#include <stdio.h>

#include <stdlib.h>

#define MAX\_SIZE 100

// Define a structure for a binary tree node

struct TreeNode {

int data;

};

// Function to initialize a binary tree

void initTree(struct TreeNode tree[], int size) {

for (int i = 0; i < size; i++) {

tree[i].data = -1; // Initialize all nodes with -1 to indicate empty

}

}

// Function to insert a value into the binary tree

void insert(struct TreeNode tree[], int size, int value) {

for (int i = 0; i < size; i++) {

if (tree[i].data == -1) {

tree[i].data = value;

break;

}

}

}

// Function to display the binary tree in an in-order traversal

void displayInOrder(struct TreeNode tree[], int size, int index) {

if (index < size && tree[index].data != -1) {

displayInOrder(tree, size, 2 \* index + 1);

printf("%d ", tree[index].data);

displayInOrder(tree, size, 2 \* index + 2);

}

}

// Function to display the binary tree in a preorder traversal

void displayPreOrder(struct TreeNode tree[], int size, int index) {

if (index < size && tree[index].data != -1) {

printf("%d ", tree[index].data);

displayPreOrder(tree, size, 2 \* index + 1);

displayPreOrder(tree, size, 2 \* index + 2);

}

}

// Function to display the binary tree in a postorder traversal

void displayPostOrder(struct TreeNode tree[], int size, int index) {

if (index < size && tree[index].data != -1) {

displayPostOrder(tree, size, 2 \* index + 1);

displayPostOrder(tree, size, 2 \* index + 2);

printf("%d ", tree[index].data);

}

}

int main() {

int size;

printf("Enter the number of nodes: ");

scanf("%d", &size);

struct TreeNode binaryTree[MAX\_SIZE];

initTree(binaryTree, size);

printf("Enter the values for the nodes:\n");

for (int i = 0; i < size; i++) {

int value;

scanf("%d", &value);

insert(binaryTree, size, value);

}

printf("Binary Tree in in-order traversal: ");

displayInOrder(binaryTree, size, 0);

printf("\n");

printf("Binary Tree in preorder traversal: ");

displayPreOrder(binaryTree, size, 0);

printf("\n");

printf("Binary Tree in postorder traversal: ");

displayPostOrder(binaryTree, size, 0);

printf("\n");

return 0;

}