Lab8 Binary Search Tree

1. Main program

1) Menu 구성: **(1. Insert, 2. Delete, 3.Search, 4.Print 5. Traverse 6. Leaf 7. Quit)**

2) For each command;

- **Insert:** "Enter number to insert: " **insert_tree**(Num)
- **Delete:** If (!tree_empty()) "Enter number to delete" **delete_tree**(Num)
  else "**Tree is empty"**
- **Search:** if (!tree_empty()) "Enter number to search: " temp=**search_tree**(root, Num)
  if (temp==NULL) "NOT found" else " number is found"
  else "Tree is empty"
- **Traverse:** if (!tree_empty()) **inrder/preorder/postorder();** else "Tree is empty"
- **Leaf** (node) : // if tree is empty then return count 0;
  if (node->left == NULL) &&(node->right == NULL) count++;
  else count = leaf(node->left) + leaf(node->right);
- **Print:** **Draw_tree()** // lab7 의 drawtree 함수 이용할 수 있음

3. 알고리즘 (강의노트 참조)

Delete ➔ **FINDMAX 알고리즘 사용할 것.**

4. 테스트 절차(예):

1) **Delete** : Tree is empty

2) **Traverse** : Tree is empty

3) **Insert :** (30 40 20 10 50) 순서대로 insert 후, DRAWTRE 로 트리를 보일것

4) Leaf: 2

5) Delete test:

- Single 노드 테스트→ **delete** 20, DRAWTREE 로 트리를 보일것
- Leaf 노드 테스트 → **delete** 50, DRAWTREE 로 트리를 보일것
- 양쪽노드 테스트 → **delete** 30, DRAWTREE 로 트리를 보일것

6) **Search: 30** "Not Found" **Search: 10** "Found"

7) **Traverse**:

- inorder: 10 40 postorder: 40 10 preorder: 10 40