

## CollegeWork\DataStructure\binary-tree-uisng-linekd-list.c

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  typedef struct Binary_Tree{
5      int data;
6      struct Binary_Tree *l;
7      struct Binary_Tree *r;
8  }btree;
9
10 btree *lChild(btree *t){
11     return t->l;
12 }
13 btree *rChild(btree *t){
14     return t->r;
15 }
16
17 void add(btree **t, int new_data){
18     btree *new_btree = (btree*)malloc(sizeof(btree));
19     new_btree->data = new_data;
20     new_btree->l = NULL;
21     new_btree->r = NULL;
22     *t = new_btree;
23 }
24
25 void delete(btree **t){
26     (*t)->r = NULL;
27     (*t)->l = NULL;
28     (*t) = NULL;
29 }
30
31 void display(btree *t){
32     if(!t){
33         printf("No Element Found\n");
34         return;
35     }
36     btree *l = t->l;
37     btree *r = t->r;
38     if(l == NULL){
39         if(r == NULL){
40             printf("%d (0,0)",t->data);
41         }else{
42             printf("%d (0,%d)",t->data,r->data);
43         }
44     }else if(r == NULL){
45         printf("%d (%d,0)",t->data,l->data);
46     }else{
47         printf("%d (%d,%d)",t->data,l->data,r->data);
48     }
49     printf("\n");
50     return;
51 }
```

```

52
53 void inorderTraversal(btree* root) {
54     if (root) {
55         display(root);
56         inorderTraversal(root->l);
57         inorderTraversal(root->r);
58         // display(root);
59     }
60 }
61
62 void main(){
63     btree *t1 = NULL;
64     add(&t1,1);
65     add(&t1->l,2);
66     add(&t1->r,3);
67     add(&t1->l->l,4);
68     add(&t1->r->l,5);
69     add(&t1->l->r,6);
70     add(&t1->r->r,7);
71     add(&t1->l->l->r,8);
72     add(&t1->r->r->l,9);
73     inorderTraversal(t1);
74 }
75
76 /*
77 OUTPUT
78 PS S:\Workspace\CollegeWork\DataStructure> gcc .\binary-tree-uisng-linekd-list.c
79 PS S:\Workspace\CollegeWork\DataStructure> ./a
80 1 (2,3)
81 2 (4,6)
82 4 (0,8)
83 8 (0,0)
84 6 (0,0)
85 3 (5,7)
86 5 (0,0)
87 7 (9,0)
88 9 (0,0)
89 PS S:\Workspace\CollegeWork\DataStructure>
90 */

```