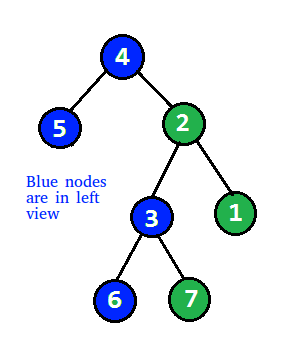
**Left View of Binary Tree:-**

Given a Binary Tree, print Left view of it. Left view of a Binary Tree is set of nodes visible when tree is visited from Left side. The task is to complete the function **leftView()**, which accepts root of the tree as argument.

Given a Binary Tree, print left view of it. Left view of a Binary Tree is set of nodes visible when tree is visited from left side.



Left view of above binary tree is: - 4 5 3 6

---------------------------------------------------------------------------------------------------

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

Left view of above tree is 1 2 4 8.

**-------------------------------------------------------------------------------------------------------------------------------**

**Example 1:**

**Input:**

  1

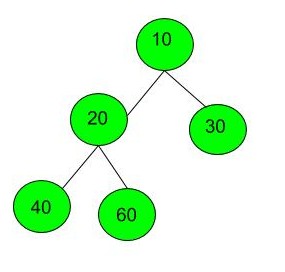
 /  \

3    2

**Output:** 1 3

**Example 2:**

**Input:**



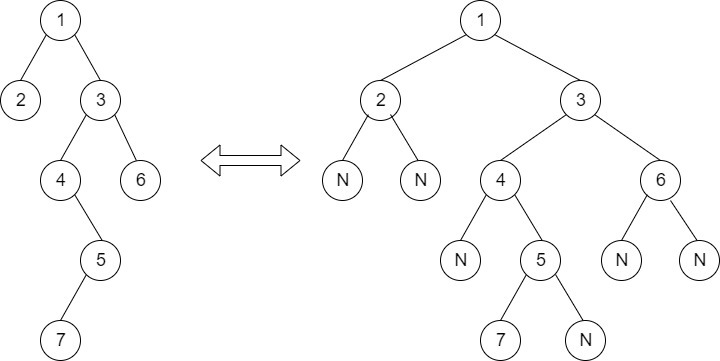
**Output:** 10 20 40

**Your Task:**  
You just have to **complete**the function **leftView()**that prints the left view. The newline is automatically appended by the driver code.  
**Expected Time Complexity:**O(N).  
**Expected Auxiliary Space:**O(Height of the Tree).

**Constraints:**  
1 <= Number of nodes <= 100  
1 <= Data of a node <= 1000

**Input Format:**

The tree in the input is given in the form of a string as described below.

1. The values in the string are in the order of **level order** traversal of the tree where, numbers denote node values, and a character “N” denotes **NULL** child.
2. For example:  
     
   For the above tree, the string will be: 1 2 3 N N 4 6 N 5 N N 7 N

Output will be: 1 2 4 5 7