이진 트리를 배열로 직렬화하고, 반대로 역직렬화하는 기능을 구현하라. 즉 다음과 같은 트리는 [1, 2, 3, null, null, 4, 5]형태로 직렬화할 수 있을 것이다.

Input: root = [1,2,3,null,null,4,5]
Output: [1,2,3,null,null,4,5]

```
# Definition for a binary tree node.
# class TreeNode(object):
#     def __init__(self, x):
#         self.val = x
#         self.left = None
#         self.right = None
```

1.bfs

```
import collections

class Codec:
    def serialize(self, root):
        """Encodes a tree to a single string.

        :type root: TreeNode
        :rtype: str
        """
        if root == None:
            return ''
        result = []
        q = collections.deque([root])
        while any(q):
                for _ in range(len(q)):
                    node = q.popleft()
                    if node == None:
                        node = TreeNode('!')
                    else:
                        q.append(node.left)
                        q.append(node.right)
                    result.append(str(node.val))
        return ' '.join(result)


    def deserialize(self, data):
        """Decodes your encoded data to tree.

        :type data: str
        :rtype: TreeNode
        """
        if data == '':
            return None
        data = collections.deque(data.split())
        q = collections.deque([TreeNode(int(data.popleft()))])
        result = TreeNode(0, q[0], None)
```

```
        while any([True for d in data if d != '!']):
            node = q.popleft()
            left = data.popleft()
            if left != '!':
                node.left = TreeNode(int(left))
                q.append(node.left)
            else:
                node.left = None
            right = data.popleft()
            if right != '!':
                node.right = TreeNode(int(right))
                q.append(node.right)
            else:
                node.right = None
        return result.left

import collections

class Codec:
    def serialize(self, root: TreeNode) -> str:
        queue = collections.deque([root])
        result = ['#']
        while queue:
            node = queue.popleft()
            if node:
                queue.append(node.left)
                queue.append(node.right)

                result.append(str(node.val))
            else:
                result.append('#')
        return ' '.join(result)

    def deserialize(self, data: str) -> TreeNode:
        if data == '# #':
            return None
        nodes = data.split()
        root = TreeNode(int(nodes[1]))
        queue = collections.deque([root])
        index = 2
        while queue:
            node = queue.popleft()
            if nodes[index] is not '#':
                node.left = TreeNode(int(nodes[index]))
                queue.append(node.left)
            index += 1

            if nodes[index] is not '#':
                node.right = TreeNode(int(nodes[index]))
                queue.append(node.right)
```

```
    index += 1
return root
```