PROBLEM

Serialize and deserialize a binary tree

Medium **Accuracy: 51.67%** **Submissions: 68K+** **Points: 4**

Done with winning Geekbits? Now win GfG Bag, GfG T-shirt & much more just by writing your experiences. Start Writing, Start Winning.

Serialization is to store a tree in an array so that it can be later restored and deserialization is reading tree back from the array. Complete the functions

- **serialize() :** stores the tree into an array **a** and returns the array.
- **deSerialize() :** deserializes the array to the tree and returns the root of the tree.

Note: Multiple nodes can have the same data and the node values are **always positive integers**. Your code will be correct if the tree returned by **deSerialize(serialize(input_tree))** is same as the input tree. Driver code will print the in-order traversal of the tree returned by deSerialize(serialize(input_tree)).

Example 1:

```
Input:
      1
    /   \
   2     3
Output:
2 1 3
```

Example 2:

```
Input:
         10
       /    \
      20     30
    /   \
   40   60
Output:
40 20 60 10 30
```

Your Task:
The task is to complete two functions **serialize** which takes the root node of the tree as input and stores the tree into an array and **deSerialize** which deserializes the array to the original tree and returns the root of it.

Expected Time Complexity: O(Number of nodes).
Expected Auxiliary Space: O(Number of nodes).

Constraints:
1 <= Number of nodes <= 10^4
1 <= Data of a node <= 10^9

CODE

```
#User function Template for python3



'''
class Node:
   def __init__(self,val):
      self.data = val
      self.left = None
      self.right = None
'''


class Solution:
   #Function to serialize a tree and return a list
containing nodes of tree.
   def serialize(self, root):
      # arr = []
      res = []
      q = []
      q.append(root)
      while(q):
         temp = q.pop(0)
         res.append(temp)
         if(temp.data == -1):
            continue
         if(temp.left):
            q.append(temp.left)
         else:
            q.append(Node(-1))
         if(temp.right):
            q.append(temp.right)
         else:
            q.append(Node(-1))
      # for i in res:
      #     print(i.data)
      return res
      #code here

   #Function to deserialize a list and construct the tree.
   def deSerialize(self, a):
      if(len(a)==0):
         return None
```

```
else:
   root = a[0]
   j = 0
   i = 1
   while(i<len(a)):
      if(a[j].data==-1):
         j+=1
         continue
      if(a[i].data!=-1):
         a[j].left = a[i]
      else:
         a[j].left = None
      i+=1
      if(a[i].data!=-1):
         a[j].right = a[i]
      else:
         a[j].right = None
      i+=1
      j+=1
   return root
#code here
```