Serialize and Deserialize Binary Tree

Try to solve the Serialize and Deserialize Binary Tree problem.

Statement

Serialize a given binary tree to a file and deserialize it back to a tree. Make sure that the original and the deserialized trees are identical.

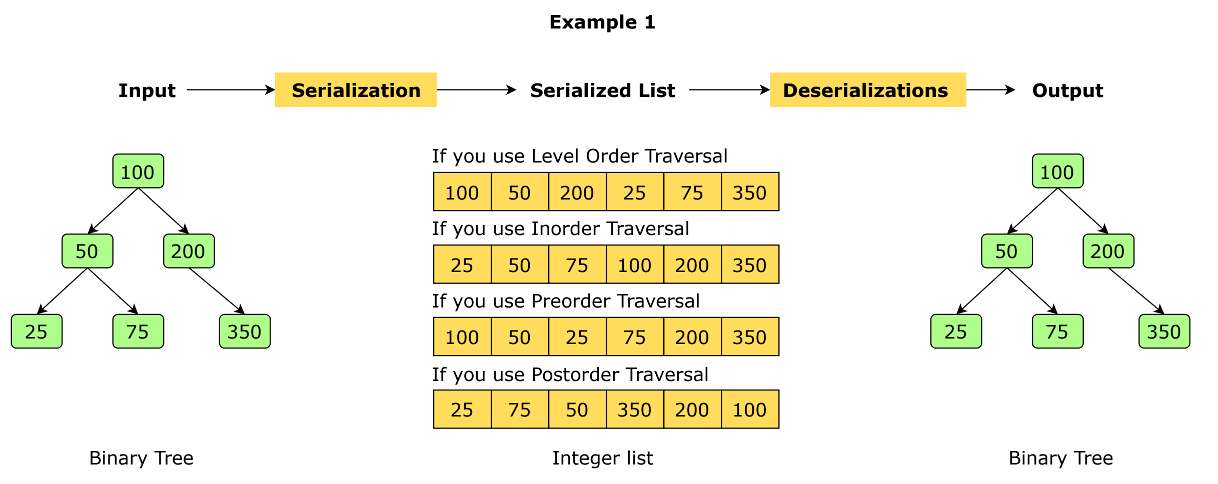
- **Serialize:** Write the tree to a file.
- **Deserialize:** Read from a file and reconstruct the tree in memory.

Serialize the tree into a list of integers, and then, deserialize it back from the list to a tree. For simplicity's sake, there's no need to write the list to the files.

Constraints:

- The number of nodes in the tree is in the range $[0, 10^4]$.
- $-1000 \leq$ `Node.value` ≤ 1000

Examples



If you use Level Order Traversal

100	50	200	25	75	350

If you use Inorder Traversal

25	50	75	100	200	350

If you use Preorder Traversal

100	50	25	75	200	350

If you use Postorder Traversal

25	75	50	350	200	100

1 of 2

Understand the problem

Let’s take a moment to make sure you’ve correctly understood the problem. The quiz below helps you check if you’re solving the correct problem:

Serialize and Deserialize Binary Tree

1

What is the serialized output of this tree using level order traversal?

```
     __ 350
    |
 __ 75 ______
|             |
25 _     ___ 200
    |   |
    50  100
```

A) [350, 75, 25, 50, 200, 100]

B) [350, 75, 25, 200, 50, 100]

C) [25, 50, 75, 100, 200, 350]

D) [50, 25, 100, 200, 75, 350]

Submit Answer

Question 1 of 2
0 attempted

Reset Quiz

Figure it out!

We have a game for you to play. Rearrange the logical building blocks to develop a clearer understanding of how to solve this problem.

Drag and drop the cards to re-arrange them in the correct sequence

Perform a depth-first traversal and serialize individual nodes to the stream.

Also, serialize a marker to represent a NULL pointer that helps deserialize the tree.

Deserialize the tree using preorder traversal.

During deserialization, create a new node for every non-marker node using preorder traversal.

Reset Show Solution Submit

Try it yourself

Implement your solution in `SerializeDeserialize.java` in the following coding playground.

Java

usercode > SerializeDeserialize.java

```java
// Definiton of a binary tree node class
// class TreeNode<T> {
//     T data;
//     TreeNode<T> left;
//     TreeNode<T> right;

//     TreeNode(T data) {
//         this.data = data;
```

```java
// }

import java.util.*;
import ds_v1.BinaryTree.TreeNode;

public class SerializeDeserialize{
    public static List<String> serialize(TreeNode<Integer> root) {

        // Replace this placeholder return statement with your code
        List<String> stream = new ArrayList<>();
        return stream;
    }
    public static TreeNode<Integer> deserialize(List<String> stream){

        // Replace this placeholder return statement with your code
        TreeNode<Integer> n = new TreeNode<Integer>(0);
        return n;
```

Powered by AI Submit

Test Cases Results

Case 1 Case 2 Case 3

Input #1

[100,50,200,25,75,350]

Serialize and Deserialize Binary Tree

Back Next

Solution: Diameter of ...

Solution: Serialize and...

☑ Mark as Completed