Traversals

In-order: 1 2 3 4 5 6 7
1) Traverse left subtree
2) Visit node (and do something)
3) Traverse right subtree

Pre-order: 4 2 1 3 6 5 7
1) Visit node (and do something)
2) Traverse left subtree.
3) Traverse right subtree.

Post-order: 1 3 2 5 7 6 4
1) Traverse left subtree
2) Traverse right subtree
3) Visit node (and do something)

Review: This is a binary search tree (BST). In a BST, a node’s left subtree must be less than the node’s value. Its right subtree must be greater than the node’s value (assuming all data are distinct values). BSTs are convenient for explaining tree traversals.

See the code snippets for today.
In-order:

Pre-order

Post-order
Alternative way to think about tree traversals

Traversal:
- Traverse left subtree: 1
- Visit root: 2
- Traverse right subtree: 3
- Visit root: 4
- Traverse right subtree:
  - Traverse left subtree: 5
  - Visit root: 6
  - Visit root: 7

(output, assuming we simply print each node when we "visit" it.)