

**Problem Statement: Boundary Traversal of a binary tree.** Write a program for the Anti-Clockwise Boundary traversal of a binary tree.

```
class Node:
```

```
    def __init__(self, data):
```

```
        self.data = data
```

```
        self.left = None
```

```
        self.right = None
```

```
def boundary_traversal(root):
```

```
    if not root:
```

```
        return
```

```
def print_leaves(node):
```

```
    if node:
```

```
        print(node.data, end=" ")
```

```
        print_leaves(node.left)
```

```
        print_leaves(node.right)
```

```
def print_left_boundary(node):
```

```
    if node:
```

```
        if node.left:
```

```
            print(node.data, end=" ")
```

```
            print_left_boundary(node.left)
```

```

        elif node.right:
            print(node.data, end=" ")
            print_left_boundary(node.right)

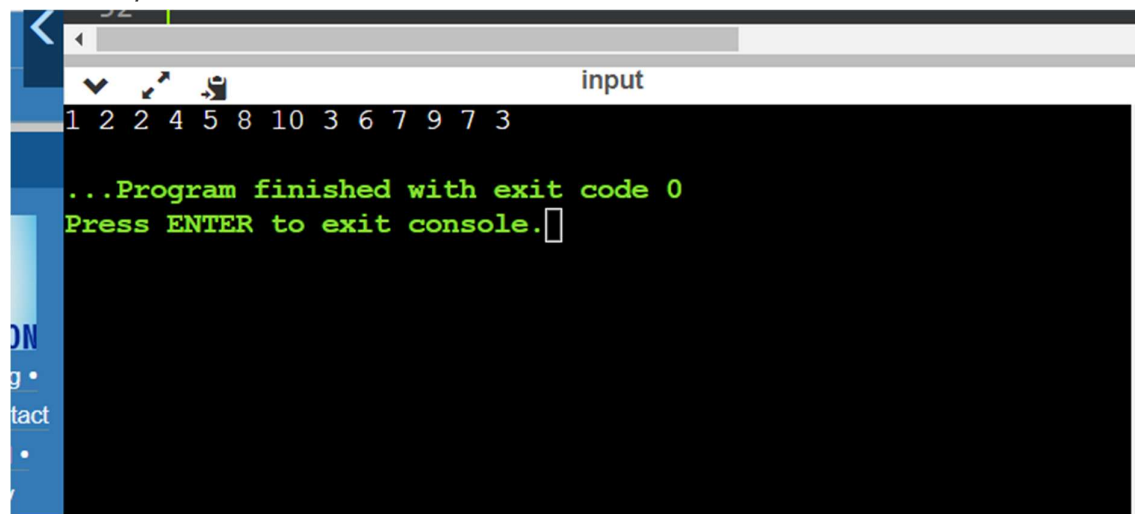
def print_right_boundary(node):
    if node:
        if node.right:
            print_right_boundary(node.right)
            print(node.data, end=" ")
        elif node.left:
            print_right_boundary(node.left)
            print(node.data, end=" ")
    print(root.data, end=" ")
    print_left_boundary(root.left)
    print_leaves(root.left)
    print_leaves(root.right)
    print_right_boundary(root.right)

root = Node(1)
root.left = Node(2)
root.right = Node(3)
root.left.left = Node(4)
root.left.right = Node(5)

```

```
root.left.right.left = Node(8)
root.left.right.left.left = Node(10)
root.right.left = Node(6)
root.right.right = Node(7)
root.right.right.left = Node(9)
```

```
boundary_traversal(root)
```



```
input
1 2 2 4 5 8 10 3 6 7 9 7 3
...Program finished with exit code 0
Press ENTER to exit console.█
```