#include #include // Structure of a tree node struct Node { int data; struct Node* left; struct Node* right; }; // Function to create a new node struct Node* createNode(int data) { struct Node* newNode = (struct Node*)malloc(sizeof(struct Node)); newNode->data = data; newNode->left = NULL; newNode->right = NULL; return newNode; } // Function to insert a node into the binary tree struct Node* insertNode(struct Node* root, int data) { struct Node* newNode = createNode(data); if (root == NULL) return newNode; struct Node* current = root; struct Node* parent = NULL; while (current != NULL) { parent = current; if (data < current->data) current = current->left; else current = current->right; } if (data < parent->data) parent->left = newNode; else parent->right = newNode; return root; } // Inorder traversal (Left, Data, Right) void inorderTraversal(struct Node* root) { if (root == NULL) return; inorderTraversal(root->left); printf("%d ", root->data); inorderTraversal(root->right); } // Preorder traversal (Data, Left, Right) void preorderTraversal(struct Node* root) { if (root == NULL) return; printf("%d ", root->data); preorderTraversal(root->left); preorderTraversal(root->right); } // Postorder traversal (Left, Right, Data) void postorderTraversal(struct Node* root) { if (root == NULL) return; postorderTraversal(root->left); postorderTraversal(root->right); printf("%d ", root->data); } int main() { struct Node* root = NULL; int choice, data; while (1) { printf("\nMenu:\n"); printf("1. Insert a node\n"); printf("2. Inorder traversal\n"); printf("3. Preorder traversal\n"); printf("4. Postorder traversal\n"); printf("5. Exit\n"); printf("\n Enter your choice: "); scanf("%d", &choice); switch (choice) { case 1: printf("Enter data to insert: "); scanf("%d", &data); root = insertNode(root, data); break; case 2: printf("\n Inorder traversal: "); inorderTraversal(root); printf("\n"); break; case 3: printf("\n Preorder traversal: "); preorderTraversal(root); printf("\n"); break; case 4: printf("\n Postorder traversal: "); postorderTraversal(root); printf("\n"); break; case 5: exit(0); default: printf("Invalid choice\n"); } } return 0; }