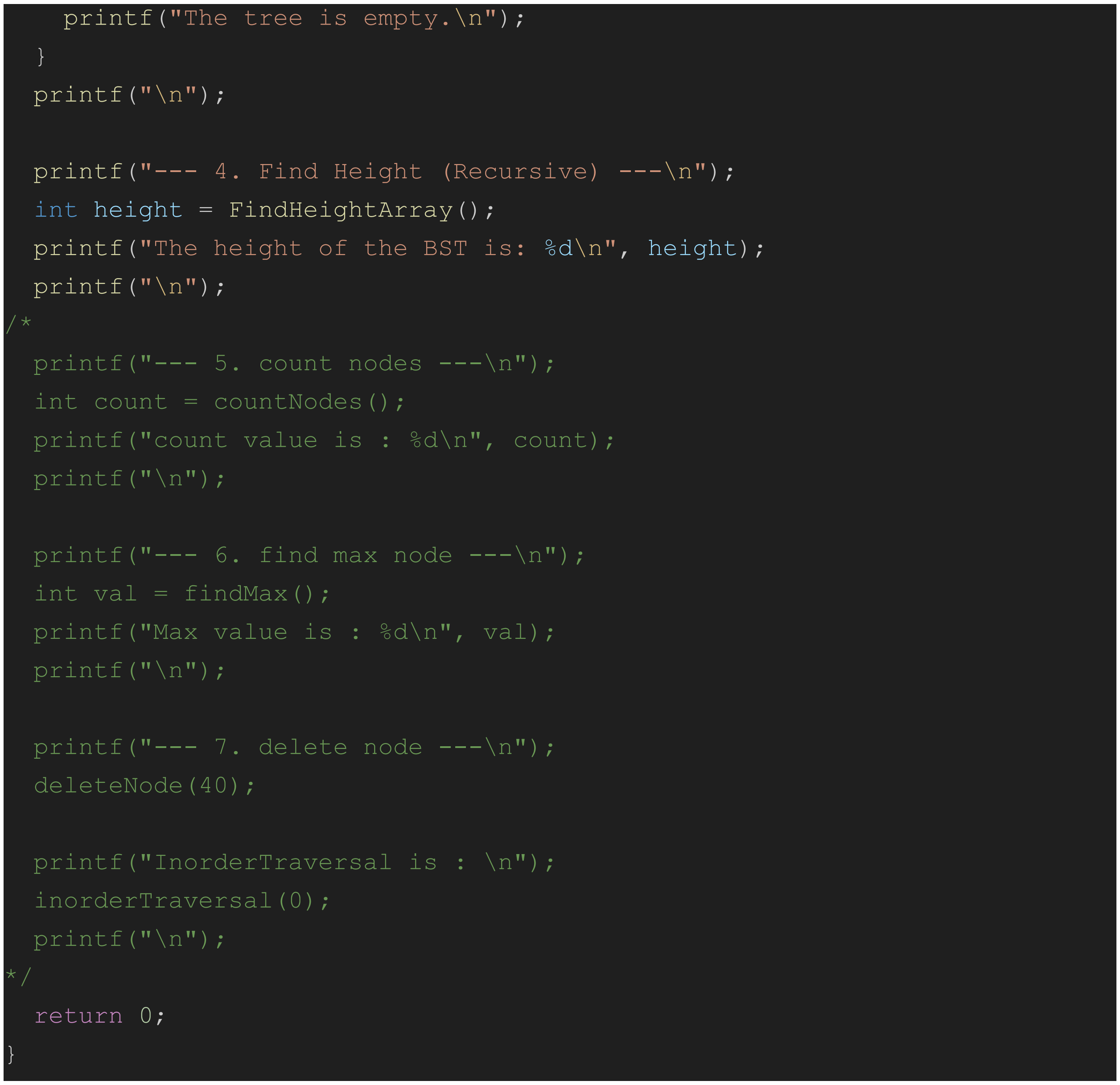
Code:

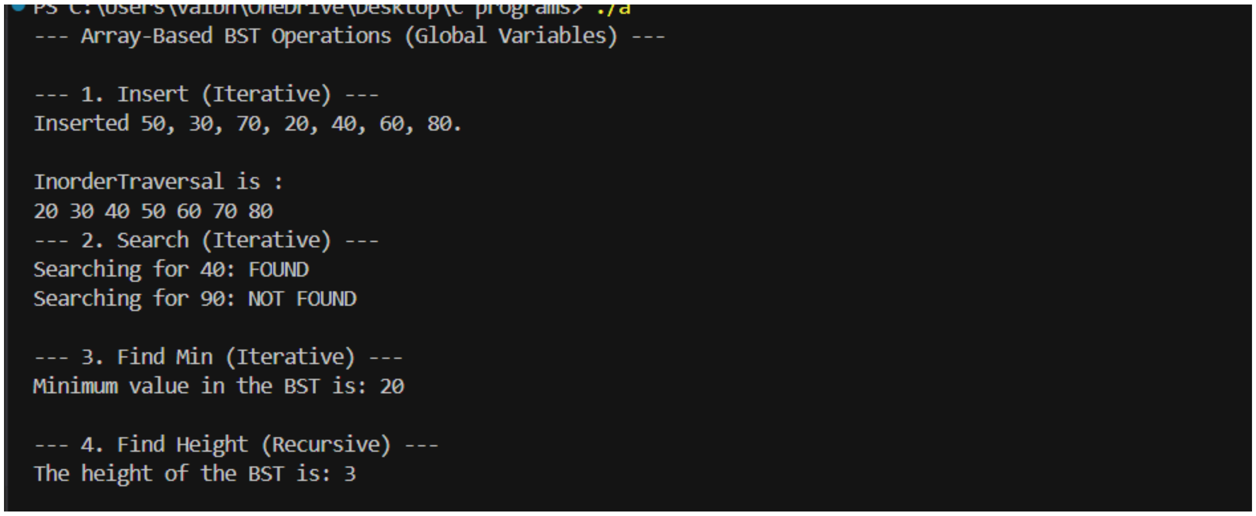
```c
#include<stdio.h>
#include<stdbool.h>
#define MAX 100
#define EMPTY -1
int bst[MAX];
int size=0;
int max(int a,int b){return (a>b)?a:b;}
int leftchild(int i){return 2*i+1;}
int rightchild(int i){return 2*i+2;}
void InorderTraversal(int index)
{
    if(index>=MAX || bst[index]==EMPTY) return;
    InorderTraversal(leftchild(index));
    printf("%d ",bst[index]);
    InorderTraversal(rightchild(index));
}
void InsertArray(int data)
{
  int index=0;
  if(bst[0]==EMPTY)
  {
    bst[0]=data;
    return;
  }
  while(index<MAX)
  {
    int next_index;
    if(data<=bst[index]) next_index=2*index+1;
    else next_index=2*index+2;
    if(next_index>=MAX)
    {
      printf("Error:Cannot insert %d,Array is full along this path
(MAX=%d)\n",data,MAX);
      return;
    }
    if(bst[next_index]==EMPTY)
    {
      bst[next_index]=data;
      if(next_index>=size) size=next_index+1;
      return;
    }
    index=next_index;

  }
}
bool SearchArray(int data)
{int index=0;
  while(index<MAX)
  {
    if(bst[index]==EMPTY) return false;
    if(bst[index]==data) return true;
    else if(data<bst[index]) index=2*index+1;
    else index=2*index+2;

  }
return false;
}
int FindMinArray()
{
  if(bst[0]==EMPTY) return EMPTY;
  int index=0;
  while(index<MAX)
  {
    int left_index=2*index+1;
    if(left_index>=MAX || bst[left_index]==EMPTY) return bst[index];
    index=left_index;
  }
  return EMPTY;
}
int FindHeightRecursive(int index)
{
  if(index>MAX) return -1;
  if(bst[index]==EMPTY) return 0;
  int leftHeight=FindHeightRecursive(2*index+1);
  int rightHeight=FindHeightRecursive(2*index+2);
  return max(leftHeight,rightHeight)+1;
}
```

```c
int FindHeightArray() { return FindHeightRecursive(0);}
void deleteNode(int val)
{
  if(size==0) return;

}

int main() {
  printf("--- Array-Based BST Operations (Global Variables) ---\n\n");
  for (int i = 0; i < MAX; i++) {
    bst[i] = EMPTY;
  }
  printf("--- 1. Insert (Iterative) ---\n");
  InsertArray(50);
  InsertArray(30);
  InsertArray(70);
  InsertArray(20);
  InsertArray(40);
  InsertArray(60);
  InsertArray(80);

  printf("Inserted 50, 30, 70, 20, 40, 60, 80.\n\n");

  printf("InorderTraversal is : \n");
  InorderTraversal(0);
  printf("\n");

  printf("--- 2. Search (Iterative) ---\n");
  int searchVal1 = 40;
  int searchVal2 = 90;

  printf("Searching for %d: %s\n", searchVal1,
         SearchArray(searchVal1) ? "FOUND" : "NOT FOUND");
  printf("Searching for %d: %s\n\n", searchVal2,
         SearchArray(searchVal2) ? "FOUND" : "NOT FOUND");
  printf("--- 3. Find Min (Iterative) ---\n");
  int minVal = FindMinArray();
  if (minVal != EMPTY) {
    printf("Minimum value in the BST is: %d\n", minVal);
  } else {
```



```c
    printf("The tree is empty.\n");
  }
  printf("\n");

  printf("--- 4. Find Height (Recursive) ---\n");
  int height = FindHeightArray();
  printf("The height of the BST is: %d\n", height);
  printf("\n");
/*
  printf("--- 5. count nodes ---\n");
  int count = countNodes();
  printf("count value is : %d\n", count);
  printf("\n");

  printf("--- 6. find max node ---\n");
  int val = findMax();
  printf("Max value is : %d\n", val);
  printf("\n");

  printf("--- 7. delete node ---\n");
  deleteNode(40);

  printf("InorderTraversal is : \n");
  inorderTraversal(0);
  printf("\n");
*/
  return 0;
}
```

Output:



```
PS C:\Users\vaibh\OneDrive\Desktop\C programs> ./a
--- Array-Based BST Operations (Global Variables) ---

--- 1. Insert (Iterative) ---
Inserted 50, 30, 70, 20, 40, 60, 80.

InorderTraversal is :
20 30 40 50 60 70 80
--- 2. Search (Iterative) ---
Searching for 40: FOUND
Searching for 90: NOT FOUND

--- 3. Find Min (Iterative) ---
Minimum value in the BST is: 20

--- 4. Find Height (Recursive) ---
The height of the BST is: 3
```