

# Problem 589: N-ary Tree Preorder Traversal

## Problem Information

**Difficulty:** Easy

**Acceptance Rate:** 76.50%

**Paid Only:** No

**Tags:** Stack, Tree, Depth-First Search

## Problem Description

Given the `root` of an n-ary tree, return \_the preorder traversal of its nodes ' values\_.

Nary-Tree input serialization is represented in their level order traversal. Each group of children is separated by the null value (See examples)

**Example 1:**



**Input:** root = [1,null,3,2,4,null,5,6] **Output:** [1,3,5,6,2,4]

**Example 2:**



**Input:** root =  
[1,null,2,3,4,5,null,null,6,7,null,8,null,9,10,null,null,11,null,12,null,13,null,null,14] **Output:**  
[1,2,3,6,7,11,14,4,8,12,5,9,13,10]

**Constraints:**

\* The number of nodes in the tree is in the range `[0, 104]`. \* `0 <= Node.val <= 104` \* The height of the n-ary tree is less than or equal to `1000`.

**Follow up:** Recursive solution is trivial, could you do it iteratively?

## Code Snippets

### C++:

```
/*
// Definition for a Node.
class Node {
public:
    int val;
    vector<Node*> children;

    Node() {}

    Node(int _val) {
        val = _val;
    }

    Node(int _val, vector<Node*> _children) {
        val = _val;
        children = _children;
    }
};

class Solution {
public:
    vector<int> preorder(Node* root) {

    }
};
*/
```

### Java:

```
/*
// Definition for a Node.
class Node {
    public int val;
    public List<Node> children;

    public Node() {}
}
```

```

public Node(int _val) {
    val = _val;
}

public Node(int _val, List<Node> _children) {
    val = _val;
    children = _children;
}
};

/*
class Solution {
    public List<Integer> preorder(Node root) {
        ...
    }
}

```

### Python3:

```

"""
# Definition for a Node.
class Node:
    def __init__(self, val: Optional[int] = None, children: Optional[List['Node']] = None):
        self.val = val
        self.children = children
"""

class Solution:
    def preorder(self, root: 'Node') -> List[int]:

```