```
#include <stdio.h>
#define MAX_SIZE 15  // Maximum size of the binary tree array
int tree[MAX_SIZE];  // Array to store the binary tree
// Function to initialize the tree with -1 (indicating empty nodes)
void initializeTree() {
    for (int i = 0; i < MAX_SIZE; i++) {
        tree[i] = -1;  // -1 represents an empty node
    }
}
// Function to insert a value at a given index
void insertNode(int value, int index) {
    if (index >= MAX_SIZE) {
        printf("Index out of bounds!\n");
        return;
    }
    if (tree[index] != -1) {
        printf("Node already occupied at index %d\n", index);
        return;
    }
    tree[index] = value;
}
// Function to display the binary tree as an array
void displayTree() {
    printf("Binary Tree (Array Representation):\n");
    for (int i = 0; i < MAX_SIZE; i++) {
```

```
        if (tree[i] == -1)
            printf("_ ");  // Empty node
        else
            printf("%d ", tree[i]);
    }
    printf("\n");
}
int main() {
    initializeTree();
    // Insert nodes
    insertNode(1, 0);  // Root node
    insertNode(2, 1);  // Left child of root
    insertNode(3, 2);  // Right child of root
    insertNode(4, 3);  // Left child of node at index 1
    insertNode(5, 4);  // Right child of node at index 1
    insertNode(6, 5);  // Left child of node at index 2
    insertNode(7, 6);  // Right child of node at index 2
    // Display tree
    displayTree();
    return 0;
}
```