binary-tree-using-linked-list.cpp

```
//implement binary tree using linked ilst.
#include <stdio.h>
#include <stdlib.h>

typedef struct Binary_Tree{
   int data;
   struct Binary_Tree *l;
   struct Binary_Tree *r;
}btree;

btree *lChild(btree *t){
   return t->l;
}
btree *rChild(btree *t){
   return t->r;
}

void add(btree **t, int new_data){
   btree *new_btree = (btree*)malloc(sizeof(btree));
   new_btree->data = new_data;
   new_btree->l = NULL;
   new_btree->r = NULL;
   *t = new_btree;
}

void delete(btree **t){
   (*t)->r = NULL;
   (*t)->l = NULL;
   (*t) = NULL;
}

void display(btree *t){
   printf("%d (",t->data);
   if(t->l){
      printf("%d,",t->l->data);
   }else{
      printf("0,");
   }
   if(t->r){
      printf("%d)",t->r->data);
   }else{
      printf("0)");
   }
   printf("\n");
}

void preorderTraversal(btree* root) {
   if (root) {
      display(root);
      preorderTraversal(root->l);
      preorderTraversal(root->r);
      // display(root);
   }
}
```

```
void main(){
   btree *t1 = NULL;
   add(&t1,1);
   add(&t1->l,2);
   add(&t1->r,3);
   add(&t1->l->l,4);
   add(&t1->r->l,5);
   add(&t1->l->r,6);
   add(&t1->r->r,7);
   add(&t1->l->l->r,8);
   add(&t1->r->r->l,9);
   preorderTraversal(t1);
}
```

OUTPUT

```
PS S:\WorkSpace\CollegeWork\DataStructure> gcc .\binary-tree-uisng-linekd-list.c
PS S:\WorkSpace\CollegeWork\DataStructure> ./a
1 (2,3)
2 (4,6)
4 (0,8)
8 (0,0)
6 (0,0)
3 (5,7)
5 (0,0)
7 (9,0)
9 (0,0)
PS S:\WorkSpace\CollegeWork\DataStructure>
```