

```

1: #include<stdio.h>
2: #include<malloc.h>
3:
4: struct node{
5:     int data;
6:     struct node* left;
7:     struct node* right;
8: };
9:
10: struct node* createNode(int data){
11:     struct node *n; // creating a node pointer
12:     n = (struct node *) malloc(sizeof(struct node)); // Allocating
13:     n->data = data; // Setting the data
14:     n->left = NULL; // Setting the left and right children to NULL
15:     n->right = NULL; // Setting the left and right children to NULL
16:     return n; // Finally returning the created node
17: }
18:
19: void preOrder(struct node* root){
20:     if(root!=NULL){
21:         printf("%d ", root->data);
22:         preOrder(root->left);
23:         preOrder(root->right);
24:     }
25: }
26:
27: void postOrder(struct node* root){
28:     if(root!=NULL){
29:         postOrder(root->left);
30:         postOrder(root->right);
31:         printf("%d ", root->data);
32:     }
33: }
34:
35: void inOrder(struct node* root){
36:     if(root!=NULL){
37:         inOrder(root->left);
38:         printf("%d ", root->data);
39:         inOrder(root->right);

```

```

40:     }
41: }
42:
43: int main(){
44:
45:     // Constructing the root node - Using Function (Recommended)
46:     struct node *p = createNode(4);
47:     struct node *p1 = createNode(1);
48:     struct node *p2 = createNode(6);
49:     struct node *p3 = createNode(5);
50:     struct node *p4 = createNode(2);
51:     // Finally The tree looks like this:
52:     //      4
53:     //    / \
54:     //   1   6
55:     //  / \
56:     // 5  2
57:
58:     // Linking the root node with left and right children
59:     p->left = p1;
60:     p->right = p2;
61:     p1->left = p3;
62:     p1->right = p4;
63:
64:     preOrder(p);
65:     printf("\n");
66:     postOrder(p);
67:     printf("\n");
68:     inOrder(p);
69:     return 0;
70: }
71:

```