DSA LAB ASSIGNMENT 6

Binary Tree Traversal

Name : Mahesh Jagtap

Reg No. 24MCS1O17

Date : 09/09/2024

1. Consider the given binary tree, where:

   Node A is the root.

   Nodes B, W are children of A.

   Nodes X, S are children of B.

   Nodes T, C are children of W.

   Nodes E, M are children of X.

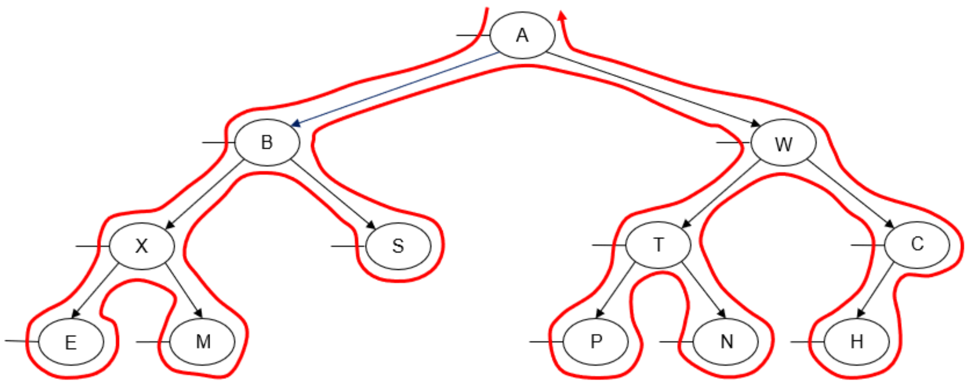
   Nodes P, N are children of T.

   Node H is the child of C.

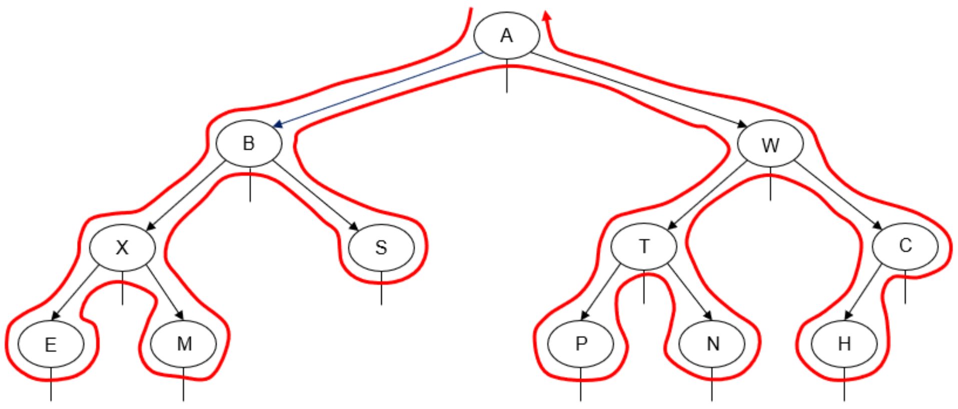
Implement the following binary tree traversals for the tree depicted in the image:

- Preorder Traversal (Root, Left, Right)
- Inorder Traversal (Left, Root, Right)
- Postorder Traversal (Left, Right, Root)

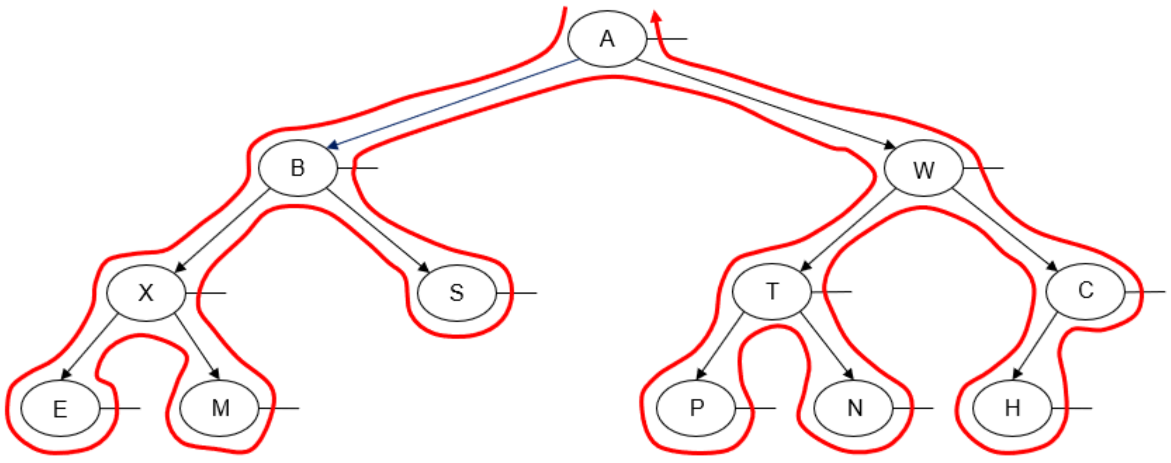
1. Preorder Traversal (Root, Left, Right)



2. Inorder Traversal (Left, Root, Right)



3. Postorder Traversal (Left, Right, Root)

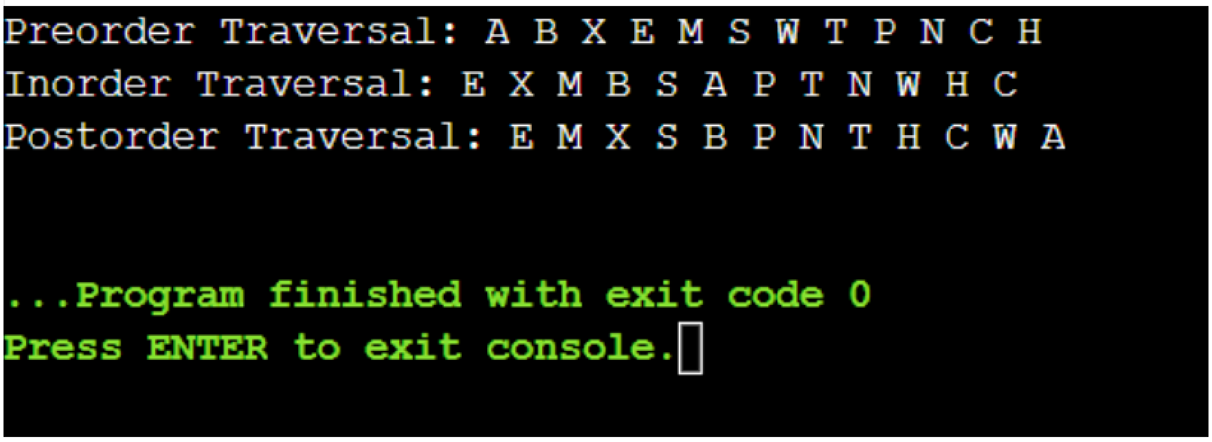


```
#include <iostream>

using namespace std;

struct Node {

   char data;

   Node* left;

   Node* right;

   Node(char val) {

      data = val;

      left = right = nullptr;

   }

};
```

```
// Preorder Traversal (Root, Left, Right)
void preorderTraversal(Node* root) {
  if (root == nullptr) return;
  cout << root->data << " ";  // Visit root
  preorderTraversal(root->left);  // Traverse left subtree
  preorderTraversal(root->right); // Traverse right subtree
}

// Inorder Traversal (Left, Root, Right)
void inorderTraversal(Node* root) {
  if (root == nullptr) return;
  inorderTraversal(root->left);  // Traverse left subtree
  cout << root->data << " ";  // Visit root
  inorderTraversal(root->right); // Traverse right subtree
}

// Postorder Traversal (Left, Right, Root)
void postorderTraversal(Node* root) {
  if (root == nullptr) return;
  postorderTraversal(root->left);  // Traverse left subtree
  postorderTraversal(root->right); // Traverse right subtree
  cout << root->data << " ";  // Visit root
}

int main() {
  // Constructing the binary tree as per the given structure
  Node* A = new Node('A');
  Node* B = new Node('B');
  Node* W = new Node('W');
  Node* X = new Node('X');
```

```
Node* S = new Node('S');
Node* T = new Node('T');
Node* C = new Node('C');
Node* E = new Node('E');
Node* M = new Node('M');
Node* P = new Node('P');
Node* N = new Node('N');
Node* H = new Node('H');
A->left = B;
A->right = W;

B->left = X;
B->right = S;

W->left = T;
W->right = C;

X->left = E;
X->right = M;

T->left = P;
T->right = N;

C->left = H;

// Display the traversals
cout << "Preorder Traversal: ";
preorderTraversal(A);
cout << endl;
 cout << "Inorder Traversal: ";
inorderTraversal(A);
```

```
    cout << endl;
    cout << "Postorder Traversal: ";
    postorderTraversal(A);
    cout << endl;
    return 0;
}
```



```
Preorder Traversal: A B X E M S W T P N C H
Inorder Traversal: E X M B S A P T N W H C
Postorder Traversal: E M X S B P N T H C W A


...Program finished with exit code 0
Press ENTER to exit console.
```