Data Structures
Tree Traversals

COMP128 Data Structures

Tree Traversals

“Traversal” means to visit every node in the tree in a prescribed order, usually to perform some action on each node;

There are three kinds of traversals characterized by when they visit a node in relation to its subtrees:

- **Pre-order:** visit root node, traverse left subtree, traverse right subtree
- **In-order:** traverse left subtree, visit root node, traverse right subtree
- **Post-order:** traverse left subtree, traverse right subtree, visit root node

Preorder Traversal

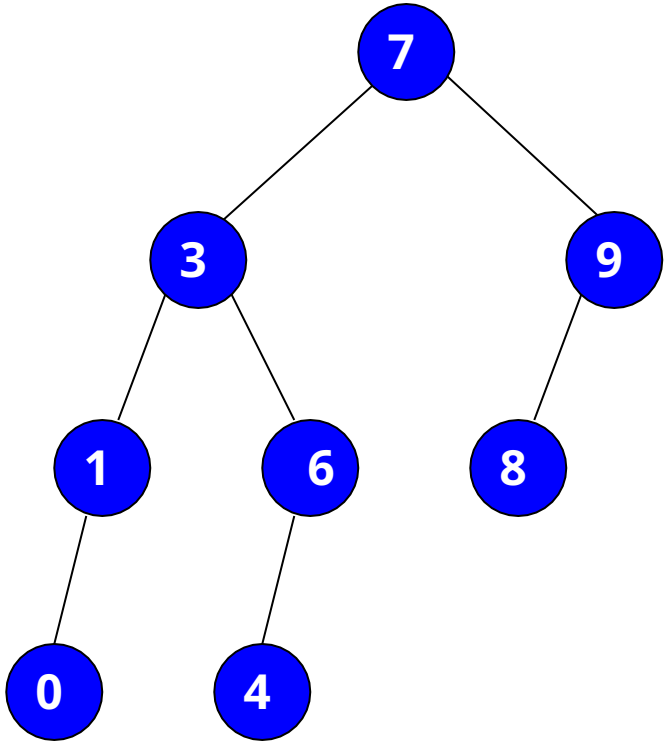
Algorithm for Preorder:

If the tree is empty

- Return;

Else

- Visit the root node;
- Preorder traverse the left subtree;
- Preorder traverse the right subtree;



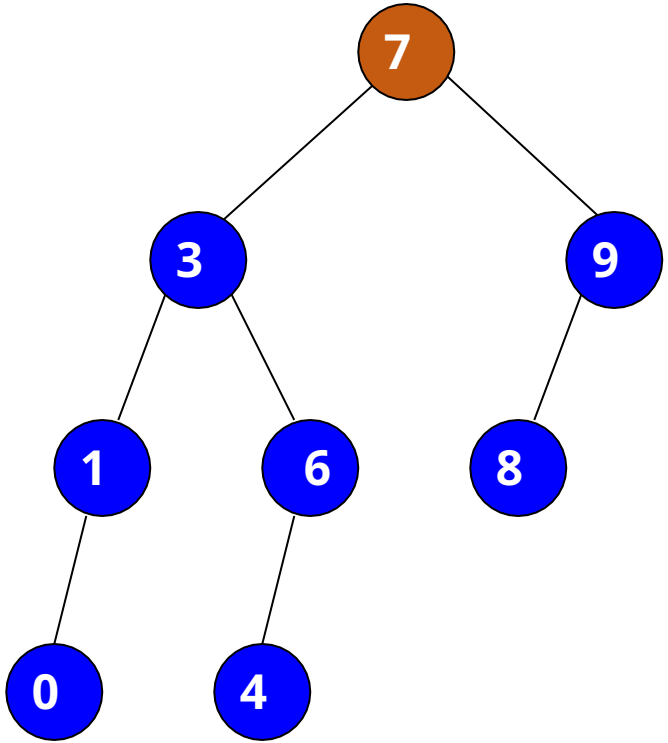
Preorder Traversal

Algorithm for Preorder:

If the tree is empty
- Return;

Else
- Visit the root node;
- Preorder traverse the left subtree;
- Preorder traverse the right subtree;

Order: 7



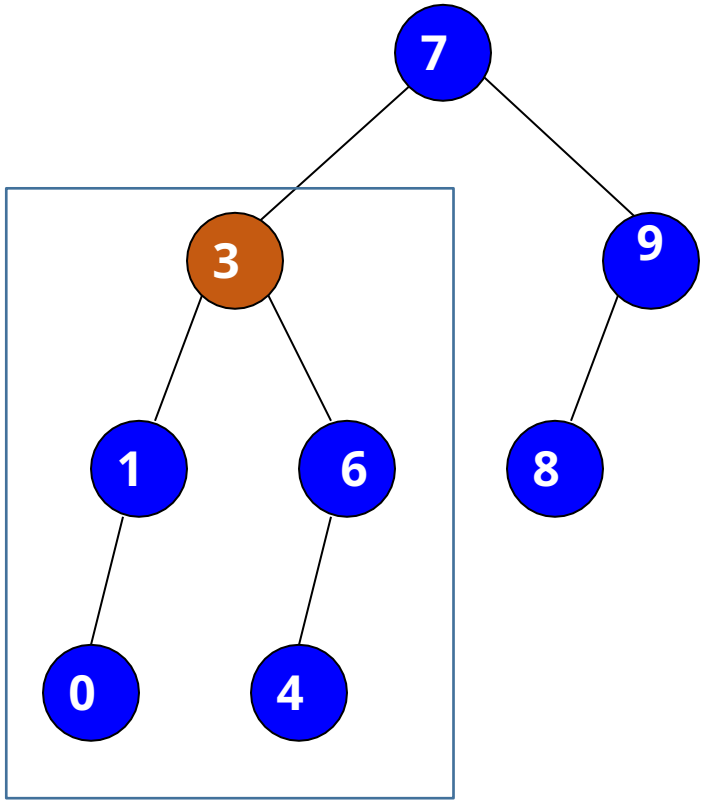
Preorder Traversal

Algorithm for Preorder:

If the tree is empty
- Return;

Else
- Visit the root node;
- Preorder traverse the left subtree;
- Preorder traverse the right subtree;

Order: 7, 3



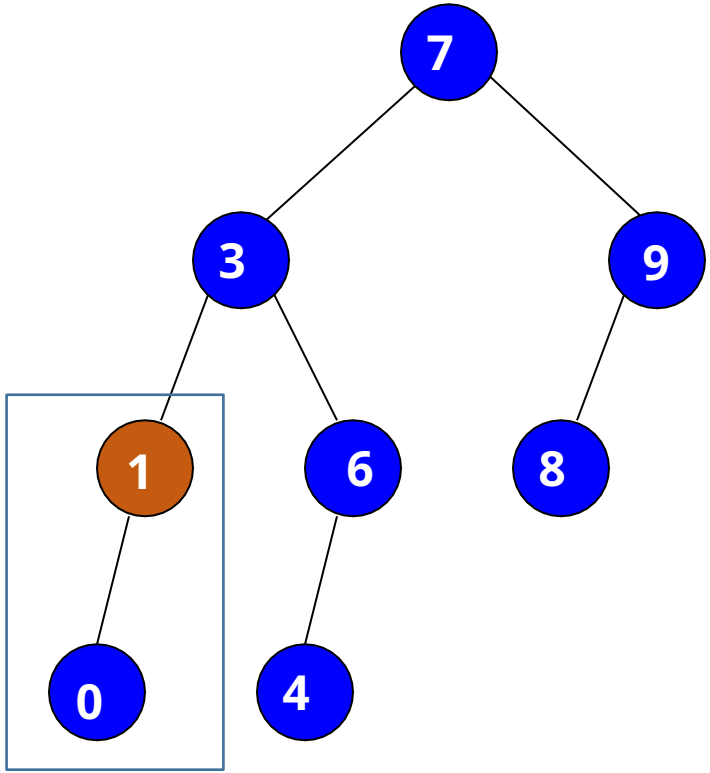
Preorder Traversal

Algorithm for Preorder:

If the tree is empty
- Return;

Else
- Visit the root node;
- Preorder traverse the left subtree;
- Preorder traverse the right subtree;

Order: 7, 3, 1



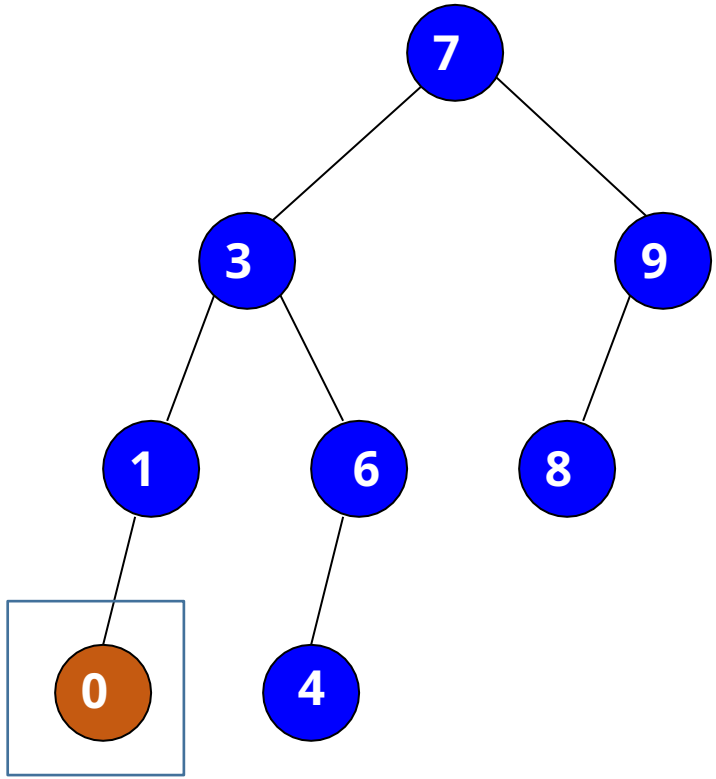
Preorder Traversal

Algorithm for Preorder:

If the tree is empty
- Return;

Else
- Visit the root node;
- Preorder traverse the left subtree;
- Preorder traverse the right subtree;

Order: 7, 3, 1, 0



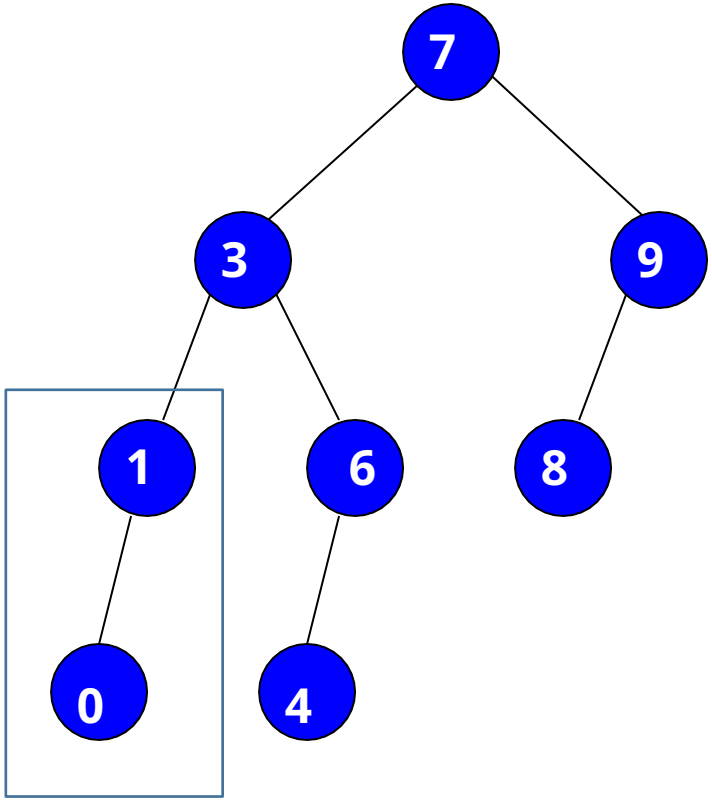
Preorder Traversal

Algorithm for Preorder:

If the tree is empty
- Return;

Else
- Visit the root node;
- Preorder traverse the left subtree;
- Preorder traverse the right subtree;

Order: 7, 3, 1, 0



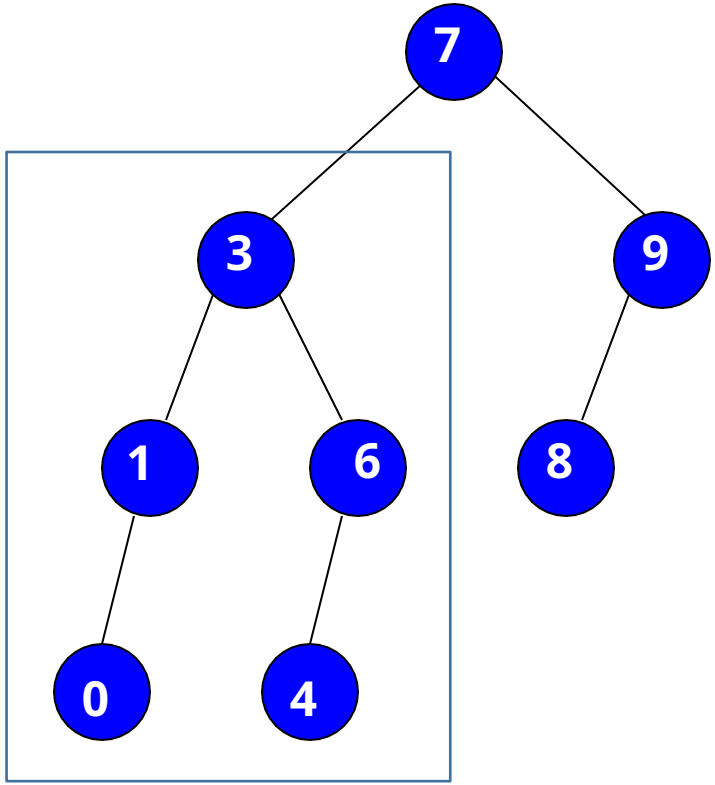
Preorder Traversal

Algorithm for Preorder:

If the tree is empty
- Return;

Else
- Visit the root node;
- Preorder traverse the left subtree;
- Preorder traverse the right subtree;

Order: 7, 3, 1, 0



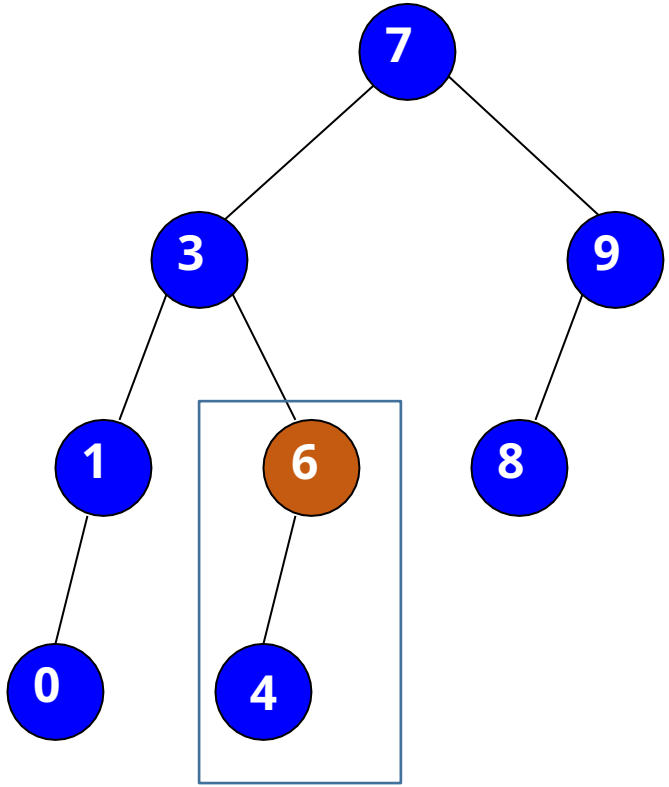
Preorder Traversal

Algorithm for Preorder:

If the tree is empty
- Return;

Else
- Visit the root node;
- Preorder traverse the left subtree;
- Preorder traverse the right subtree;

Order: 7, 3, 1, 0



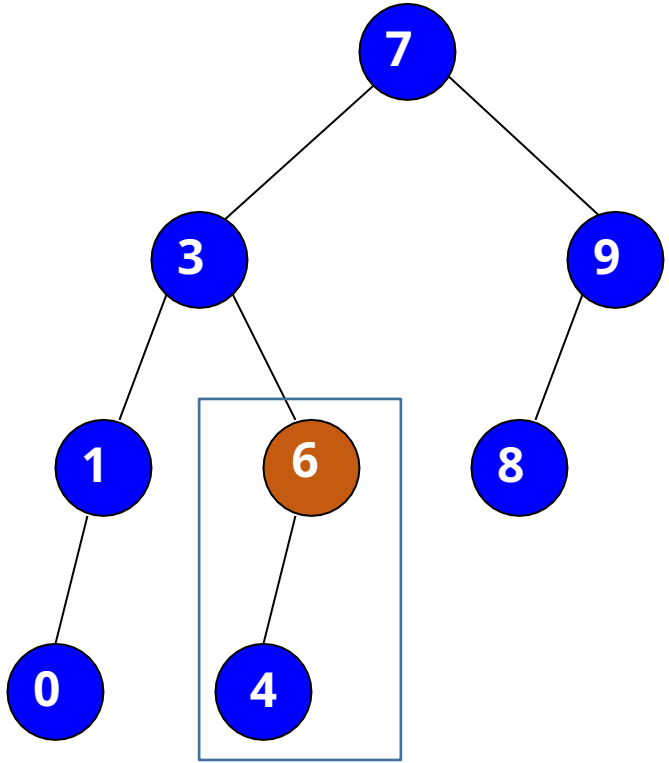
Preorder Traversal

Algorithm for Preorder:

If the tree is empty
- Return;

Else
- Visit the root node;
- Preorder traverse the left subtree;
- Preorder traverse the right subtree;

Order: 7, 3, 1, 0, 6



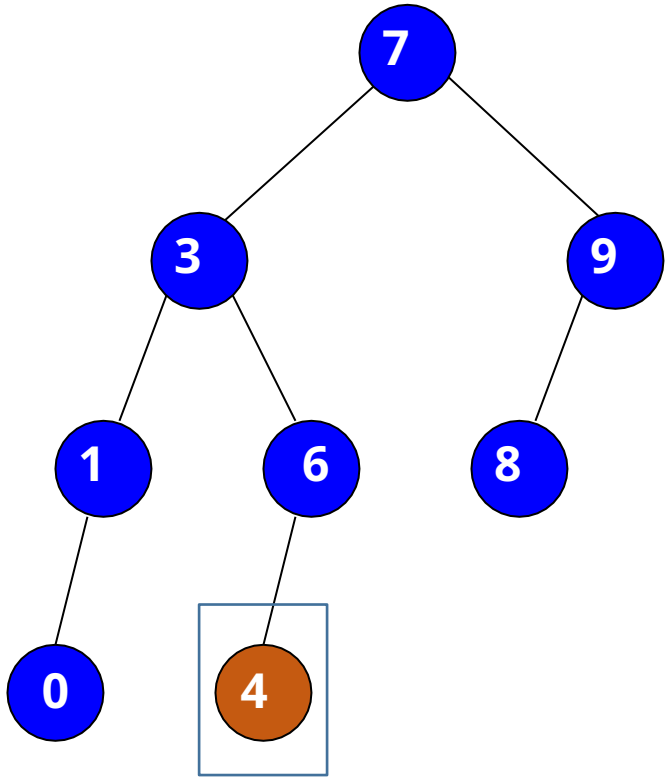
Preorder Traversal

Algorithm for Preorder:

If the tree is empty
- Return;

Else
- Visit the root node;
- Preorder traverse the left subtree;
- Preorder traverse the right subtree;

Order: 7, 3, 1, 0, 6, 4



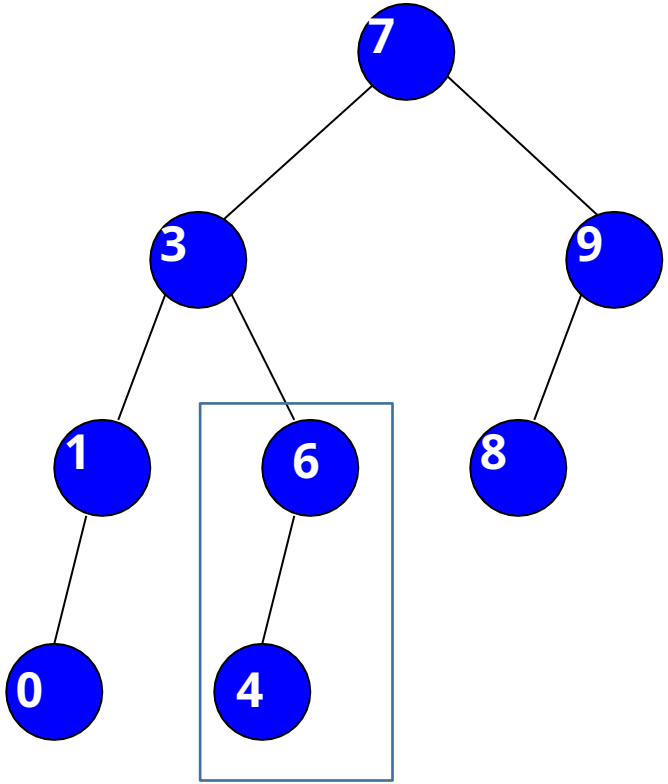
Preorder Traversal

Algorithm for Preorder:

If the tree is empty
- Return;

Else
- Visit the root node;
- Preorder traverse the left subtree;
- Preorder traverse the right subtree;

Order: 7, 3, 1, 0, 6, 4



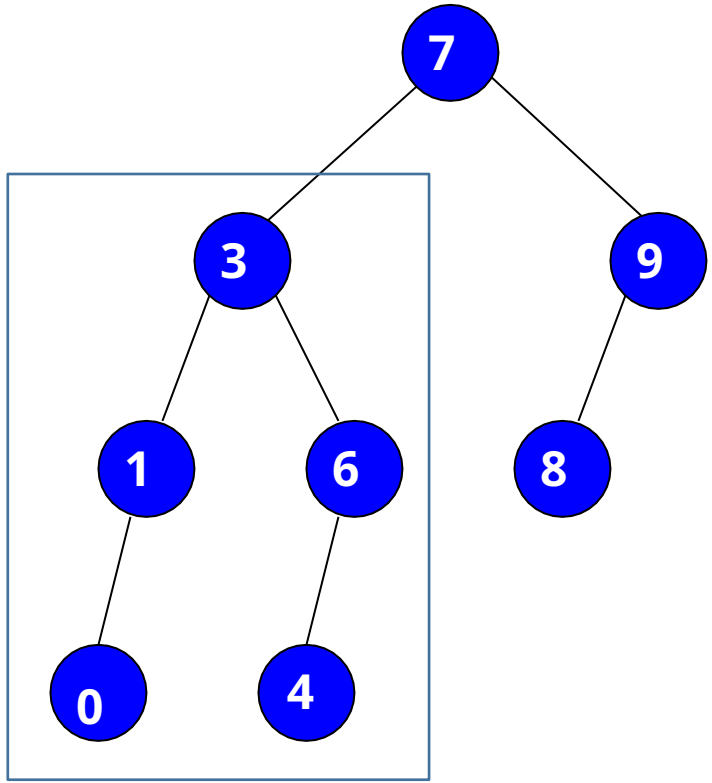
Preorder Traversal

Algorithm for Preorder:

If the tree is empty
- Return;

Else
- Visit the root node;
- Preorder traverse the left subtree;
- Preorder traverse the right subtree;

Order: 7, 3, 1, 0, 6, 4



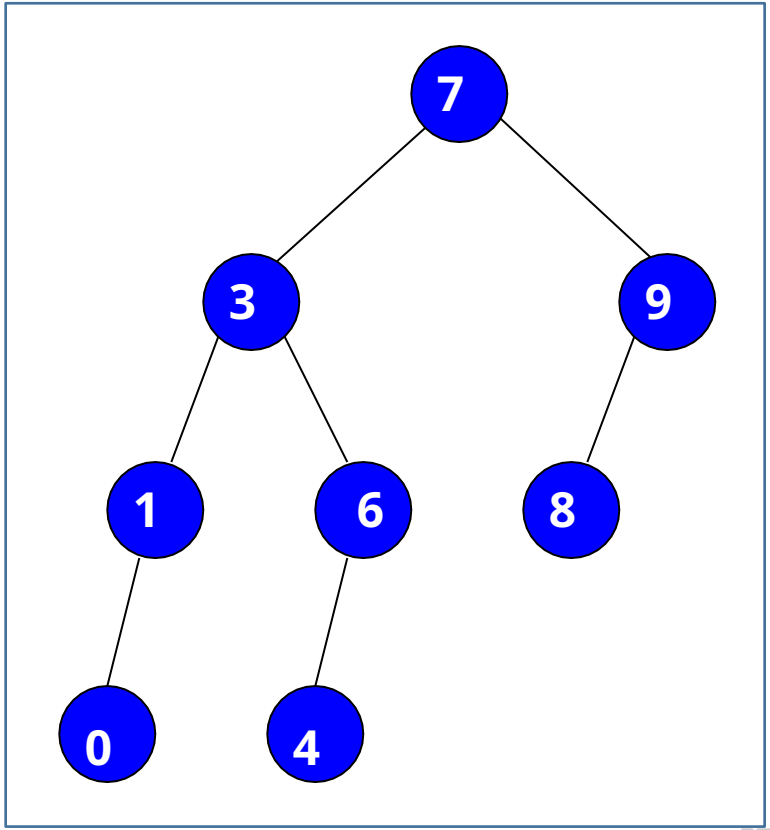
Preorder Traversal

Algorithm for Preorder:

If the tree is empty
- Return;

Else
- Visit the root node;
- Preorder traverse the left subtree;
- Preorder traverse the right subtree;

Order: 7, 3, 1, 0, 6, 4



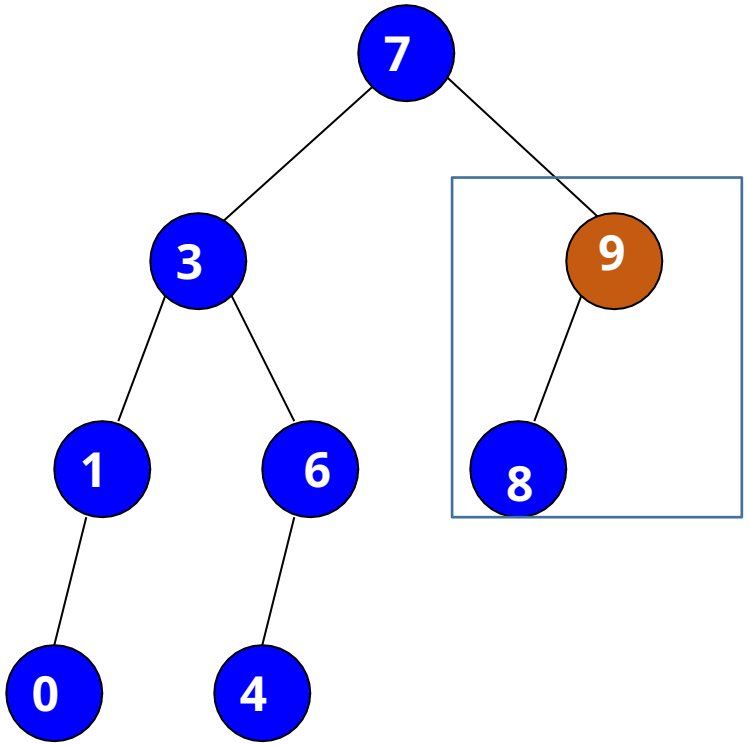
Preorder Traversal

Algorithm for Preorder:

If the tree is empty
- Return;

Else
- Visit the root node;
- Preorder traverse the left subtree;
- Preorder traverse the right subtree;

Order: 7, 3, 1, 0, 6, 4, 9



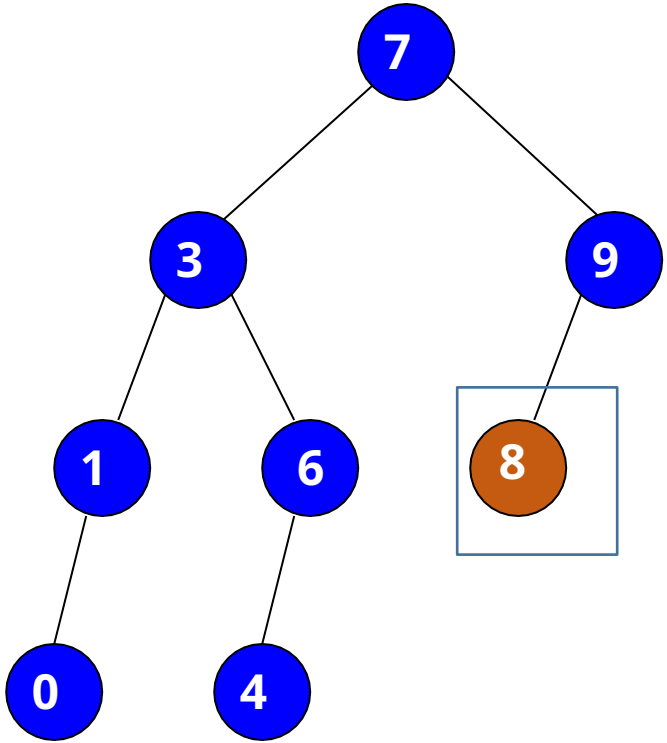
Preorder Traversal

Algorithm for Preorder:

If the tree is empty
- Return;

Else
- Visit the root node;
- Preorder traverse the left subtree;
- Preorder traverse the right subtree;

Order: 7, 3, 1, 0, 6, 4, 9, 8



Inorder Traversal

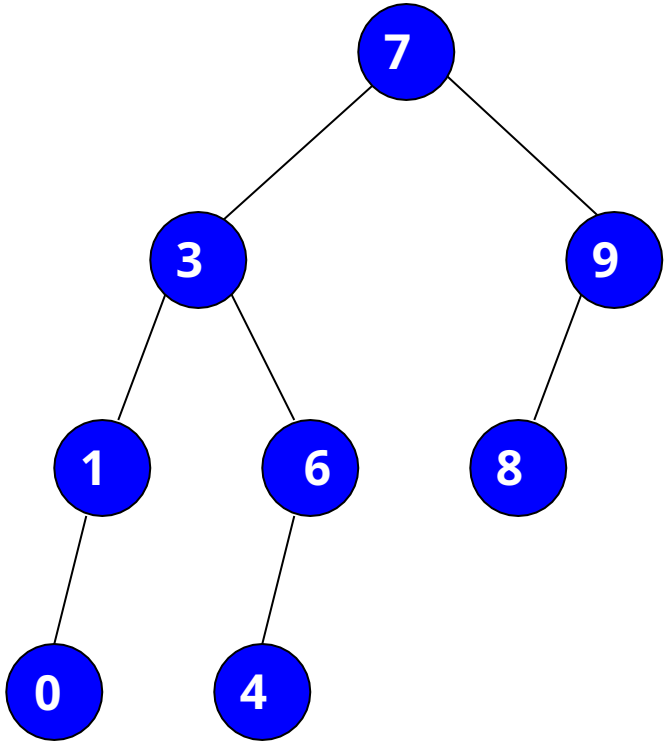
Algorithm for Inorder:

If the tree is empty

- Return;

Else

- Inorder traverse the left subtree;
- Visit the root;
- Inorder traverse the right subtree;

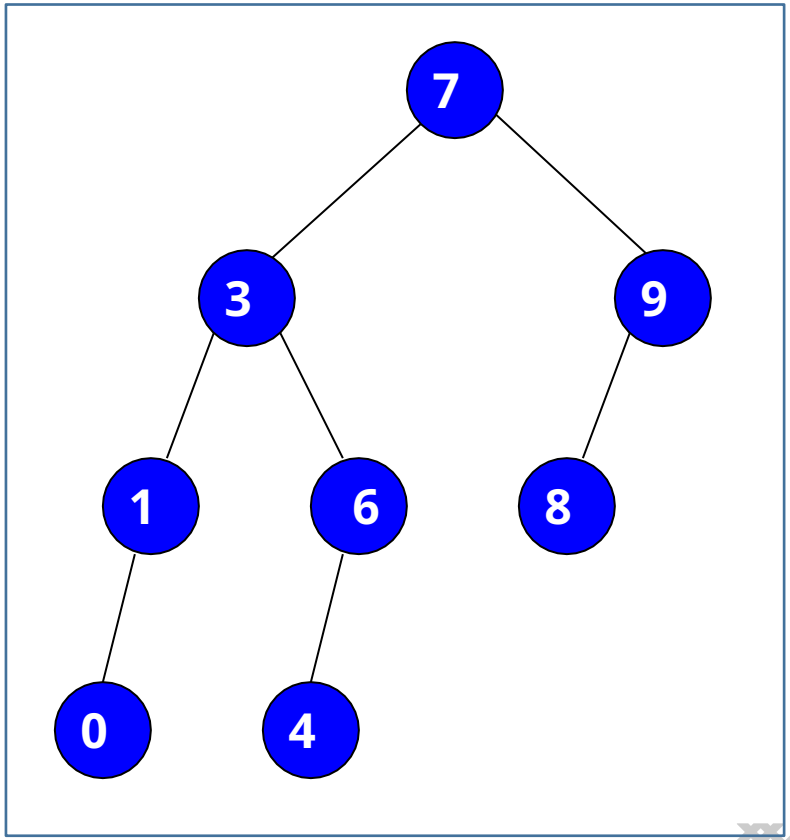


Inorder Traversal

Algorithm for Inorder:

If the tree is empty
- Return;

Else
- Inorder traverse the left subtree;
- Visit the root;
- Inorder traverse the right subtree;



Inorder Traversal

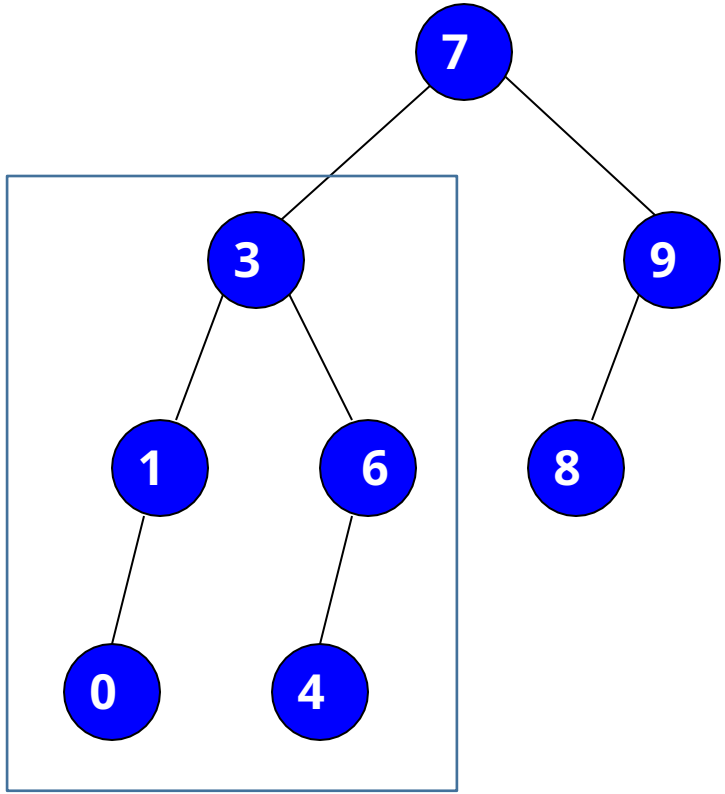
Algorithm for Inorder:

If the tree is empty

- Return;

Else

- Inorder traverse the left subtree;
- Visit the root;
- Inorder traverse the right subtree;

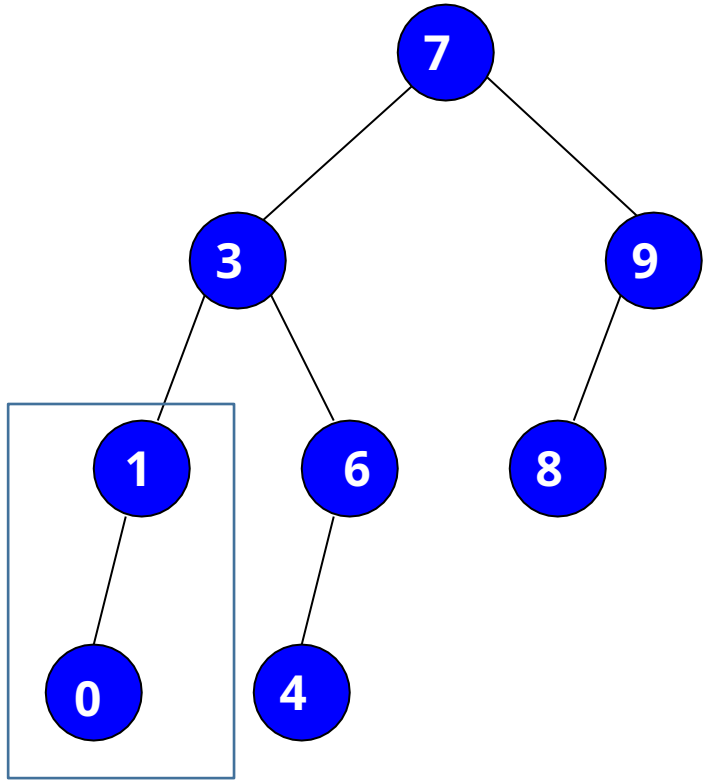


Inorder Traversal

Algorithm for Inorder:

If the tree is empty
- Return;

Else
- Inorder traverse the left subtree;
- Visit the root;
- Inorder traverse the right subtree;



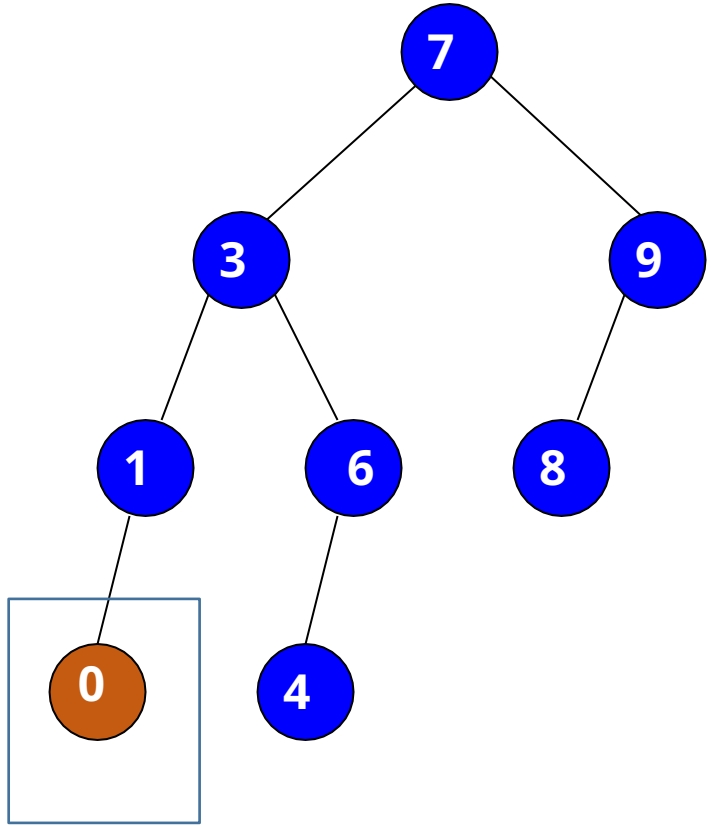
Inorder Traversal

Algorithm for Inorder:

If the tree is empty
- Return;

Else
- Inorder traverse the left subtree;
- Visit the root;
- Inorder traverse the right subtree;

Order: 0



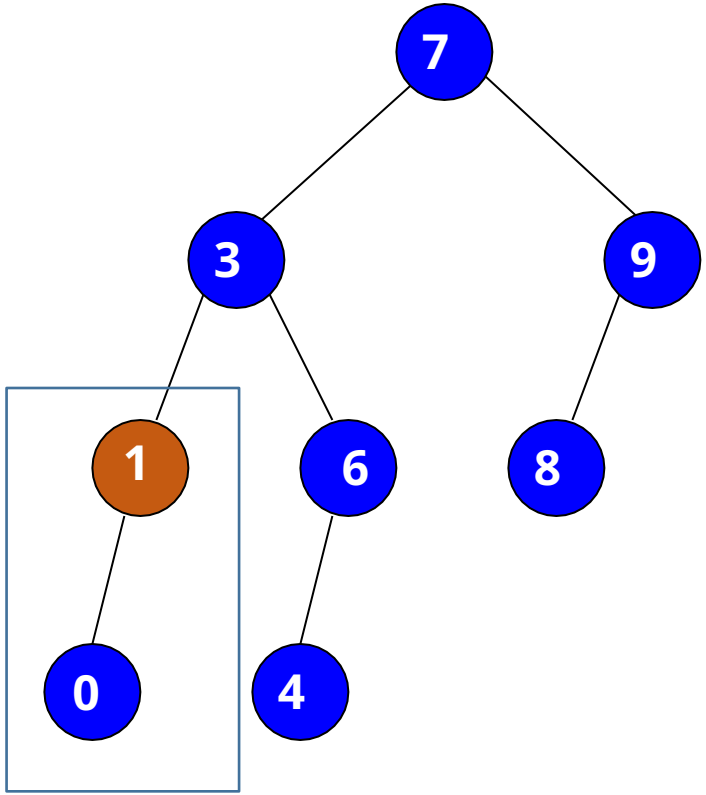
Inorder Traversal

Algorithm for Inorder:

If the tree is empty
- Return;

Else
- Inorder traverse the left subtree;
- Visit the root;
- Inorder traverse the right subtree;

Order: 0, 1



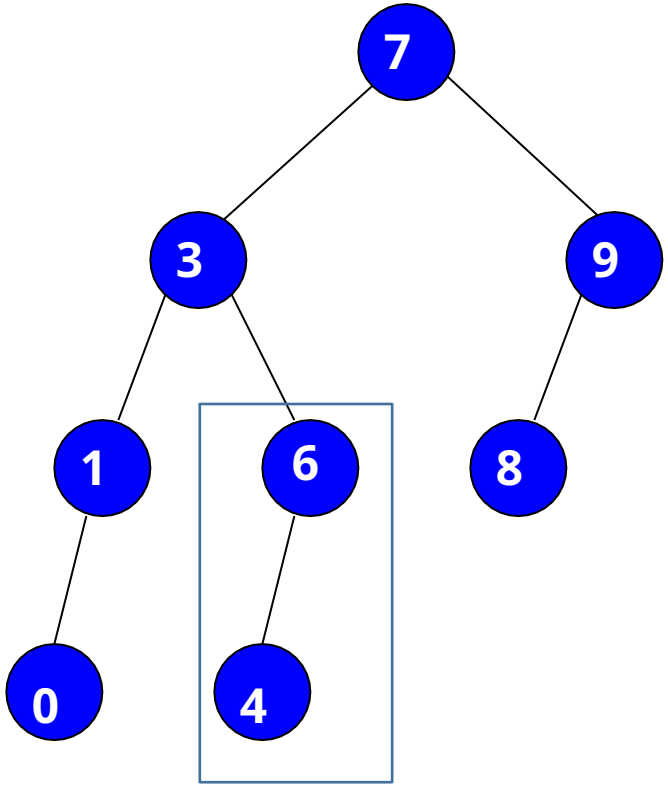
Inorder Traversal

Algorithm for Inorder:

If the tree is empty
- Return;

Else
- Inorder traverse the left subtree;
- Visit the root;
- Inorder traverse the right subtree;

Order: 0, 1, 3



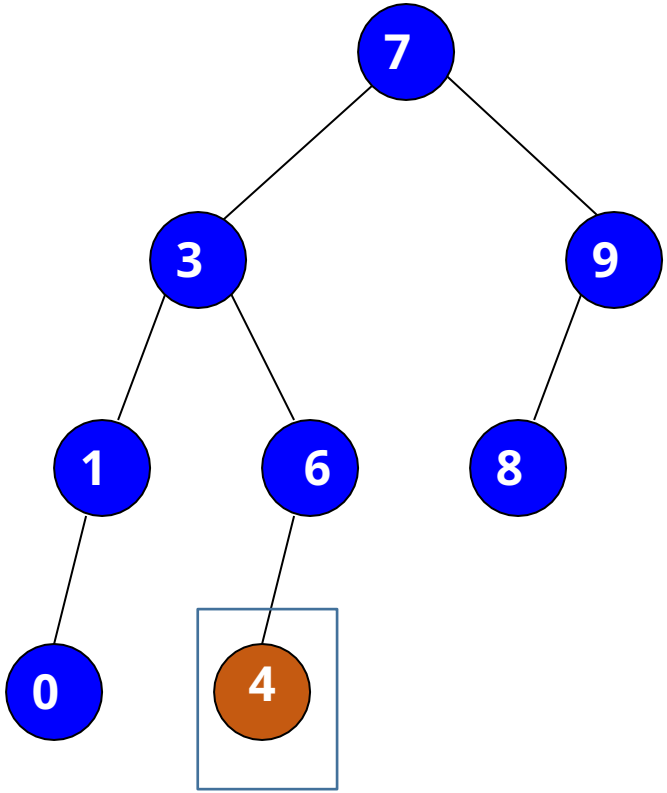
Inorder Traversal

Algorithm for Inorder:

If the tree is empty
- Return;

Else
- Inorder traverse the left subtree;
- Visit the root;
- Inorder traverse the right subtree;

Order: 0, 1, 3, 4



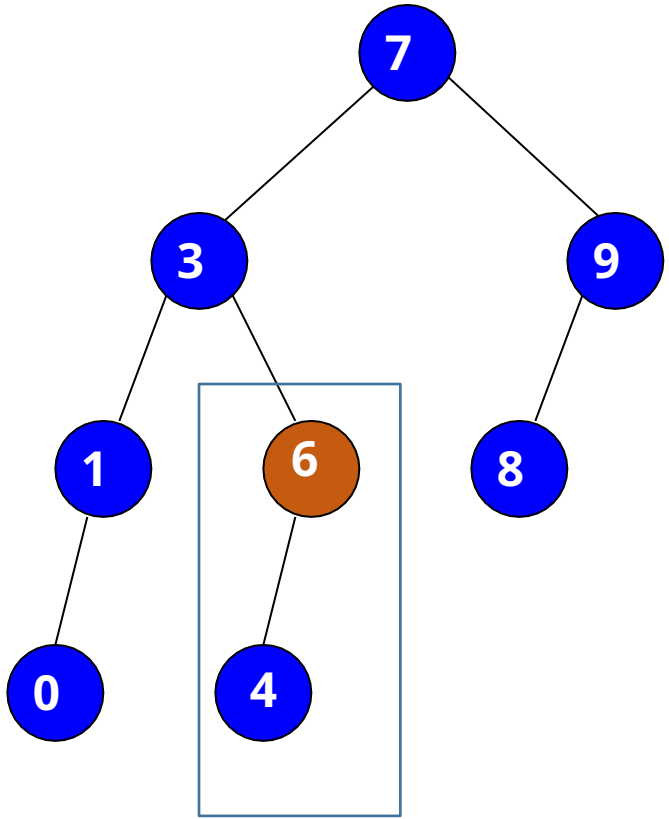
Inorder Traversal

Algorithm for Inorder:

If the tree is empty
- Return;

Else
- Inorder traverse the left subtree;
- Visit the root;
- Inorder traverse the right subtree;

Order: 0, 1, 3, 4, 6



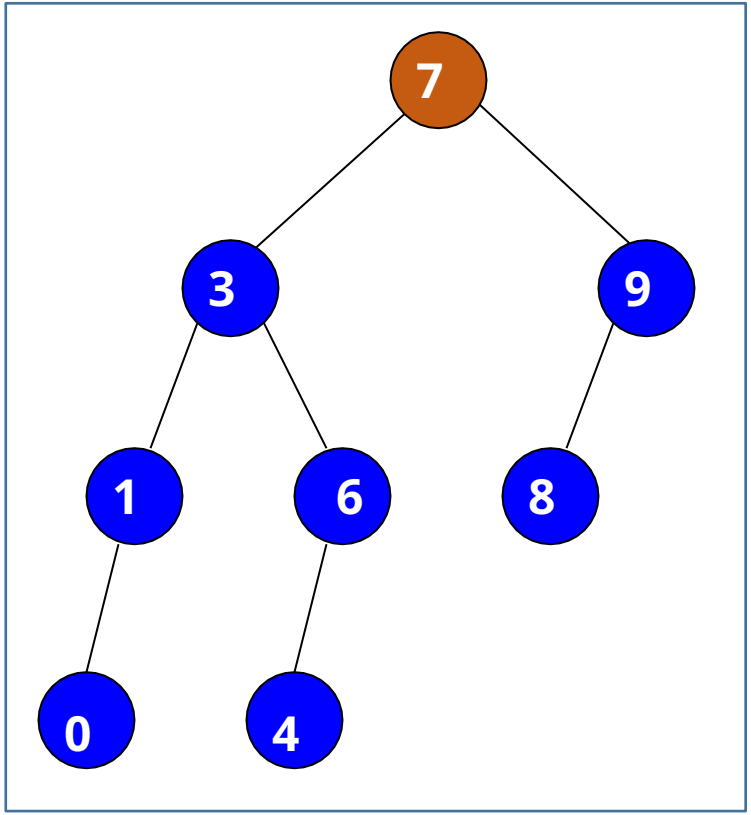
Inorder Traversal

Algorithm for Inorder:

If the tree is empty
- Return;

Else
- Inorder traverse the left subtree;
- Visit the root;
- Inorder traverse the right subtree;

Order: 0, 1, 3, 4, 6, 7



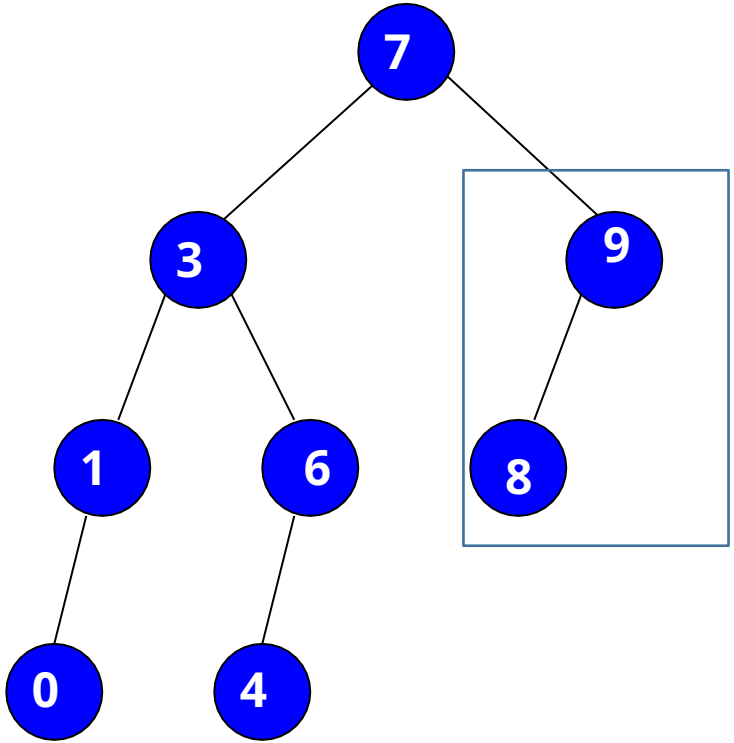
Inorder Traversal

Algorithm for Inorder:

If the tree is empty
- Return;

Else
- Inorder traverse the left subtree;
- Visit the root;
- Inorder traverse the right subtree;

Order: 0, 1, 3, 4, 6, 7



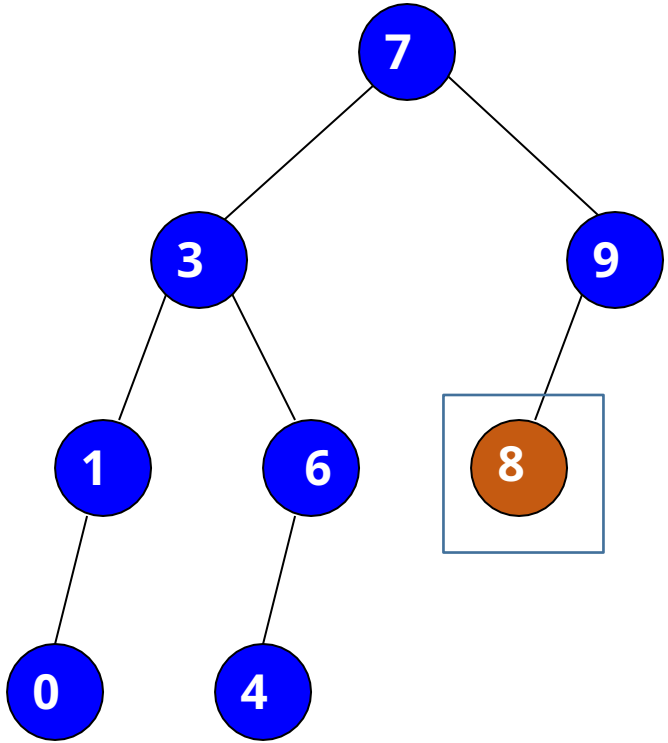
Inorder Traversal

Algorithm for Inorder:

If the tree is empty
- Return;

Else
- Inorder traverse the left subtree;
- Visit the root;
- Inorder traverse the right subtree;

Order: 0, 1, 3, 4, 6, 7, 8



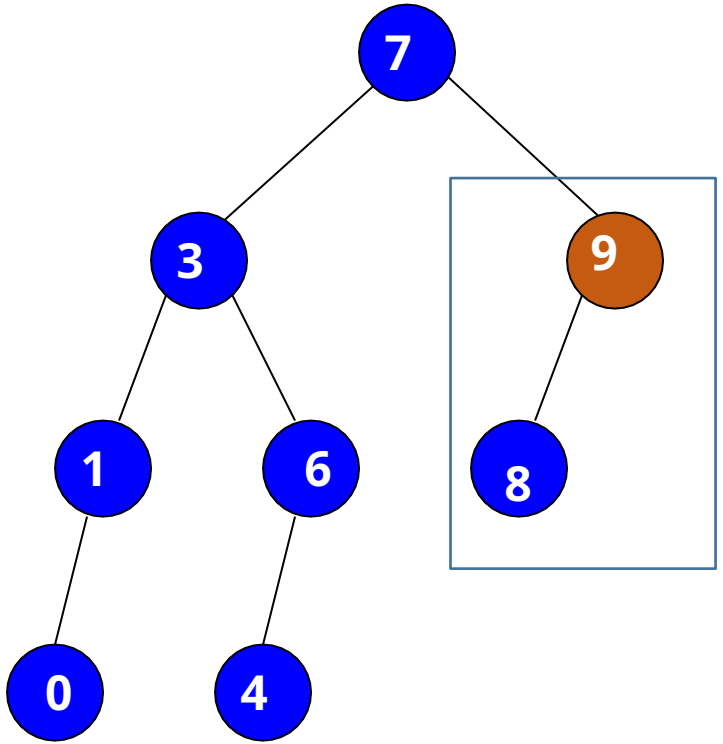
Inorder Traversal

Algorithm for Inorder:

If the tree is empty
- Return;

Else
- Inorder traverse the left subtree;
- Visit the root;
- Inorder traverse the right subtree;

Order: 0, 1, 3, 4, 6, 7, 8, 9



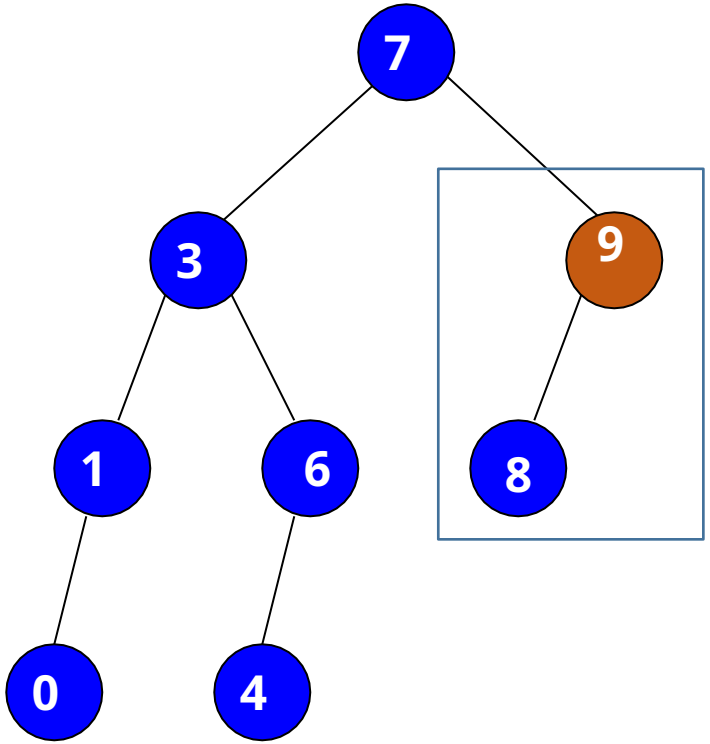
Inorder Traversal

Algorithm for Inorder:

If the tree is empty
- Return;

Else
- Inorder traverse the left subtree;
- Visit the root;
- Inorder traverse the right subtree;

Order: 0, 1, 3, 4, 6, 7, 8, 9



Postorder Traversal

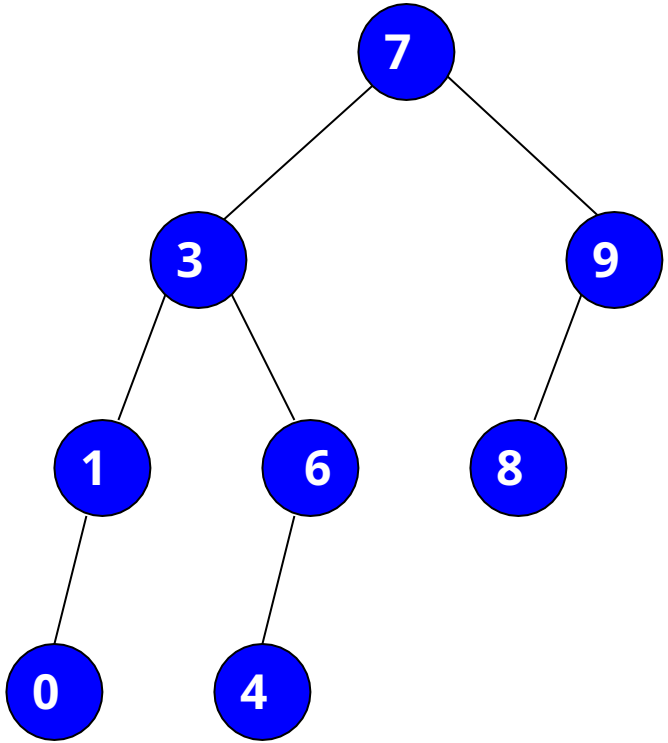
Algorithm for Postorder:

If the tree is empty

- Return;

Else

- Postorder traverse the left subtree;
- Postorder traverse the right subtree;
- Visit the root;



Postorder Traversal

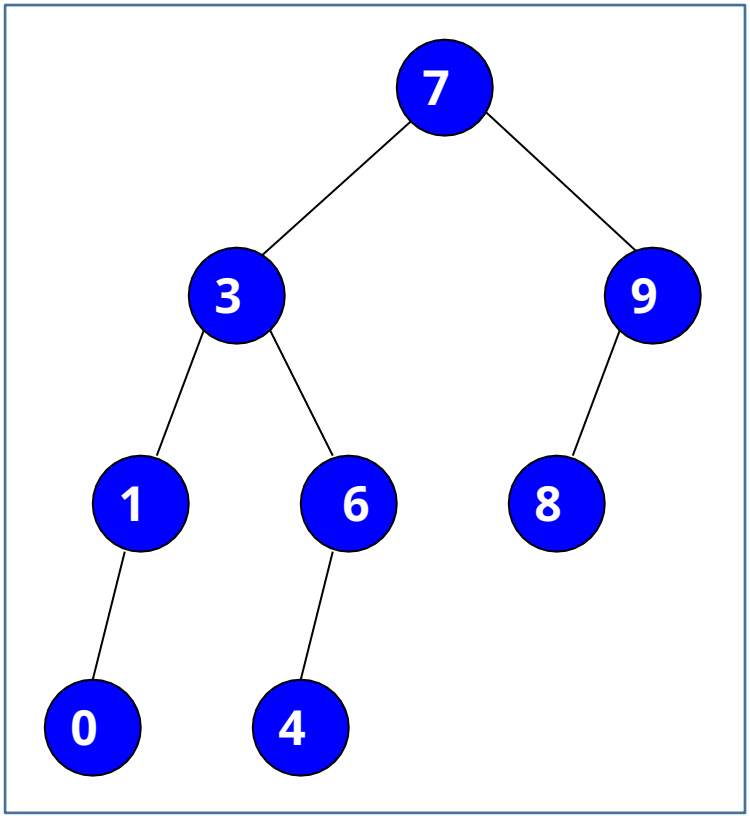
Algorithm for Postorder:

If the tree is empty

- Return;

Else

- Postorder traverse the left subtree;
- Postorder traverse the right subtree;
- Visit the root;



Postorder Traversal

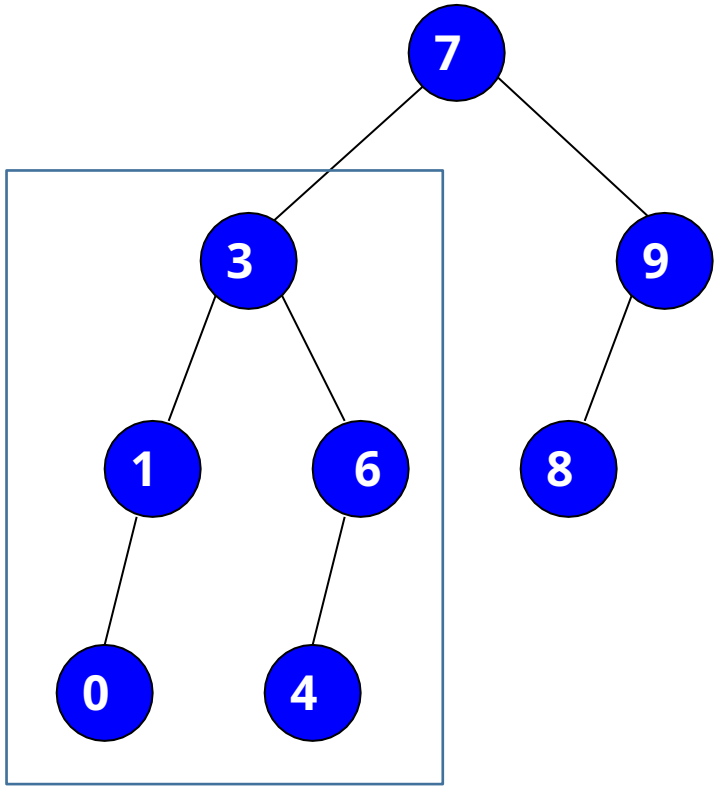
Algorithm for Postorder:

If the tree is empty

- Return;

Else

- Postorder traverse the left subtree;
- Postorder traverse the right subtree;
- Visit the root;



Postorder Traversal

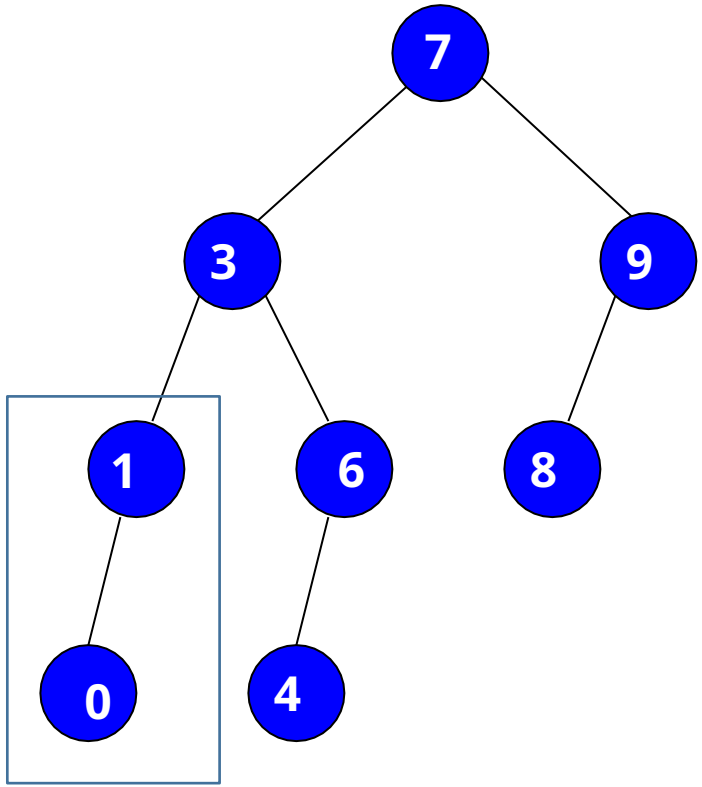
Algorithm for Postorder:

If the tree is empty

- Return;

Else

- Postorder traverse the left subtree;
- Postorder traverse the right subtree;
- Visit the root;



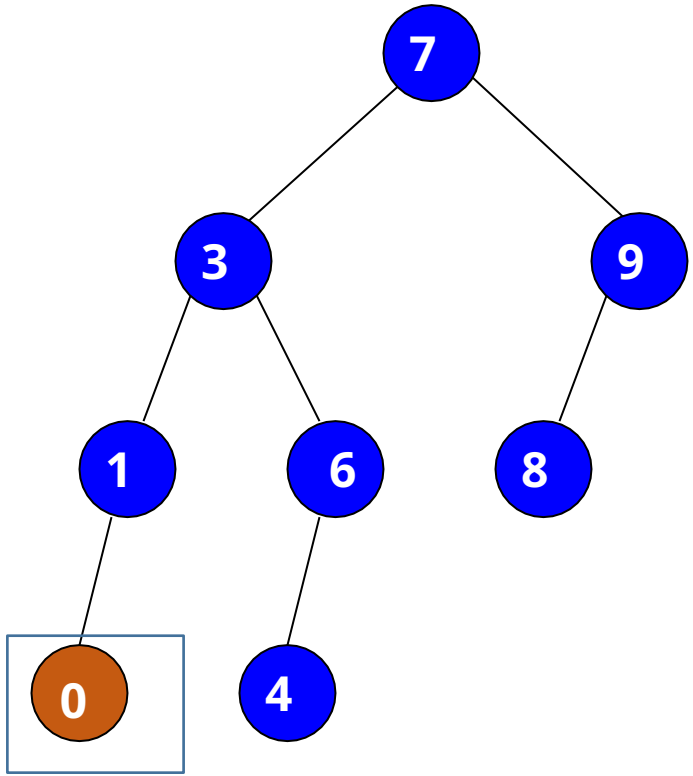
Postorder Traversal

Algorithm for Postorder:

If the tree is empty
- Return;

Else
- Postorder traverse the left subtree;
- Postorder traverse the right subtree;
- Visit the root;

Order: 0



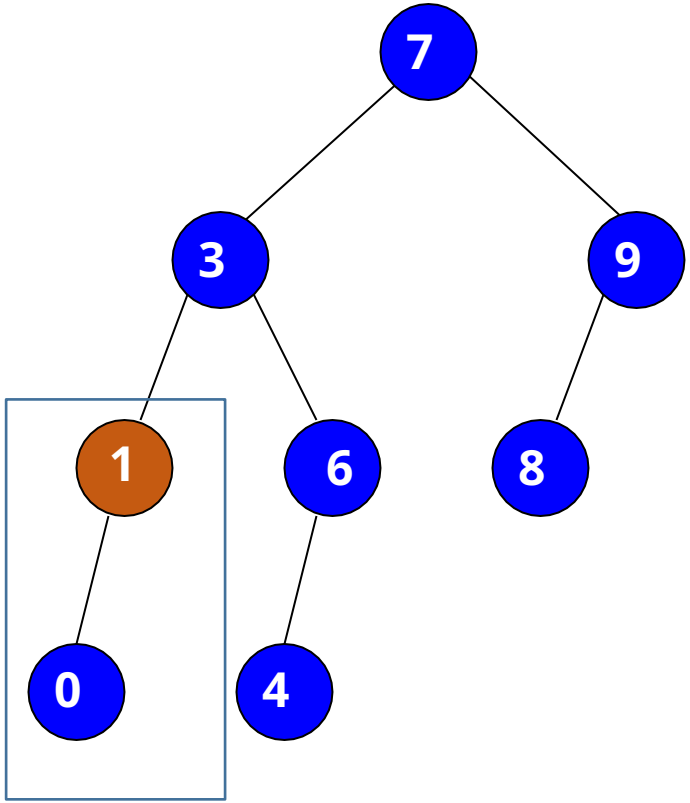
Postorder Traversal

Algorithm for Postorder:

If the tree is empty
- Return;

Else
- Postorder traverse the left subtree;
- Postorder traverse the right subtree;
- Visit the root;

Order: 0, 1



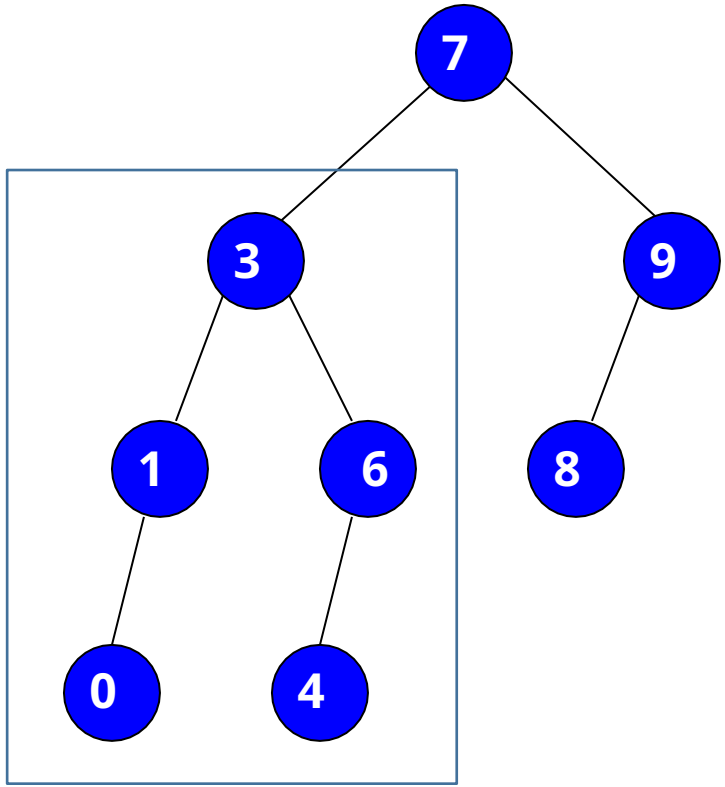
Postorder Traversal

Algorithm for Postorder:

If the tree is empty
- Return;

Else
- Postorder traverse the left subtree;
- Postorder traverse the right subtree;
- Visit the root;

Order: 0, 1



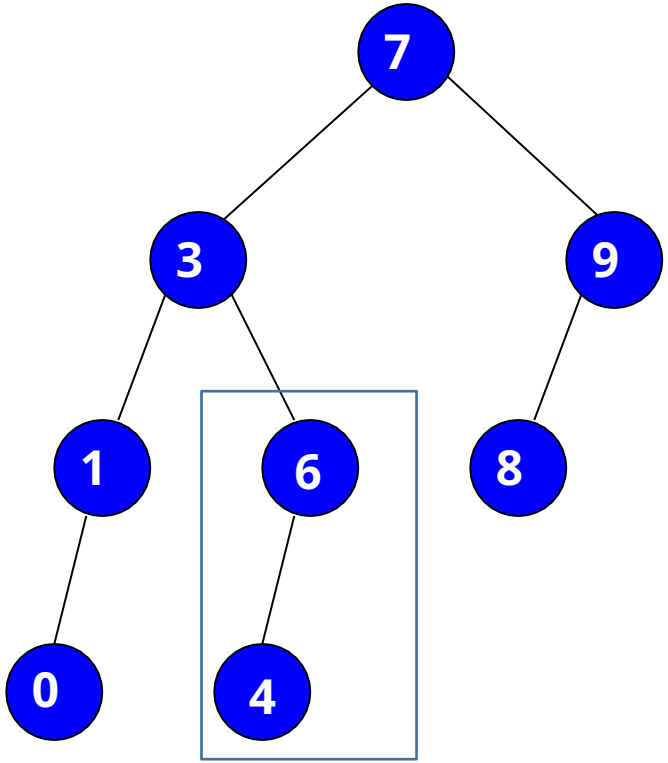
Postorder Traversal

Algorithm for Postorder:

If the tree is empty
- Return;

Else
- Postorder traverse the left subtree;
- Postorder traverse the right subtree;
- Visit the root;

Order: 0, 1



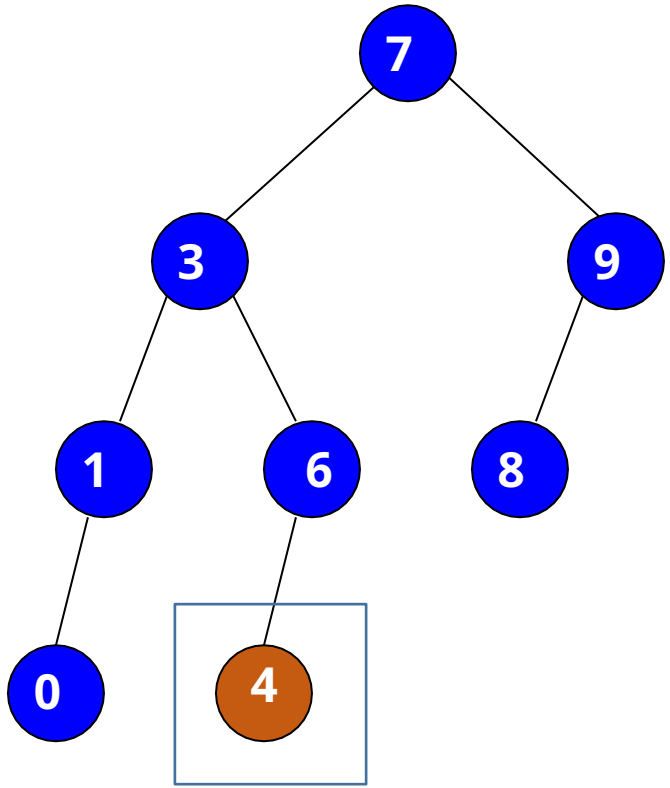
Postorder Traversal

Algorithm for Postorder:

If the tree is empty
- Return;

Else
- Postorder traverse the left subtree;
- Postorder traverse the right subtree;
- Visit the root;

Order: 0, 1, 4, 6



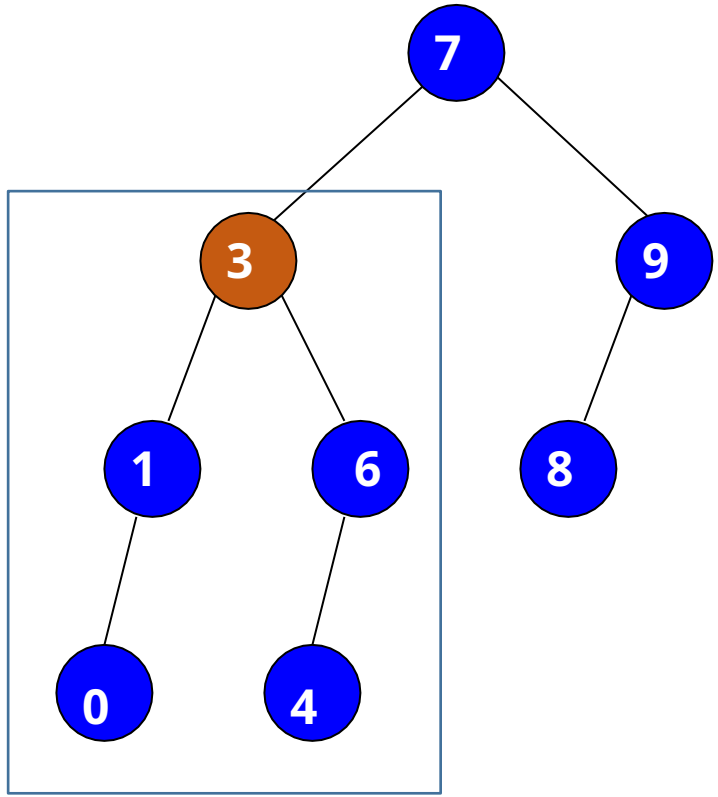
Postorder Traversal

Algorithm for Postorder:

If the tree is empty
- Return;

Else
- Postorder traverse the left subtree;
- Postorder traverse the right subtree;
- Visit the root;

Order: 0, 1, 4, 6, 3



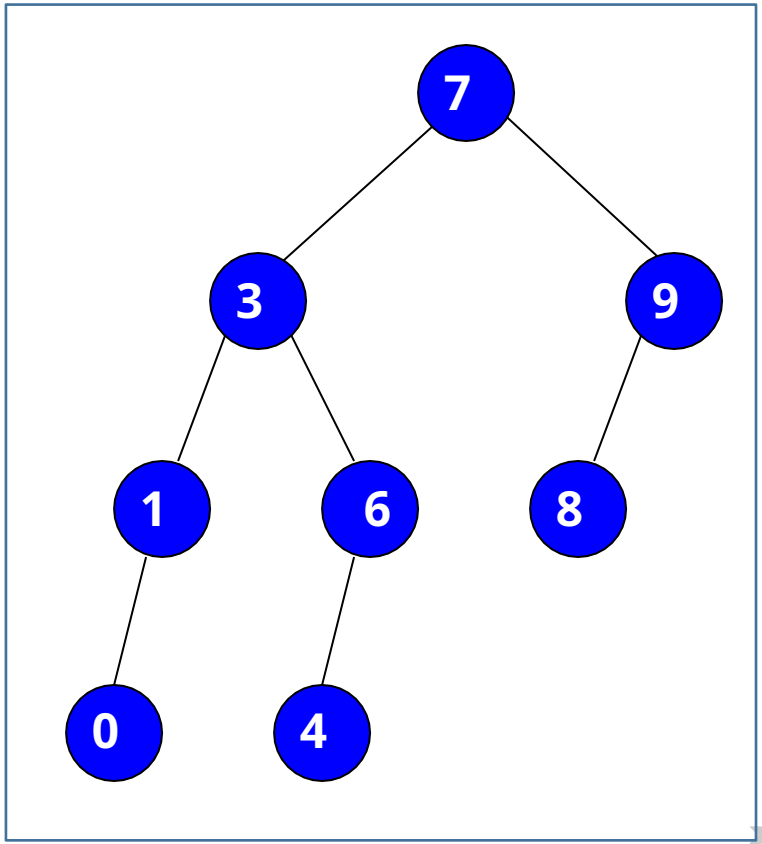
Postorder Traversal

Algorithm for Postorder:

If the tree is empty
- Return;

Else
- Postorder traverse the left subtree;
- Postorder traverse the right subtree;
- Visit the root;

Order: 0, 1, 4, 6, 3



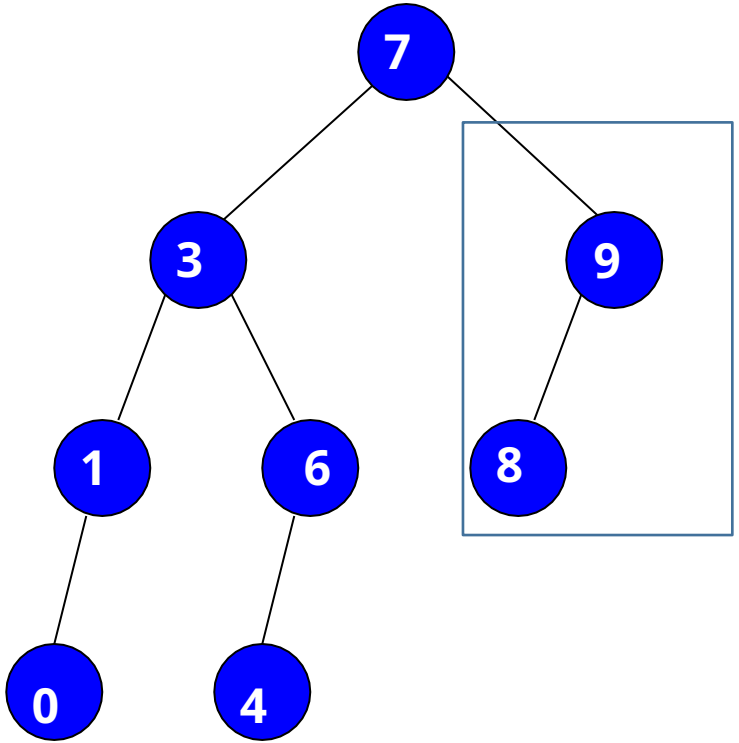
Postorder Traversal

Algorithm for Postorder:

If the tree is empty
- Return;

Else
- Postorder traverse the left subtree;
- Postorder traverse the right subtree;
- Visit the root;

Order: 0, 1, 4, 6, 3



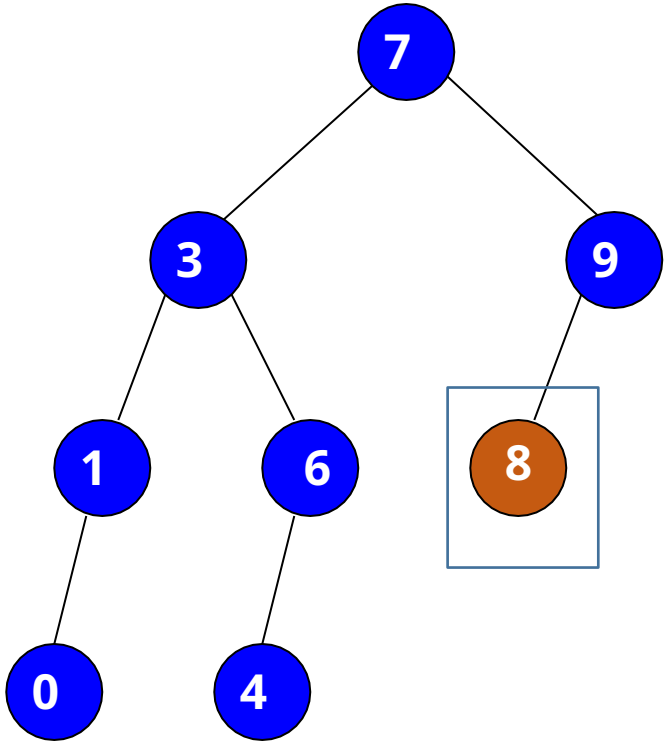
Postorder Traversal

Algorithm for Postorder:

If the tree is empty
- Return;

Else
- Postorder traverse the left subtree;
- Postorder traverse the right subtree;
- Visit the root;

Order: 0, 1, 4, 6, 3, 8



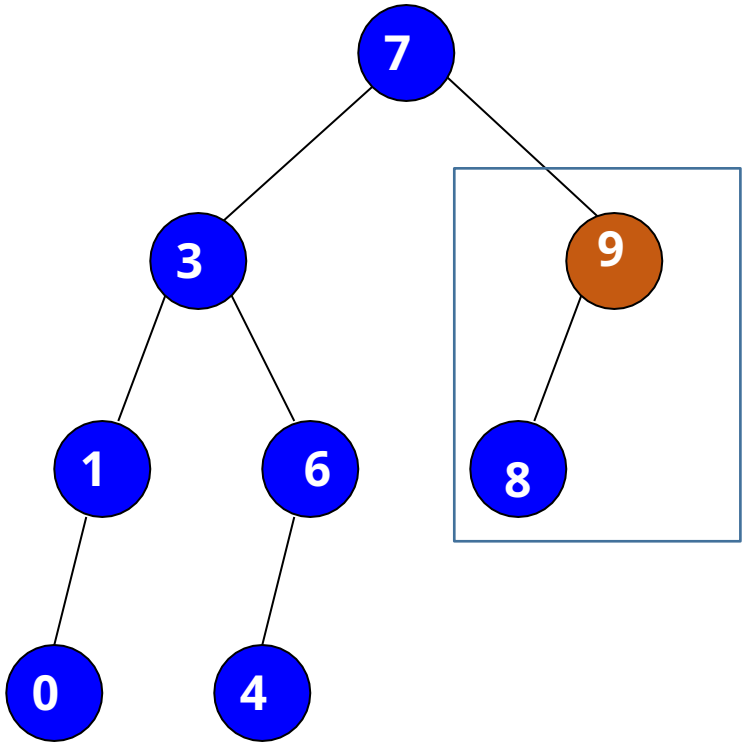
Postorder Traversal

Algorithm for Postorder:

If the tree is empty
- Return;

Else
- Postorder traverse the left subtree;
- Postorder traverse the right subtree;
- Visit the root;

Order: 0, 1, 4, 6, 3, 8, 9



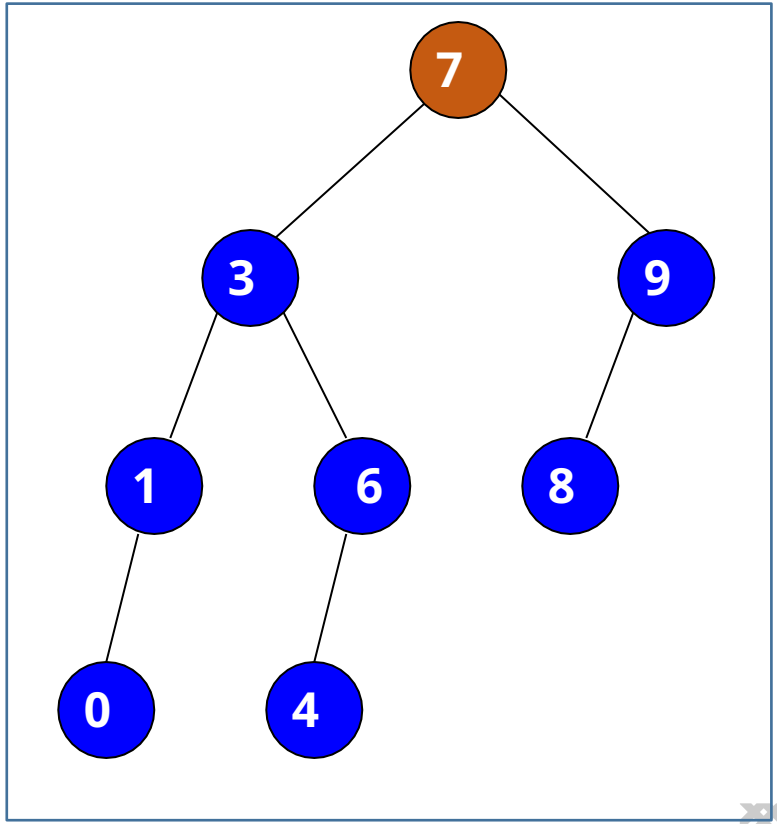
Postorder Traversal

Algorithm for Postorder:

If the tree is empty
- Return;

Else
- Postorder traverse the left subtree;
- Postorder traverse the right subtree;
- Visit the root;

Order: 0, 1, 4, 6, 3, 8, 9, 7



In-class Activity

Tree Traversal Activity