https://course.acciojob.com/idle?question=f5734fab-e776-403f-81c8-f0b370e52c70

- EASY
- Max Score: 30 Points

# Inorder traversal

---

**Given the `root` of a binary tree, return the inorder traversal of the given binary tree.**

## Input Format

**The first line of input contains a number `n`.**

**The second line of input contains `n` space seperated integer.**

## Output Format

**Return the inorder traversal of the given binary tree.**

## Example 1

Input

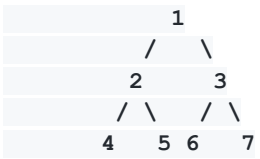
```
7
1 2 3 4 5 6 7
```

Output

```
4 2 5 1 6 3 7
```

Explanation

```
The given binary tree is
```



## Example 2

Input

```
4
1 2 3 4
```

Output

```
4 2 1 3
```

Explanation

```
The given tree is
            1
          /   \
         2     3
        /
       4
```

## Constraints

**1 <= n <= $10^3$**

### Topic Tags

- Trees

# My code

// n java

```
import java.util.LinkedList;
import java.util.Queue;
import java.io.*;
import java.util.*;

class Node {
    int data;
    Node left;
    Node right;

    Node(int data) {
        this.data = data;
        left = null;
        right = null;
    }
}

class Main {
    static Node buildTree(String str) {
        if (str.length() == 0 || str.charAt(0) == 'N') {
            return null;
        }
        String ip[] = str.split(" ");
        Node root = new Node(Integer.parseInt(ip[0]));
        Queue<Node> queue = new LinkedList<>();
        queue.add(root);
        int i = 1;
        while (queue.size() > 0 && i < ip.length) {
            Node currNode = queue.peek();
            queue.remove();
```

```
        String currVal = ip[i];
        if (!currVal.equals("N")) {
            currNode.left = new Node(Integer.parseInt(currVal));
            queue.add(currNode.left);
        }
        i++;
        if (i >= ip.length)
            break;
        currVal = ip[i];
        if (!currVal.equals("N")) {
            currNode.right = new Node(Integer.parseInt(currVal));
            queue.add(currNode.right);
        }
        i++;
    }
    return root;
}

public static void main(String[] args) throws IOException {
    Scanner sc = new Scanner(System.in);
    int n = sc.nextInt();
    sc.nextLine();
    String s = sc.nextLine();
    Node root = buildTree(s);
    Solution tree = new Solution();
    ArrayList<Integer> ans = tree.solve(root);
    for(Integer x:ans)
    System.out.print(x+" ");

    System.out.println();
```

```
        sc.close();
    }
}

class Solution {
    static ArrayList<Integer> al=new ArrayList<Integer> ();
        static void inorder(Node root)
        {

                if(root==null)
                        return;
                        inorder(root.left);
                  al.add(root.data);
                        inorder(root.right);



        }
    public ArrayList<Integer> solve(Node root) {
        // your code here
                inorder(root);
                return al;

    }
}
```