

Binary Search Tree- basic operations traversal and applications

1. count number of nodes

```
2. #include <stdio.h>
3. #include <stdlib.h>
4. struct node {
5.     int val;
6.     struct node* left;
7.     struct node* right;
8. };
9. struct node* newNode(int val) {
10.     struct node* temp = (struct node*) malloc(sizeof(struct node));
11.     temp->val = val;
12.     temp->left = NULL;
13.     temp->right = NULL;
14.     return temp;
15. }
16. struct node* insert(struct node* root, int val) {
17.     if (root == NULL) return newNode(val);
18.     if (val < root->val)
19.         root->left = insert(root->left, val);
20.     else if (val > root->val)
21.         root->right = insert(root->right, val);
22.     return root;
23. }
24. int countNodes(struct node* root) {
25.     if (root == NULL)
26.         return 0;
27.     return countNodes(root->left) + countNodes(root->right) + 1;
28. }
29. int main() {
30.     struct node* root = NULL;
31.     int n,i,val;
32.     printf("Enter the number of nodes:");
33.     scanf("%d",&n);
34.     printf("Enter the values:");
35.     for(i=0;i<n;i++){
36.         scanf("%d",&val);
37.         root=insert(root,val);
38.     }
39.     int nodeCount = countNodes(root);
40.     printf("Number of nodes: %d\n", nodeCount);
41.     return 0;
42. }
```

Output

```
/tmp/HLNGjYHEJN.o
Enter the number of nodes:4
Enter the values:45
56
12
23
Number of nodes: 4
```

2. compute height of tree

```
#include <stdio.h>
#include <stdlib.h>
struct node {
    int val;
    struct node* left;
    struct node* right;
};
struct node* newNode(int val) {
    struct node* temp = (struct node*) malloc(sizeof(struct node));
    temp->val = val;
    temp->left = NULL;
    temp->right = NULL;
    return temp;
}
struct node* insert(struct node* root, int val) {
    if (root == NULL) return newNode(val);
    if (val < root->val)
        root->left = insert(root->left, val);
    else if (val > root->val)
        root->right = insert(root->right, val);
    return root;
}
int height(struct node* root) {
    if (root == NULL)
        return 0;
    int leftHeight = height(root->left);
    int rightHeight = height(root->right);
    if (leftHeight > rightHeight)
        return leftHeight + 1;
    else
        return rightHeight + 1;
}
```

```
int main() {
    struct node* root = NULL;
    int n,i,val;
    printf("Enter the number of nodes:");
    scanf("%d",&n);
    printf("Enter the values:");
    for(i=0;i<n;i++){
        scanf("%d",&val);
        root=insert(root,val);
    }
    int treeHeight = height(root);
    printf("Height of tree: %d\n", treeHeight);
    return 0;
}
```

Output

/tmp/HLNGjYHEJN.o

Enter the number of nodes:5

Enter the values:1

2

5

7

8

Height of tree: 5