Today - Lecture 9 - CS163

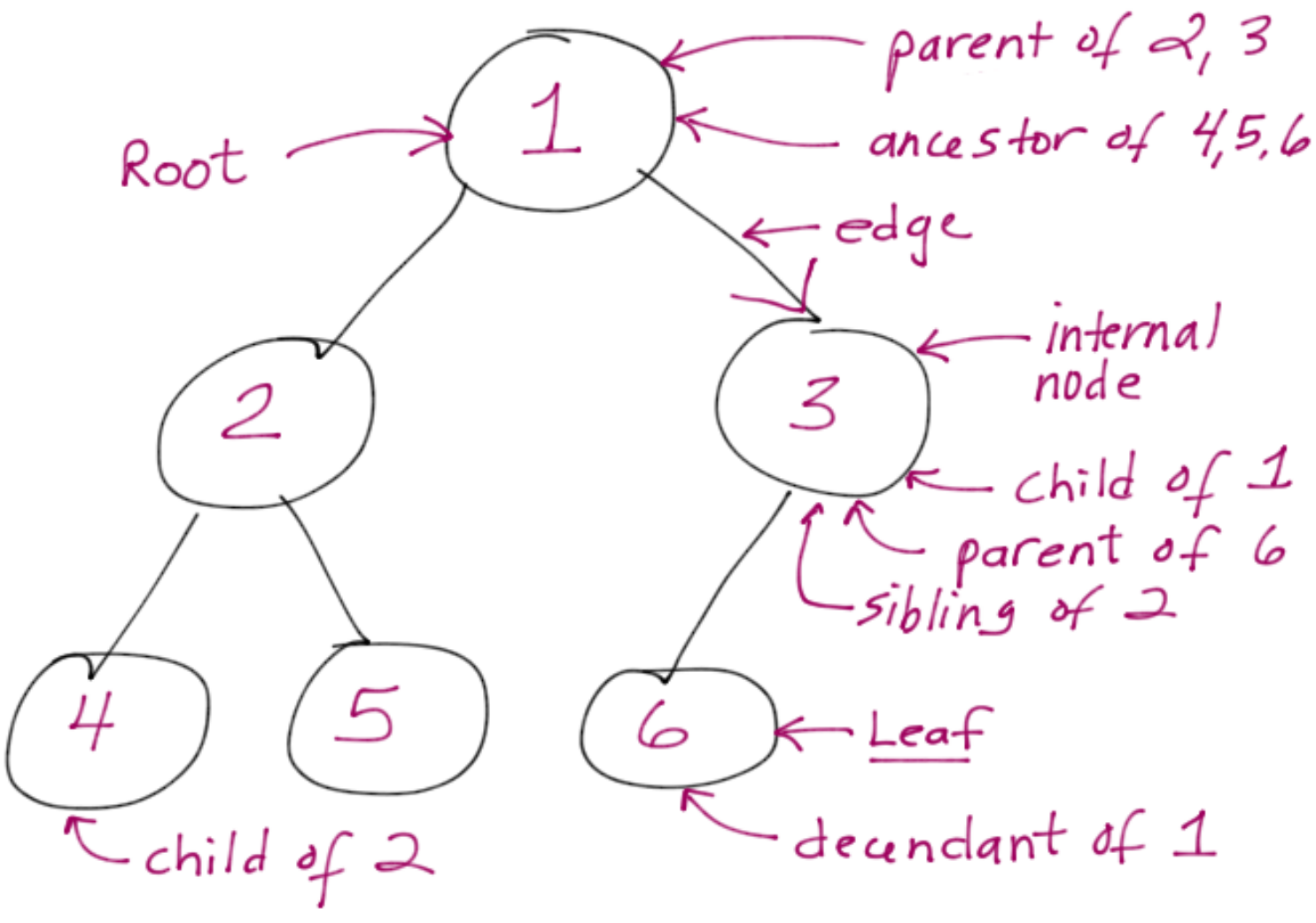
1) Topic #8 - Table ADT's implemented using trees

   - terminology
   - binary search trees
   - traversal algorithms

2) Next: Remember how recursive solutions can be applied

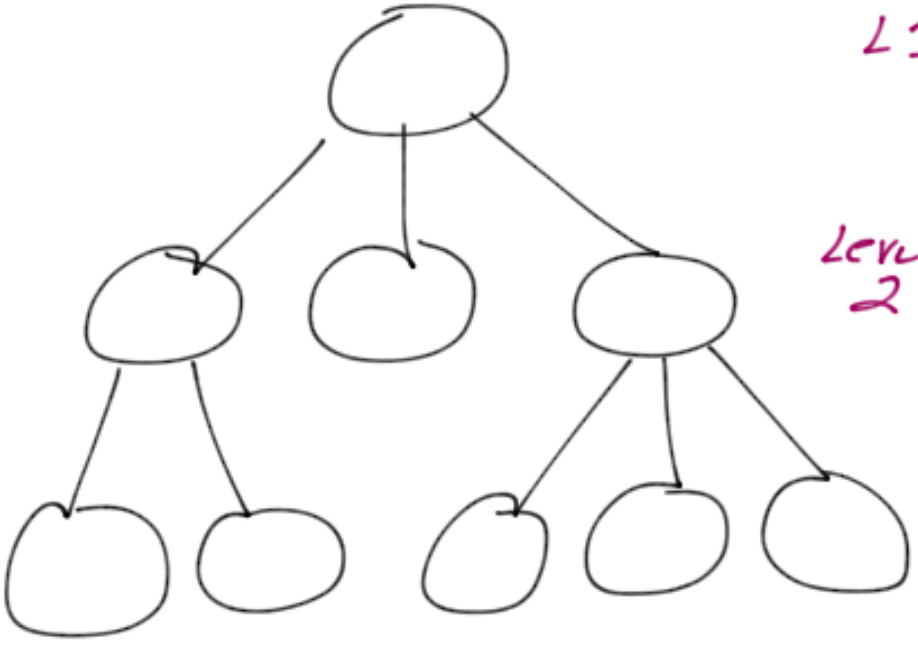
3) Make sure to Practice!

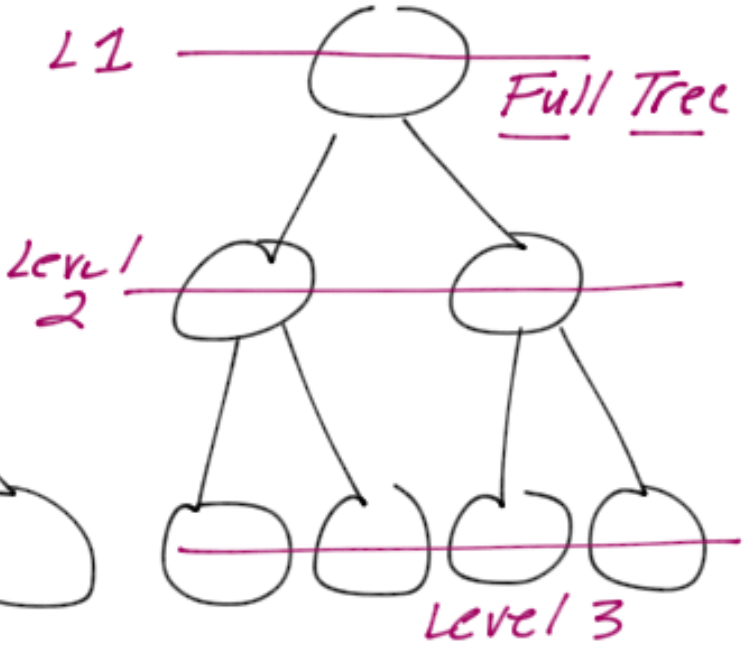
Review Terminology

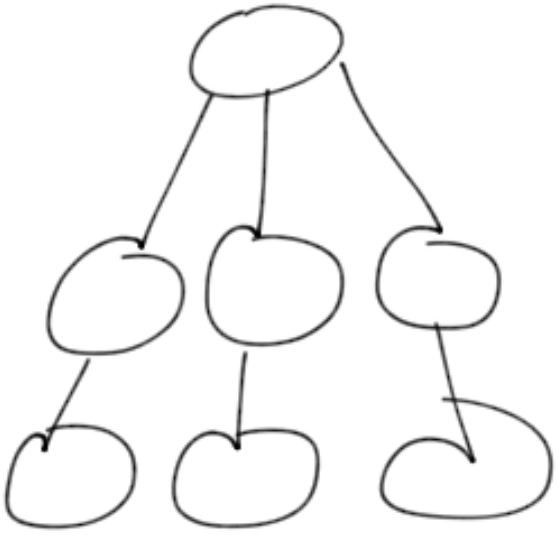


This tree represents a binary tree

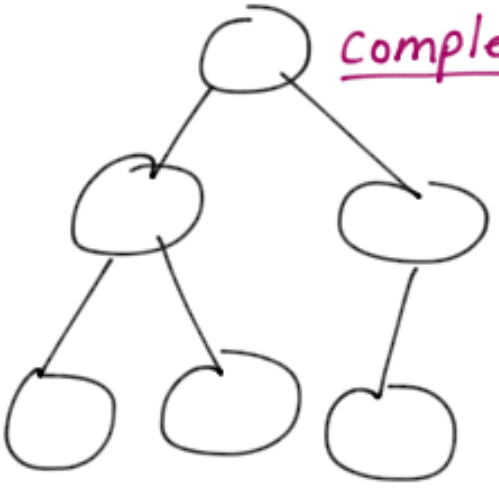
What type of Trees?







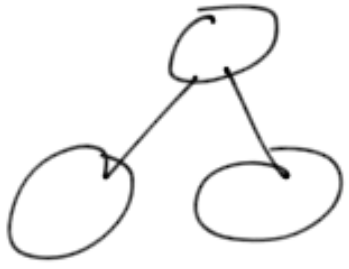
NOT a binary tree



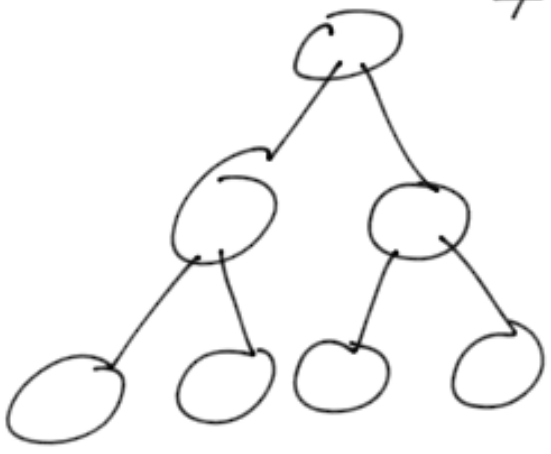
Binary tree

Full tree

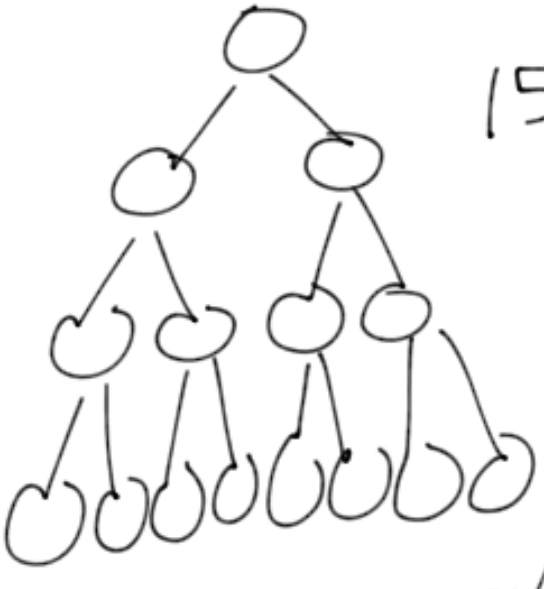
3



7

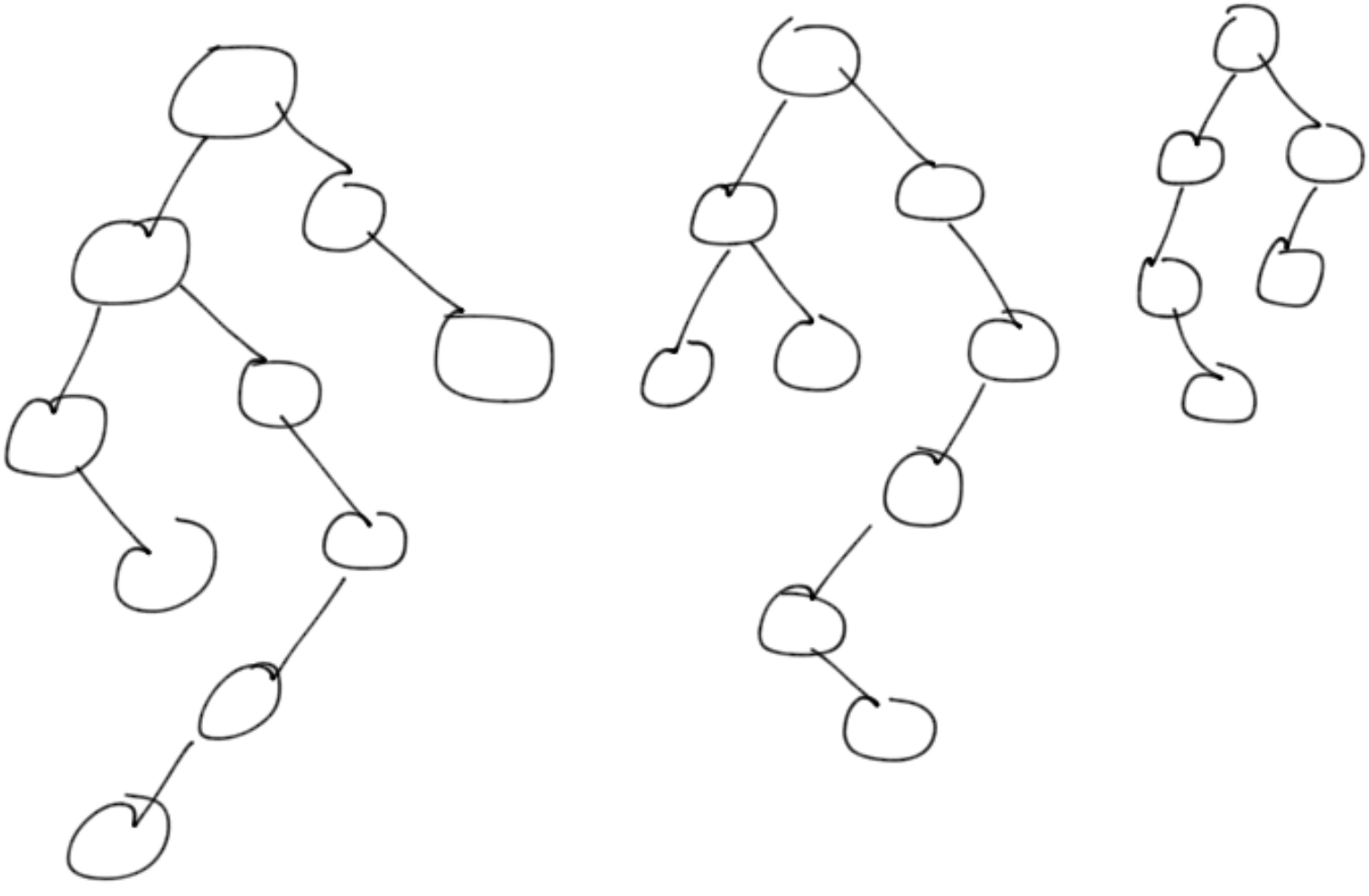


15

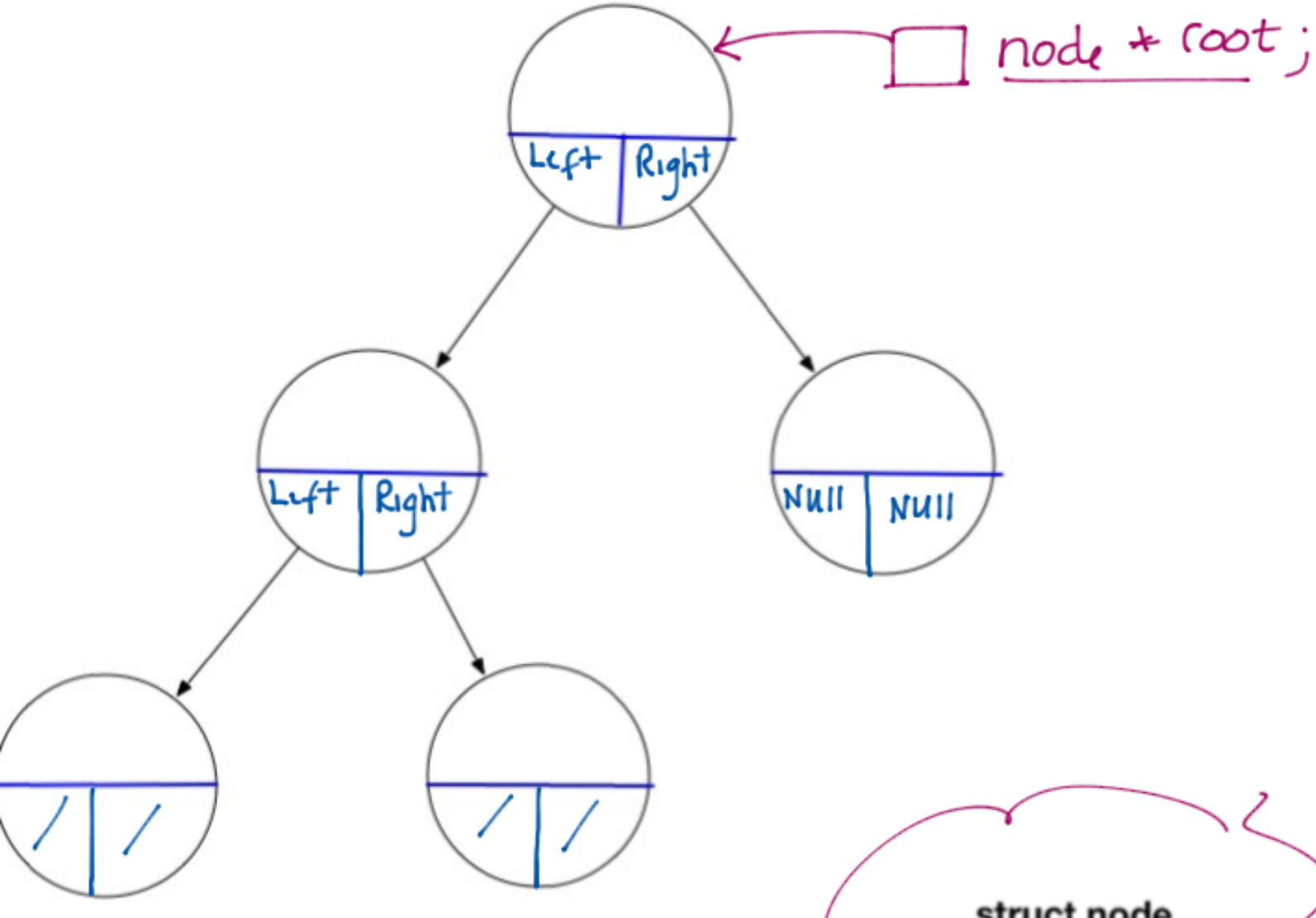


Height of a Tree –

– is the **LONGEST** path from Root to leaf



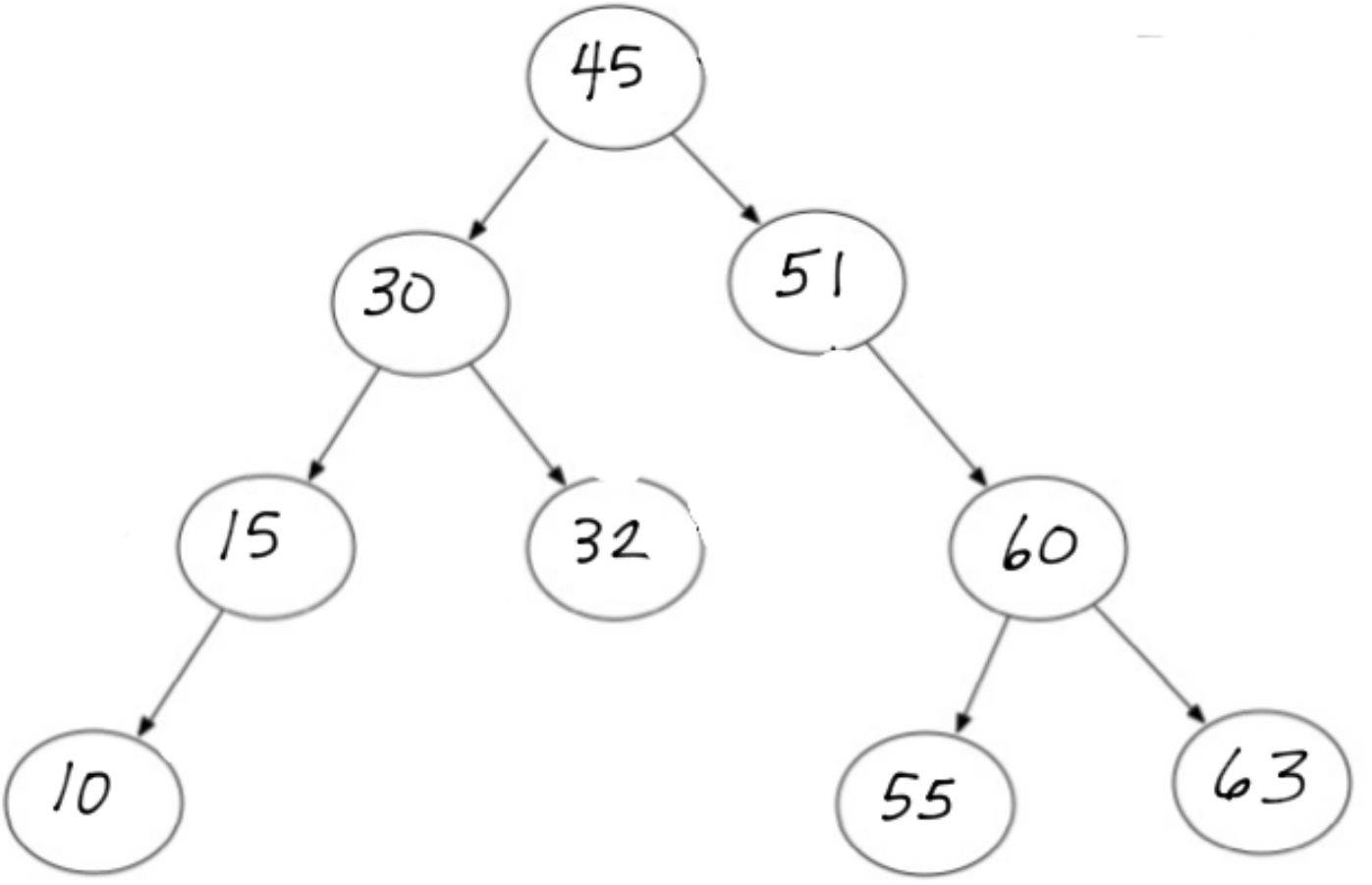
Implemented as a non-linear linked list




```
struct node
{
    student peer;
    node * left;
    node * right;
};
```

Traversal Algorithms (BST)

— INORDER Traversal —

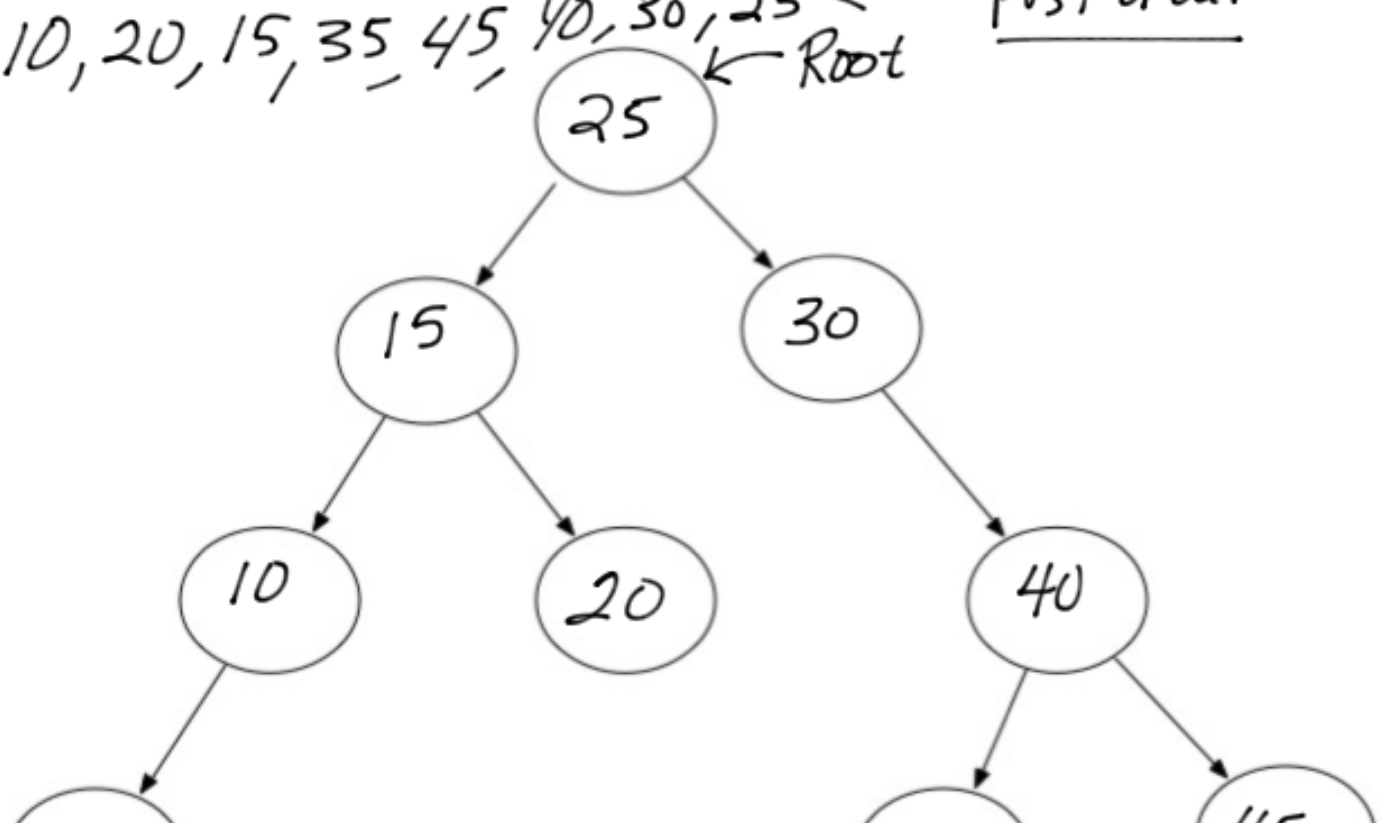



```
void traverse(node * root)
{
  if (root)
  {
     traverse(root->left);
     root->peer.display(); //cout <<root->data
     traverse(root->right);
  }
```

Pre Order & Post Order Traversal

25, 15, 10, 5, 20, 30, 40, 35, 45 ← Pre Order

5, 10, 20, 15, 35, 45, 40, 30, 25 ← Post order




```
void traverse(node * root)
{
  if (root)
  {
     traverse(root->left);
     traverse(root->right);
     root->peer.display(); //cout <<root->data
  }
}
```

Move the display for pre order!

Insertion Algorithm of a BST

binary search tree

* Always inserts at a Leaf!
* Commonly implemented using recursion

32 45 10 15 30 55 20 41 5

5 10 15 20 30 32 41 45 55