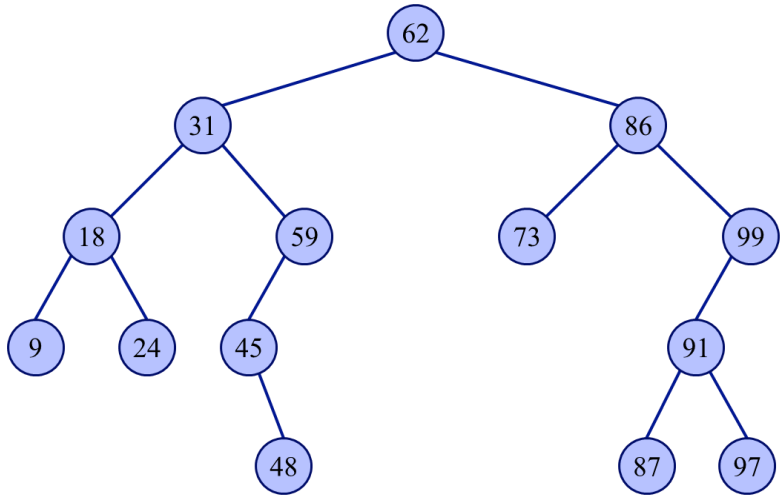
Tree Traversals:

- In-Order: 9 18 24 31 45 48 59 62 73 86 87 91 97 99
- Pre-Order: 62 31 18 9 24 59 45 48 86 73 99 91 87 97
- Post-Order: 9 24 18 48 45 59 31 73 87 97 91 99 86 62
- Level-Order: 62 31 86 18 59 73 99 9 24 45 91 48 87 97



Implementation:

In-Order:	Pre-Order	Post-Order
Go left Visit Go right	Visit Go left Go right	Go left Go right Visit

```
public void postOrder(TreeNode n) {
      if (n==null) {
            return;
      }
      postOrder(n.left);
      postOrder(n.right);
      System.out.println(n.data);
}
```

A different type of traversal: Level-Order

Strategy:

1. Create an empty Queue Q
2. Add root to Q
3. while Q is not empty:
   i) deqeueue
   ii) visit dequeued item
   iii) enqueue item's L then R

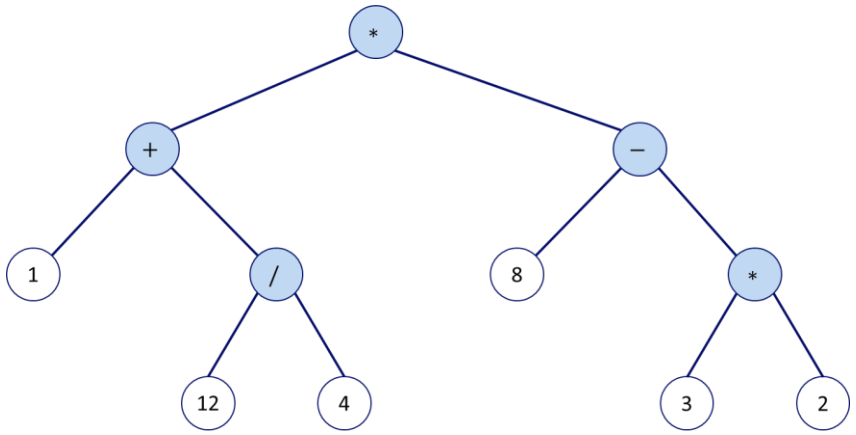
Output:

62 31 86 18 59 73 99 9 24 34 91 48 87 97

Queue:

62	31	86	18	59	73	99	9	24	45	91	48	87	97

Tree Traversal Application:



Evaluating the expression generated from a Post-Order traversal:

1. Traverse through each item in the post-order expression

2. If item is an operand, push to stack. Otherwise, pop two elements.
   i. Let A be first popped element
   ii. Let B be second popped element
   iii. Evaluate B <operator> A
   iv. Push result to stack

3. Pop final item and return it.

Post-order traversal: 1 12 4 / + 8 3 2 * − *