

Laboratory 9**1. Questions**

1. Write a C program to construct a binary search tree and perform the Preorder, post order and in order traversal.
2. Write a C program to implement a linked list to construct a tree and count the number of leaves in a tree.

2. Algorithm**1. Algorithm for program to construct a binary search tree and perform the Preorder, post order and in order traversal.****Inorder Traversal**

Algorithm Inorder(tree)

1. Traverse the left subtree, i.e., call Inorder(left-subtree)
2. Visit the root.
3. Traverse the right subtree, i.e., call Inorder(right-subtree)

Preorder Traversal

Algorithm Preorder(tree)

1. Visit the root.
2. Traverse the left subtree, i.e., call Preorder(left-subtree)
3. Traverse the right subtree, i.e., call Preorder(right-subtree)

Postorder Traversal

Algorithm Postorder(tree)

1. Traverse the left subtree, i.e., call Postorder(left-subtree)
2. Traverse the right subtree, i.e., call Postorder(right-subtree)
3. Visit the root.

2. Algorithm to implement a linked list to construct a tree and count the number of leaves in a tree.**Algorithm to get the leaf node count.**

getLeafCount(node)

- 1) If node is NULL then return 0.
- 2) Else If left and right child nodes are NULL return 1.
- 3) Else recursively calculate leaf count of the tree using below formula.

Leaf count of a tree = Leaf count of left subtree +

Leaf count of right subtree

3. Program

```
1  // C program for different tree traversals
2  #include <stdio.h>
3  #include <stdlib.h>
4
5  /* A binary tree node has data, pointer to left child
6  and a pointer to right child */
7  struct node
8  {
9      int data;
10     struct node* left;
11     struct node* right;
12 };
13
14 /* Helper function that allocates a new node with the
15 given data and NULL left and right pointers. */
16 struct node* newNode(int data)
17 {
18     struct node* node = (struct node*)
19                         malloc(sizeof(struct node));
20     node->data = data;
21     node->left = NULL;
22     node->right = NULL;
23
24     return(node);
25 }
26
27 /* Given a binary tree, print its nodes according to the
28 "bottom-up" postorder traversal. */
29 void printPostorder(struct node* node)
30 {
31     if (node == NULL)
32         return;
33
34     // first recur on left subtree
35     printPostorder(node->left);
36
37     // then recur on right subtree
38     printPostorder(node->right);
39
40     // now deal with the node
41     printf("%d ", node->data);
42 }
43
44 /* Given a binary tree, print its nodes in inorder*/
45 void printInorder(struct node* node)
46 {
47     if (node == NULL)
48         return;
49
50     /* first recur on left child */
51     printInorder(node->left);
52
53     /* then print the data of node */
54     printf("%d ", node->data);
55
56     /* now recur on right child */
57     printInorder(node->right);
58 }
59
60 /* Given a binary tree, print its nodes in preorder*/
```

```
61 void printPreorder(struct node* node)
62 {
63     if (node == NULL)
64         return;
65
66     /* first print data of node */
67     printf("%d ", node->data);
68
69     /* then recur on left subtree */
70     printPreorder(node->left);
71
72     /* now recur on right subtree */
73     printPreorder(node->right);
74 }
75
76 /* Driver program to test above functions */
77 int main()
78 {
79     struct node *root = newNode(1);
80     root->left = newNode(2);
81     root->right = newNode(3);
82     root->left->left = newNode(4);
83     root->left->right = newNode(5);
84
85     printf("\nPreorder traversal of binary tree is \n");
86     printPreorder(root);
87
88     printf("\nInorder traversal of binary tree is \n");
89     printInorder(root);
90
91     printf("\nPostorder traversal of binary tree is \n");
92     printPostorder(root);
93
94     getchar();
95     return 0;
96 }
97 |
```

Fig 1 program to construct a binary search tree and perform the Preorder, post order and in order traversal.

```

1  // C implementation to find Leaf count of a given Binary tree
2  #include <stdio.h>
3  #include <stdlib.h>
4
5  /* A binary tree node has data, pointer to left child
6  and a pointer to right child */
7  struct node
8  {
9      int data;
10     struct node* left;
11     struct node* right;
12 };
13
14 /* Function to get the count of Leaf nodes in a binary tree*/
15 unsigned int getLeafCount(struct node* node)
16 {
17     if(node == NULL)
18         return 0;
19     if(node->left == NULL && node->right==NULL)
20         return 1;
21     else
22         return getLeafCount(node->left)+
23                getLeafCount(node->right);
24 }
25
26 /* Helper function that allocates a new node with the
27 given data and NULL left and right pointers. */
28 struct node* newNode(int data)
29 {
30     struct node* node = (struct node*)
31         malloc(sizeof(struct node));
32     node->data = data;
33
34     node->data = data;
35     node->left = NULL;
36     node->right = NULL;
37
38     return(node);
39 }
40
41 /*Driver program to test above functions*/
42 int main()
43 {
44     /*create a tree*/
45     struct node *root = newNode(1);
46     root->left  = newNode(2);
47     root->right = newNode(3);
48     root->left->left = newNode(4);
49     root->left->right = newNode(5);
50
51     /*get Leaf count of the above created tree*/
52     printf("Leaf count of the tree is %d", getLeafCount(root));
53
54     getchar();
55     return 0;
56 }

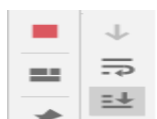
```

Fig 2 program to to implement a linked list to construct a tree and count the number of leaves in a tree.

4. Presentation of Result

```
Preorder traversal of binary tree is
1 2 4 5 3
Inorder traversal of binary tree is
4 2 5 1 3
Postorder traversal of binary tree is
4 5 2 3 1 |
```

Fig 3 Result of program to construct a binary search tree and perform the Preorder, post order and in order traversal.

A screenshot of a terminal window. On the left is a vertical toolbar with icons for a red flag, a downward arrow, a refresh/circular arrow, and a magnifying glass. To the right of the toolbar, the text "Leaf count of the tree is 3" is displayed in a monospaced font.

Leaf count of the tree is 3

Fig 4 Result of program to construct a binary search tree and perform the Preorder, post order and in order traversal.

5. Conclusion

In this lab we learnt to Write a C program to construct a binary search tree and perform the Preorder, post order and in order traversal and to Write a C program to implement a linked list to construct a tree and count the number of leaves in a tree.