DFS:

void searchDFS (Node root){

if (root == null) return null;

visit(root);

root.visit == true;

for (Node n:root.adjacent){

if (n.visited == false){

searchDFS(n);

}

}

}

BFS:

void searchBFS(Node root){

Queue q = new Queue();

root.marked = true;

q.enqueue(root); // add to the end of queue

while (!q.isEmpty()){

Node r = q.dequeue(); // Remove from the front of the queue

visit(r);

for(Node n:r.adjacent){

if (n.marked == false){

n.marked = true;

q.enqueue(n);

}

}

}

}

Balanced Tree

1. The left and right subtrees' heights differ by at most one, AND
2. The left subtree is balanced, AND
3. The right subtree is balanced