

```

#include <stdio.h>

#define MAX_SIZE 15 // Maximum size of the binary tree array

int tree[MAX_SIZE];

void inorderTraversal(int index) {
    // If the index is out of bounds or the node is empty, return
    if (index >= MAX_SIZE || tree[index] == -1) {
        return;
    }
    // Traverse the left subtree
    inorderTraversal(2 * index + 1);
    // Visit the current node (print its value)
    printf("%d ", tree[index]);
    // Traverse the right subtree
    inorderTraversal(2 * index + 2);
}

int main() {
    // Initialize all elements to -1 (empty nodes)
    for (int i = 0; i < MAX_SIZE; i++) {
        tree[i] = -1;
    }

    // Manually insert values in the binary tree array (in level-order)
    tree[0] = 1; // Root node
    tree[1] = 2; // Left child of root
}

```

```
tree[2] = 3; // Right child of root
tree[3] = 4; // Left child of node 2
tree[4] = 5; // Right child of node 2
tree[5] = 6; // Left child of node 3
tree[6] = 7; // Right child of node 3

// Perform in-order traversal and display the tree
printf("In-order Traversal of the Binary Tree:\n");
inorderTraversal(0); // Start from the root (index 0)

return 0;
}
```