SAMPLE INPUTS**

* **DO NOT USE LIST, QUEUE**

## Mirror Tree:

Given a binary tree, convert it into its mirror.

Sample Input:

10

/ \

20 30

/ \

40 60

Sample Output:

10 10

/ \ Mirror / \

20 30 —> 30 20

/ \ / \

40 60 60 40

Inorder Traversal of mirror: 30 10 60 20 40

## Level Min:

Given a binary tree, find the smallest value in each level.

Sample Input: [You can use a dictionary here]

4

/ \

9 2

/ \ \

3 -5 7

Sample Output: 4 2 -5

Explanation:

There are 3 levels in the tree

Level 0: {4}, min= 4

Level 1: {2, 9}, min= 2

Level 2: {7, 3, -5}, min = -5

## Inorder Predecessor:

Given a BST, and a reference to a Node x in the BST, find the Inorder Predecessor of the given node in the BST.

**DO NOT USE LIST**

Sample Input:

20

/ \

8 22

/ \

4 12

/ \

10 14

x = reference of the node containing 20

Sample Output: reference of the node 14

Explanation:

The inorder predecessor of a parent node is the largest (rightmost) node in the left subtree. The rightmost node in the left subtree of parent node 20 is 14.

Another explanation is that, the inorder traversal of the given tree:

4 8 10 12 14 20 22

Hence, the inorder successor of 20 is 14.

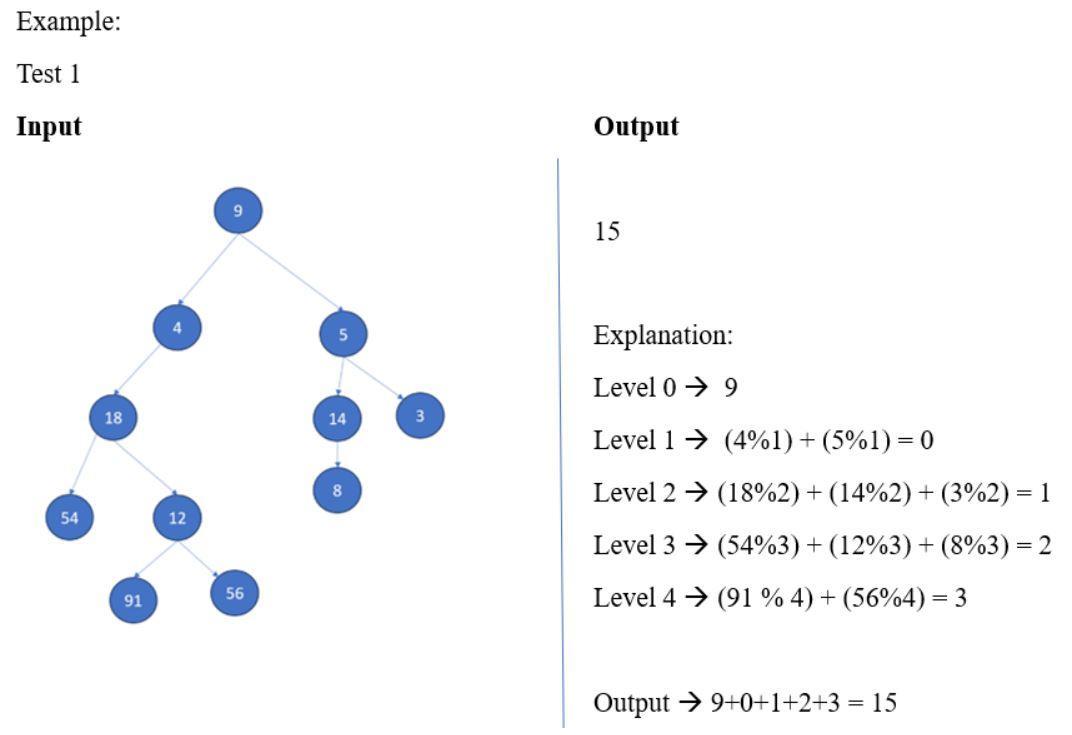
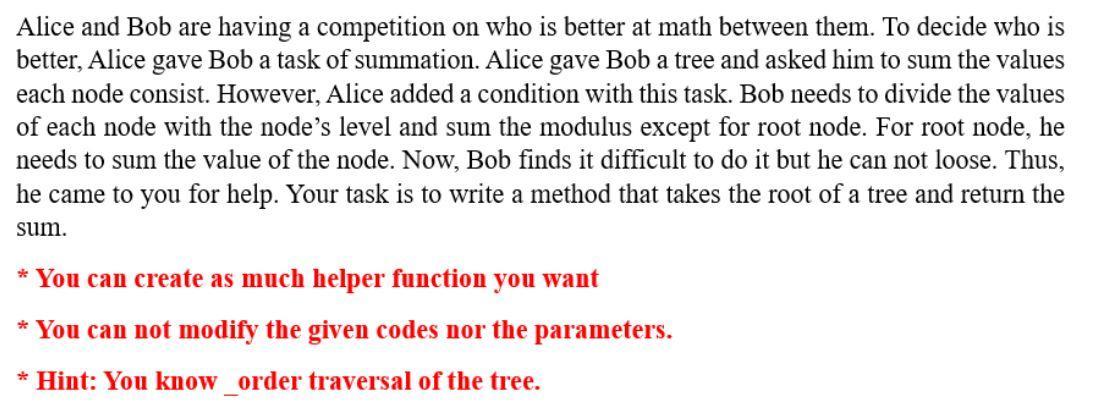
Sample Input 2:

x = reference of the node containing 8

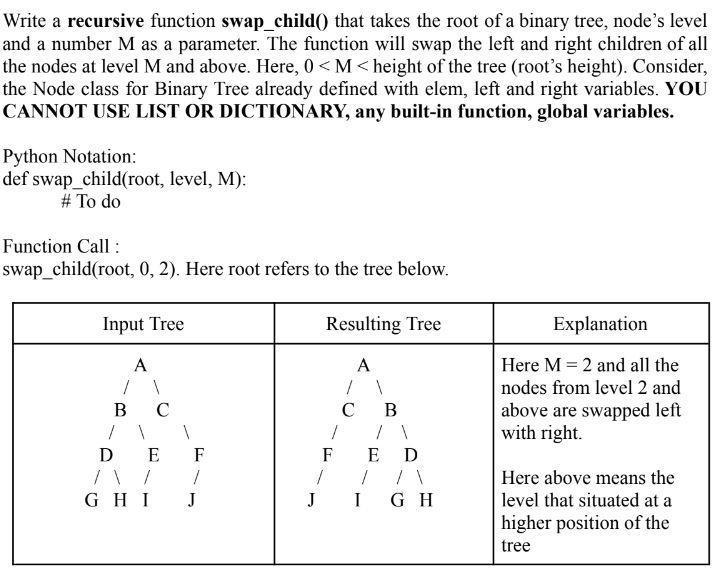
Sample Output: reference of the node 4

## 

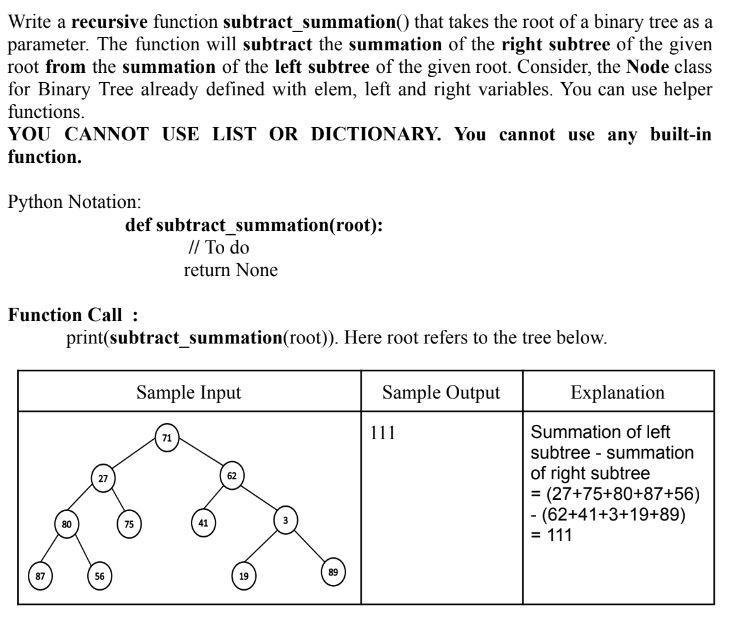
## Sum of Nodes:



1. Swap Children Nodes:



## Subtraction of Nodes:



## 

## 

## 

## Bonus Task: Difference of Level Sum

Given a Binary Tree, Write a function that finds the difference between sum of all nodes present at odd and even levels in a binary tree, i.e. sum of all odd level nodes - sum of all even level nodes.

| Sample Input: | Sample Output | Explanation |
| --- | --- | --- |
|  | 4 | -1+2+3-4-5-6+7+8 = 4 |