# Boundary Traversal of binary tree

Given a Binary Tree, find its Boundary Traversal. The traversal should be in the following order:

1. **Left boundary nodes:** defined as the path from the root to the left-most node ie- the leaf node you could reach when you always travel preferring the left subtree over the right subtree.
2. **Leaf nodes:** All the leaf nodes except for the ones that are part of left or right boundary.
3. **Reverse right boundary nodes:** defined as the path from the right-most node to the root. The right-most node is the leaf node you could reach when you always travel preferring the right subtree over the left subtree. Exclude the root from this as it was already included in the traversal of left boundary nodes.

**Note:** If the root doesn't have a left subtree or right subtree, then the root itself is the left or right boundary.

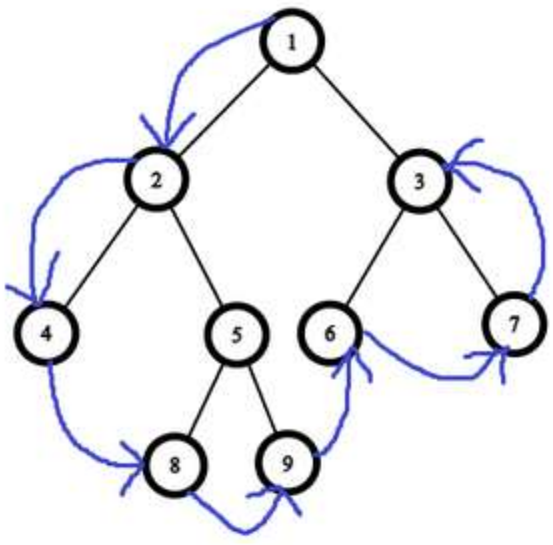
**Example 1:**

**Input:**

```
      1
    /  \
    2    3
   /\  /\
  4   5 6   7
    /\
    8   9
```

**Output:** 1 2 4 8 9 6 7 3

**Explanation:**



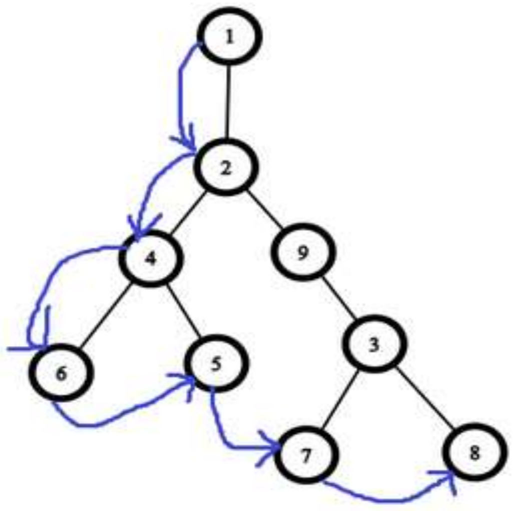
**Example 2:**

**Input:**

```
        1
       /
      2
     / \
    4   9
   / \   \
  6   5   3
         / \
        7   8
```

**Output:** 1 2 4 6 5 7 8

**Explanation:**



As you can see we have not taken the right subtree.

**Your Task:**
This is a function problem. You don't have to take input. Just complete the **function boundary()** that takes the root node as input and returns an array containing the boundary values in anti-clockwise.

**Expected Time Complexity:** O(N).
**Expected Auxiliary Space:** O(Height of the Tree).

**Constraints:**
1 ≤ Number of nodes ≤ $10^5$
1 ≤ Data of a node ≤ $10^5$