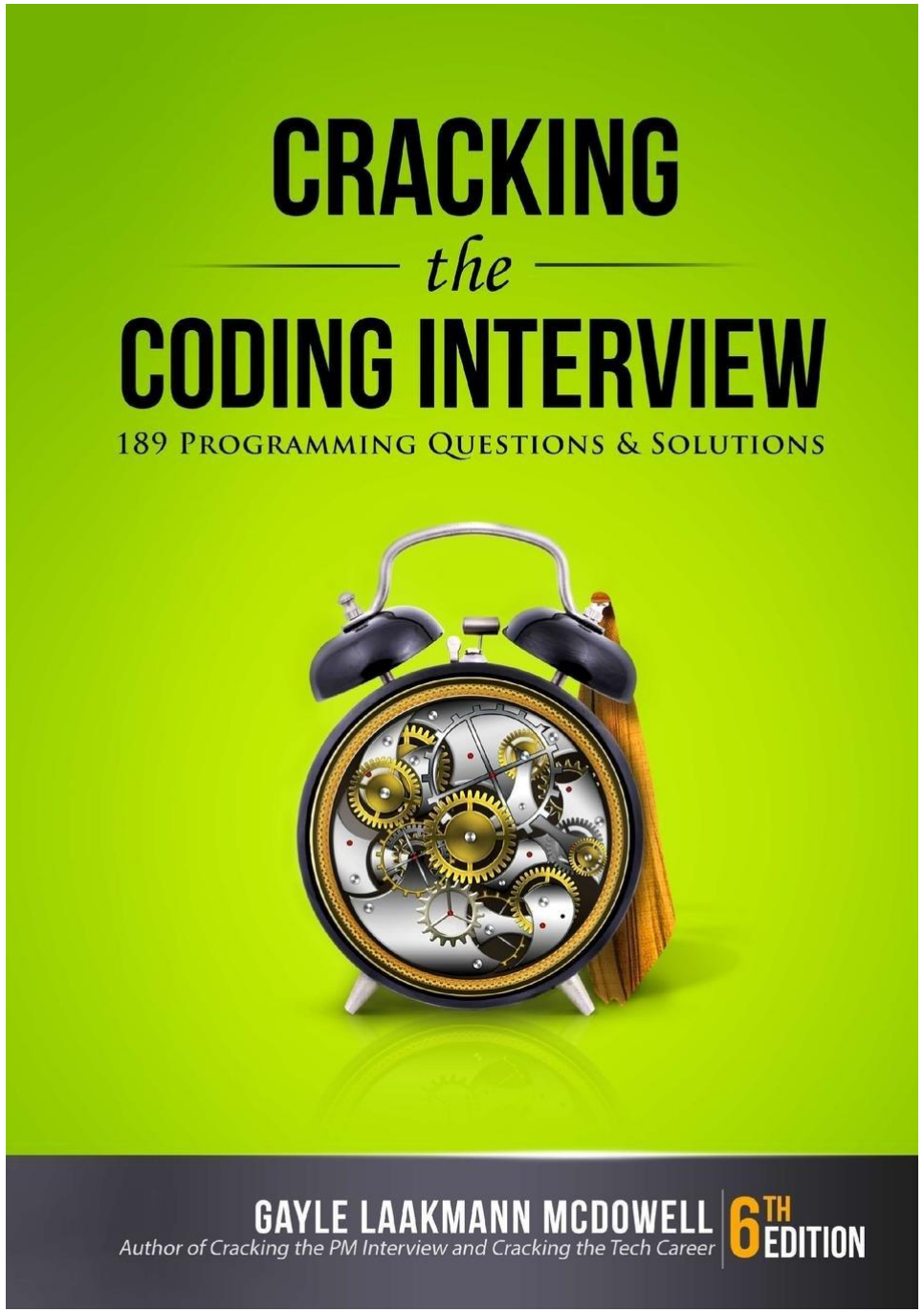
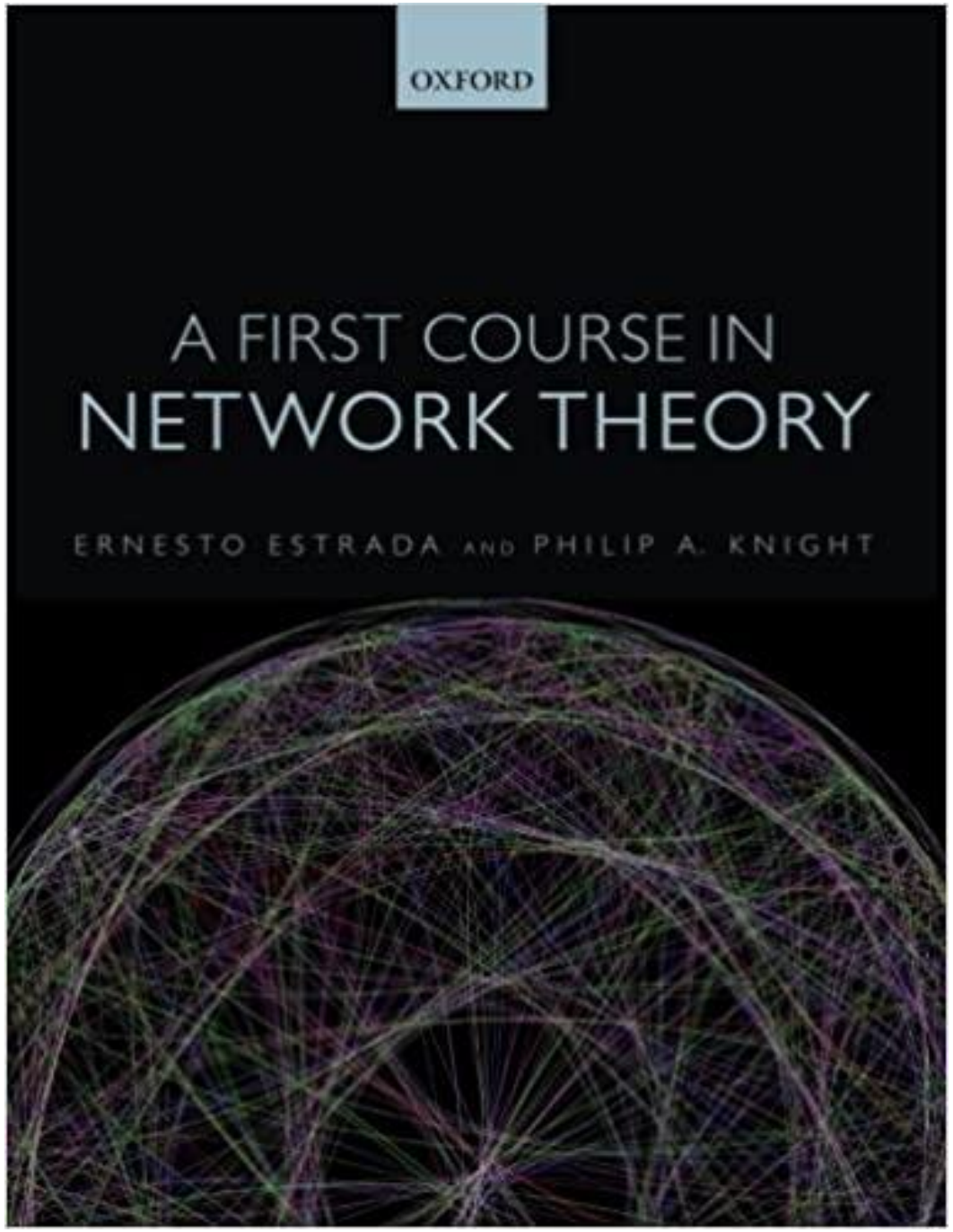
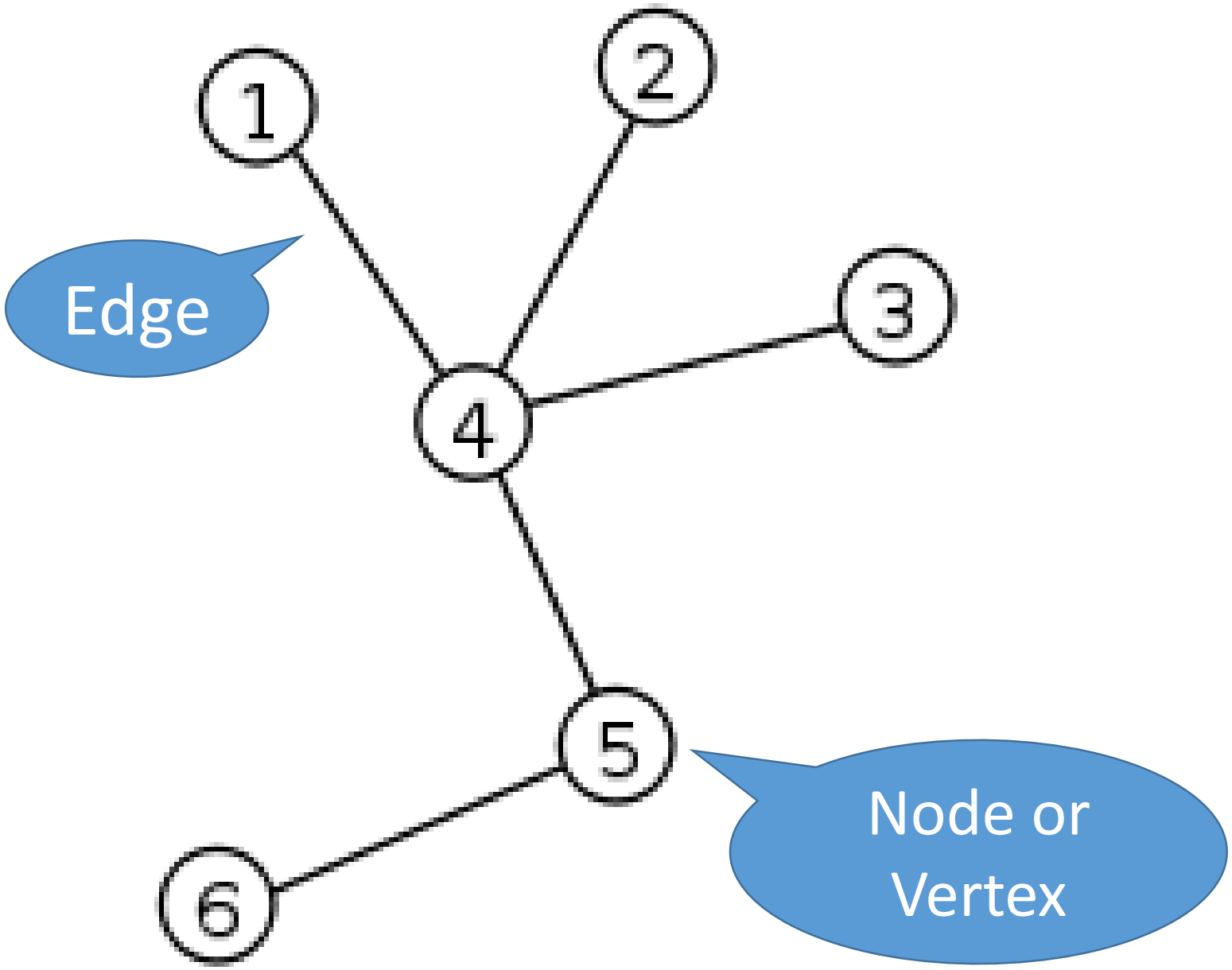

Binary Trees
Also Trees,
Binary Search Trees

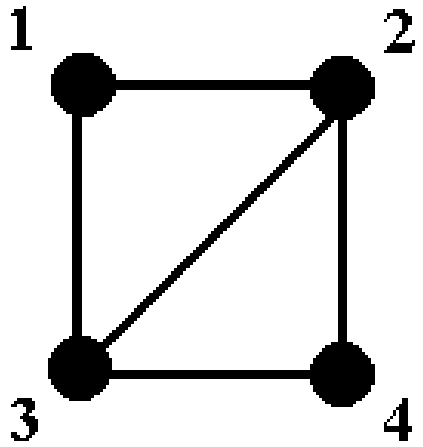

CRACKING
the
CODING INTERVIEW
189 Programming Questions & Solutions
GAYLE LAAKMANN MCDOWELL
Author of Cracking the PM Interview and Cracking the Tech Career
6TH EDITION

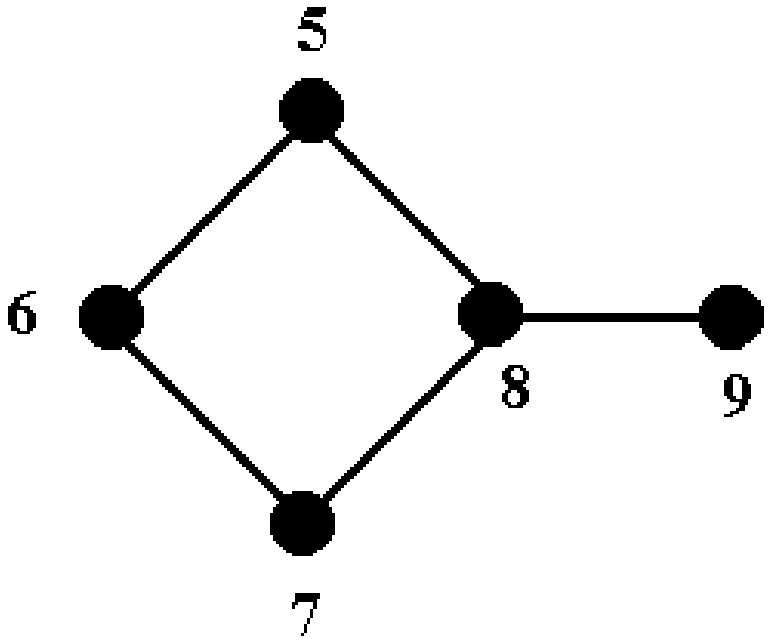

OXFORD
A FIRST COURSE IN
NETWORK THEORY
ERNESTO ESTRADA AND PHILIP A. KNIGHT

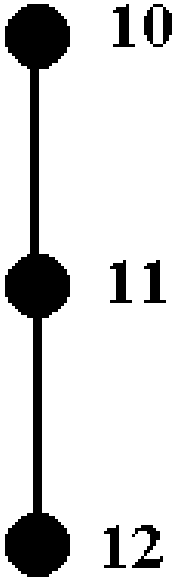


A graph is a structure constructed of nodes which are connected by edges.

A circuit is a path through a graph that starts and ends at the same vertex.

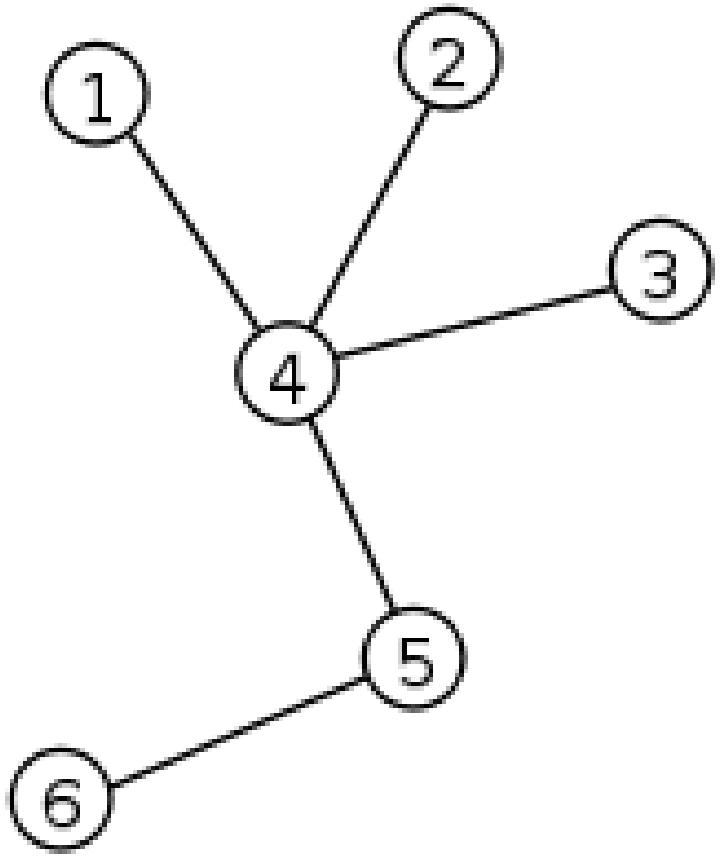


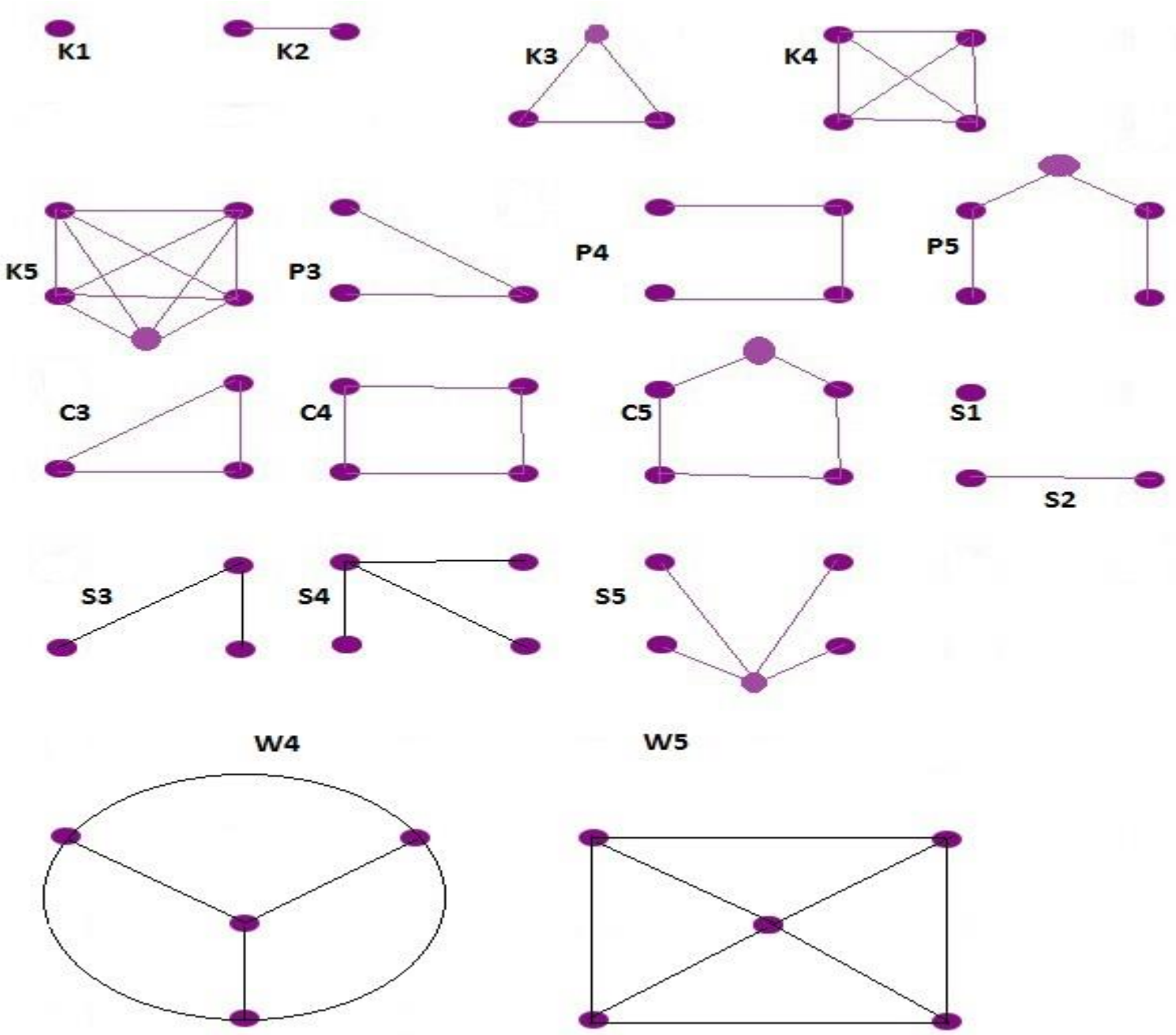




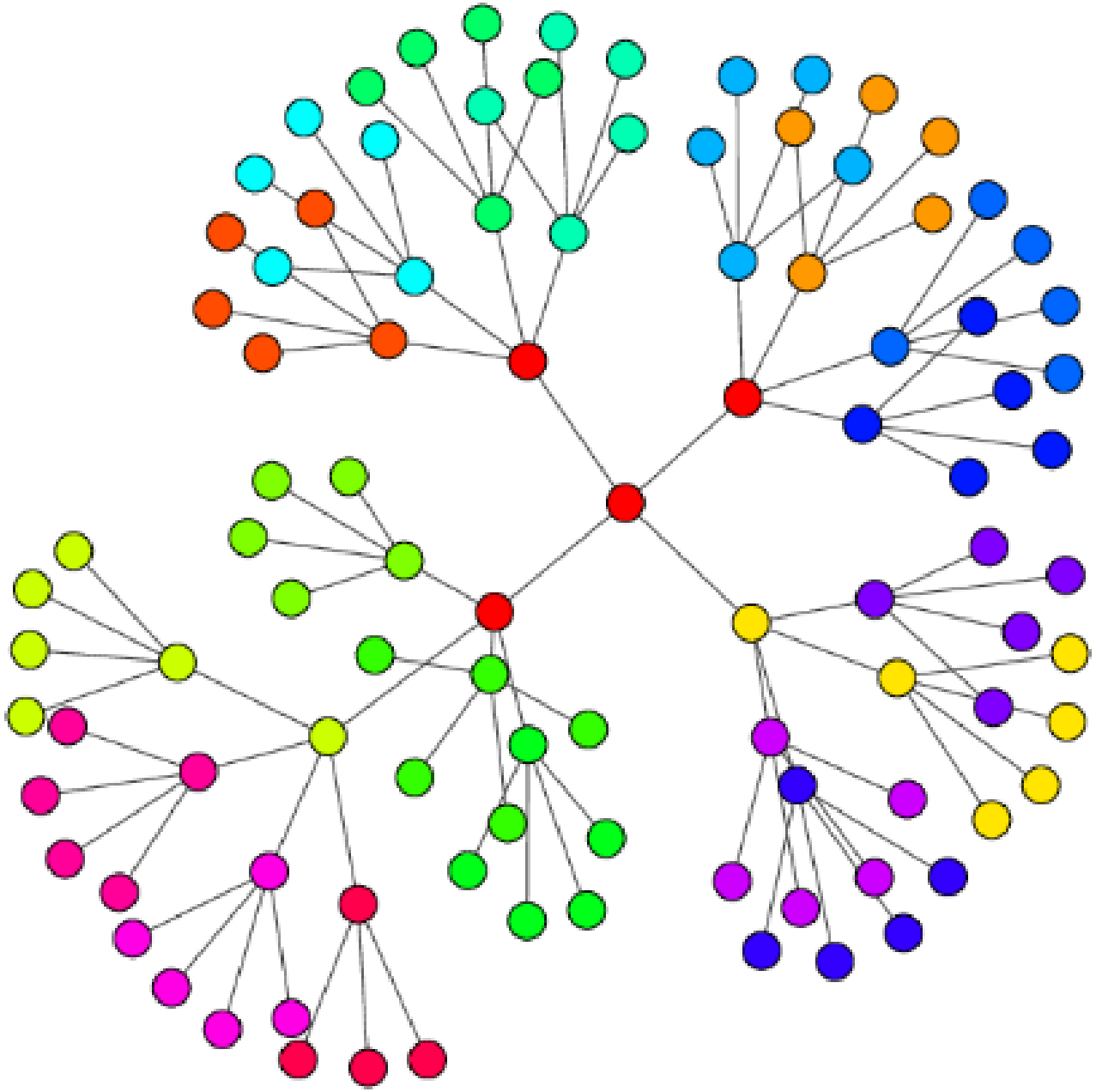
A tree is a graph in which any two vertices are connected by exactly one path.

A tree does not contain a circuit.



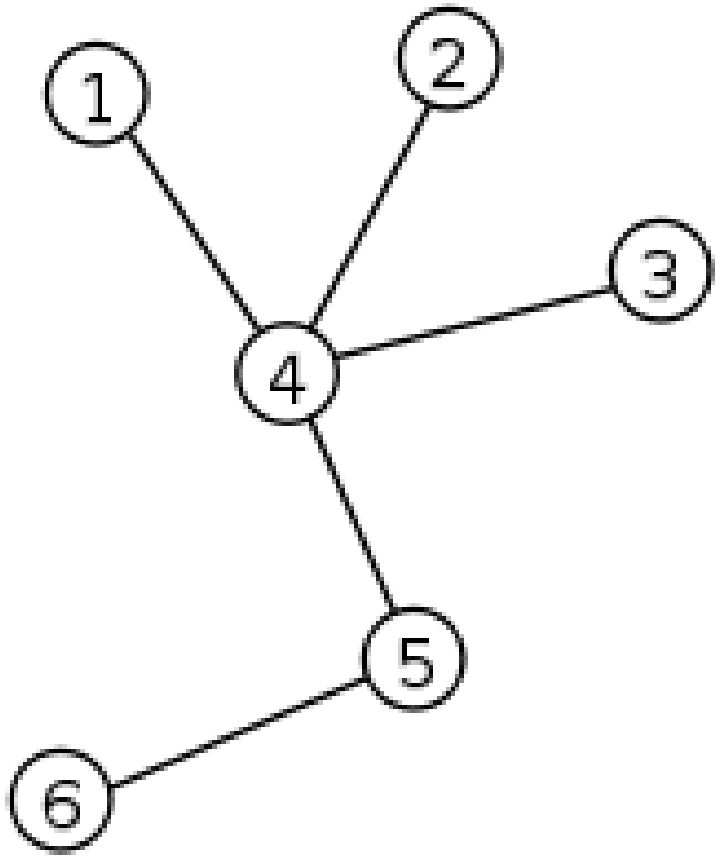


Which are trees?

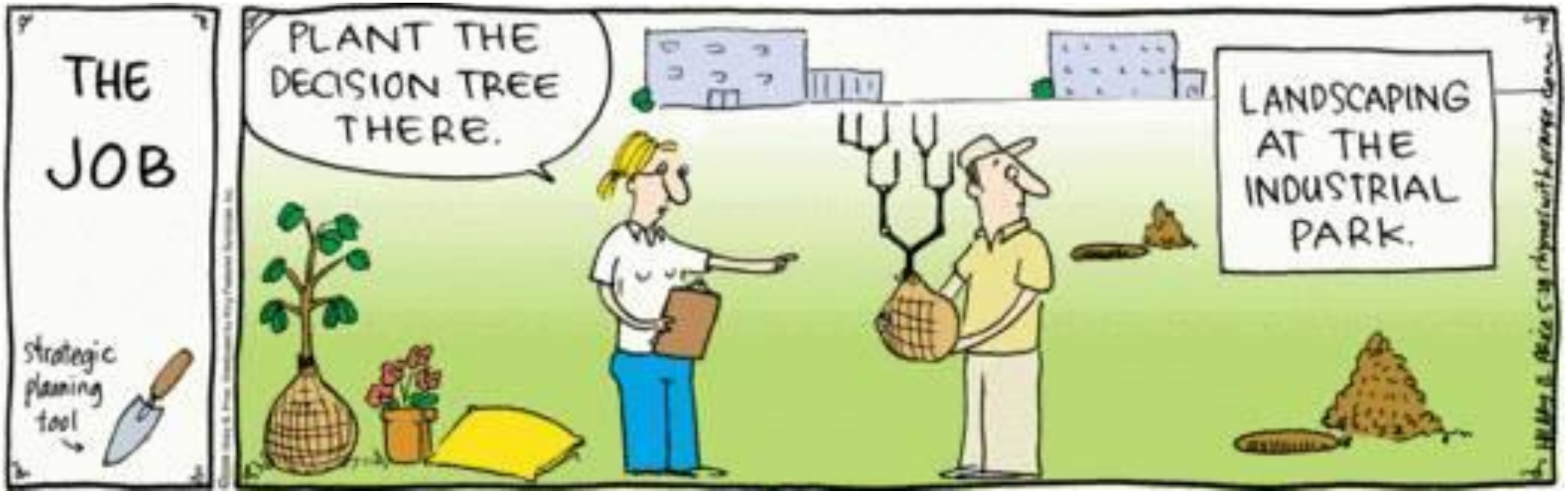


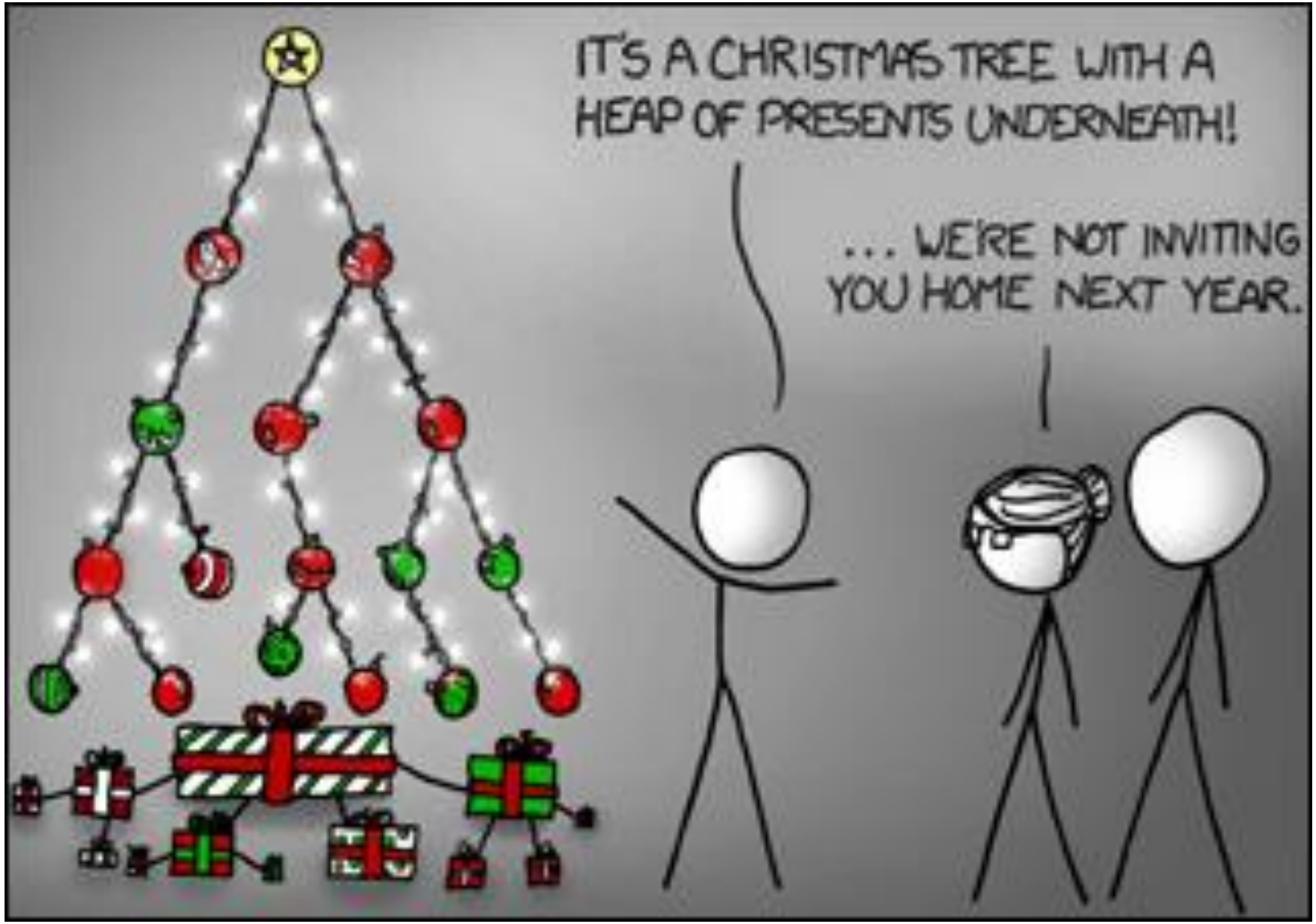
Is this a tree?

A tree of n vertices has (n-1) edges.

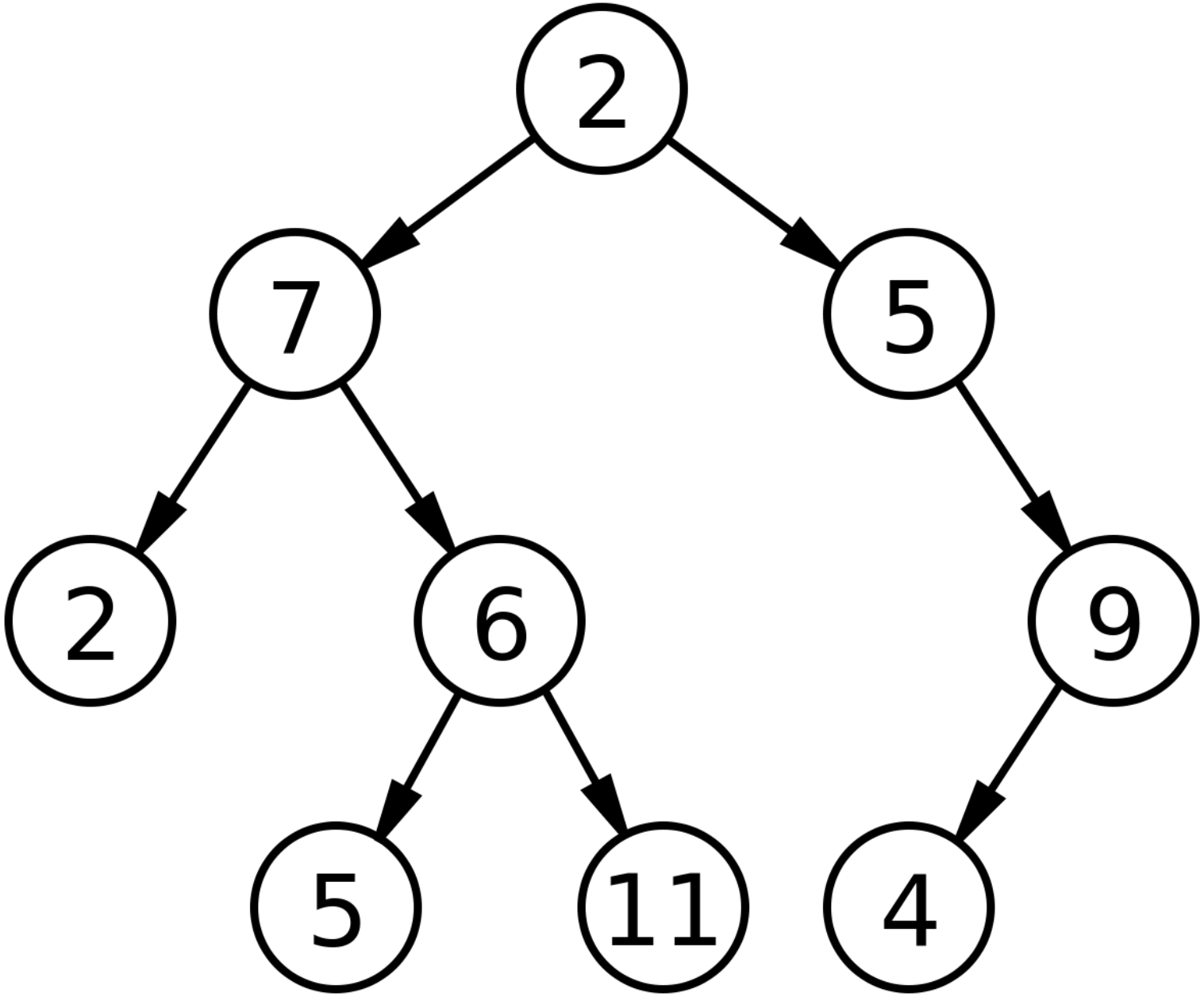


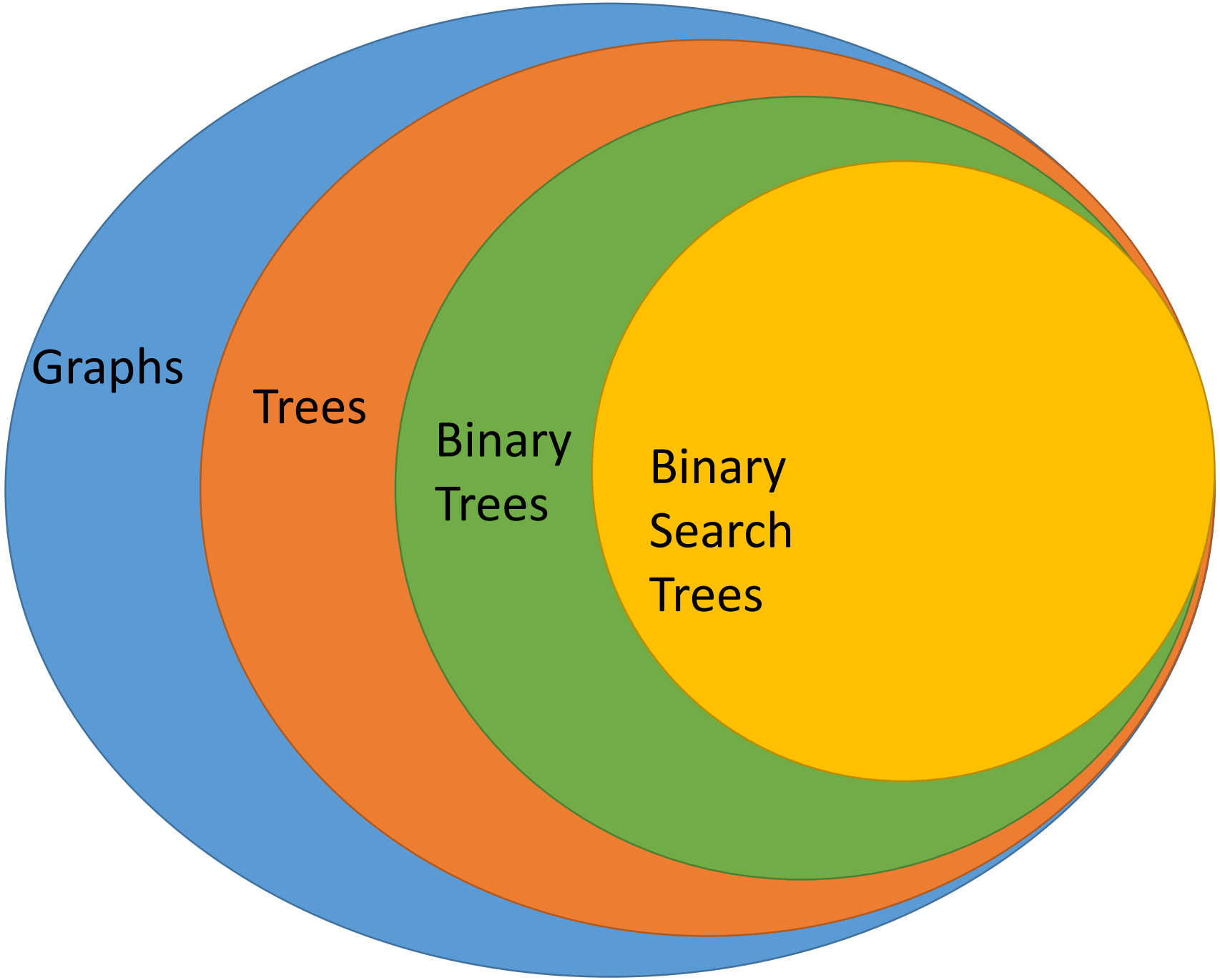
Is this a tree?

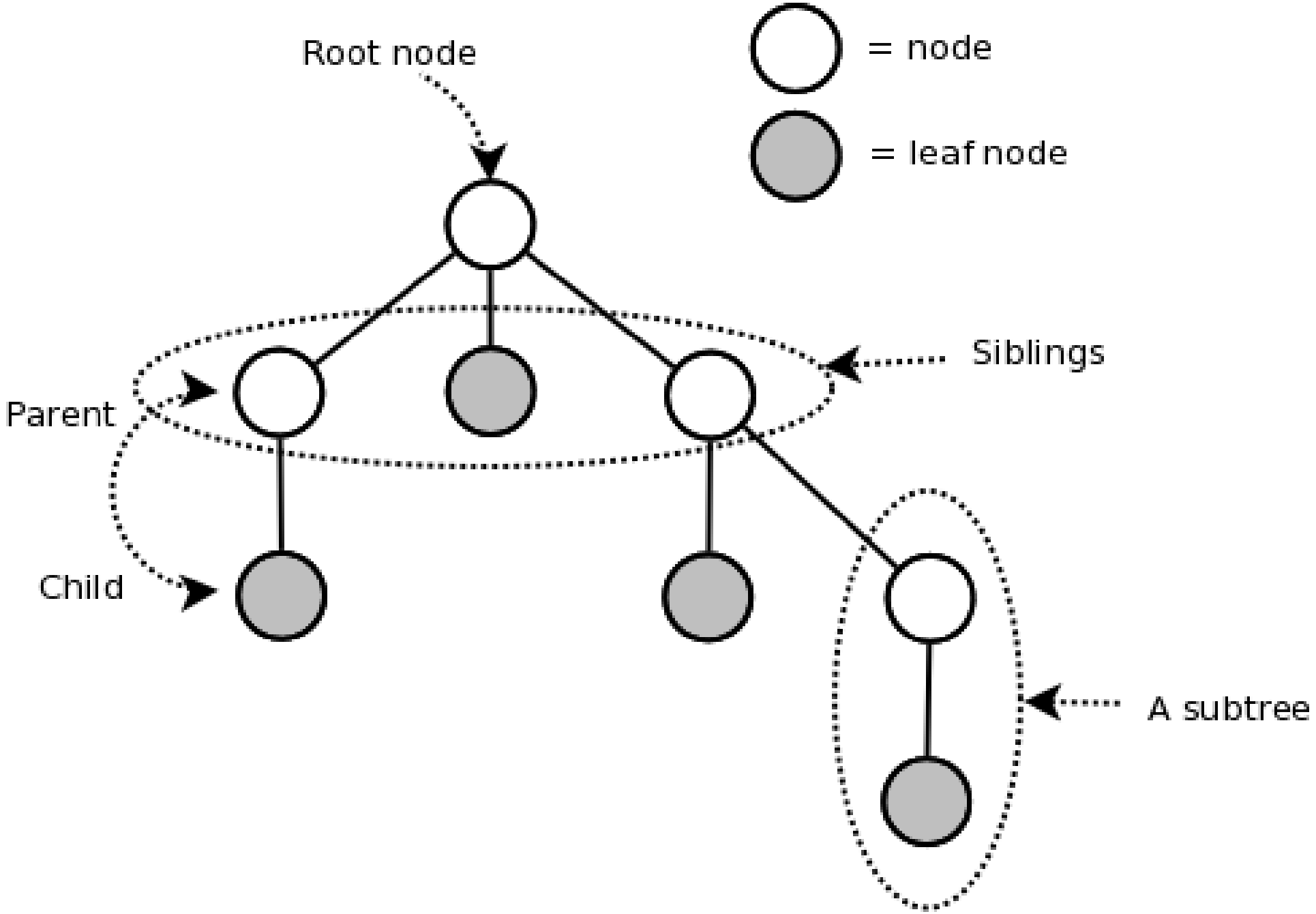

THE JOB
strategic planning tool
PLANT THE DECISION TREE THERE.
LANDSCAPING AT THE INDUSTRIAL PARK.

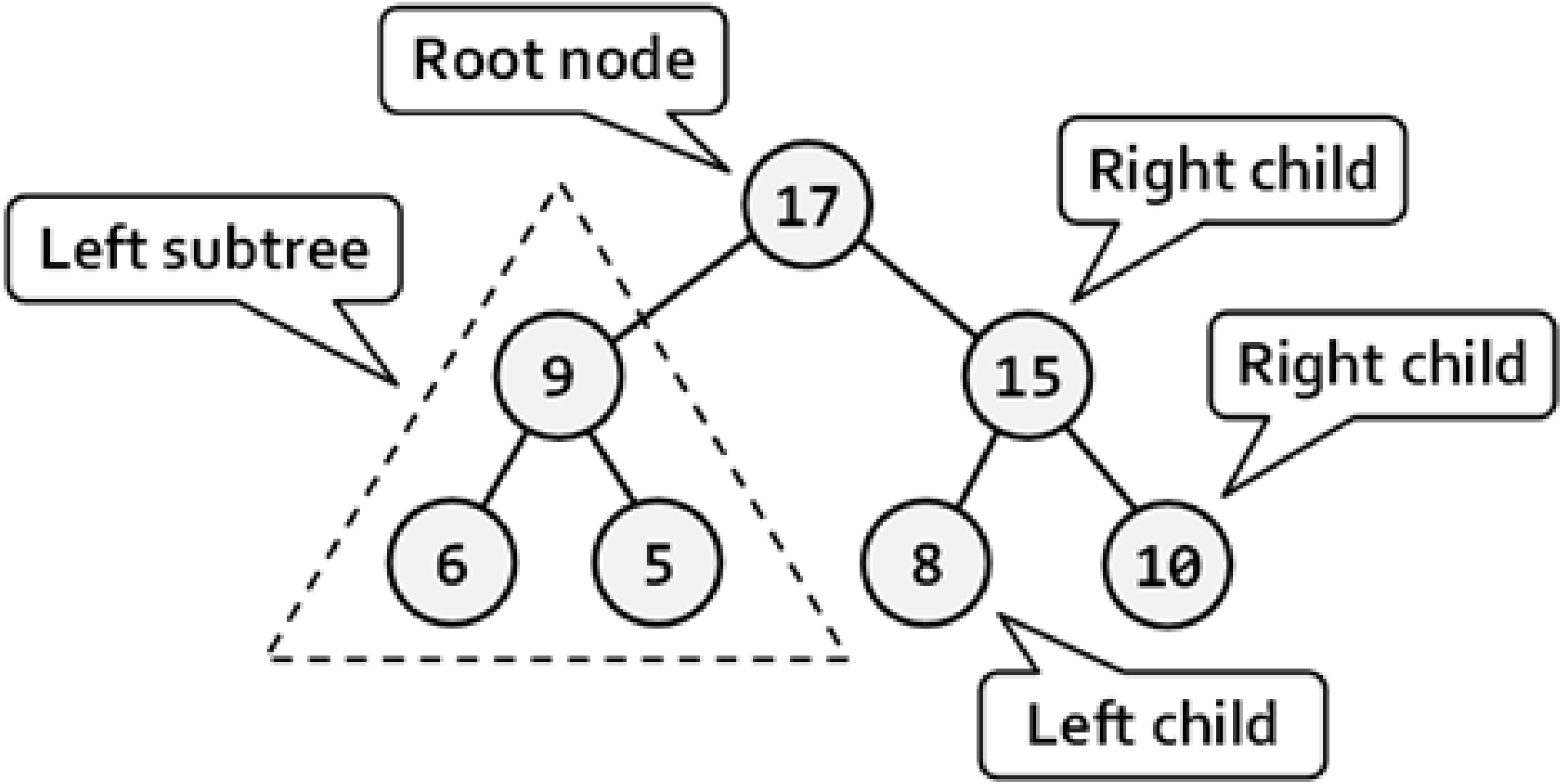

IT'S A CHRISTMAS TREE WITH A HEAP OF PRESENTS UNDERNEATH!
... WE'RE NOT INVITING YOU HOME NEXT YEAR.

In a binary tree, each vertex has at most two children.



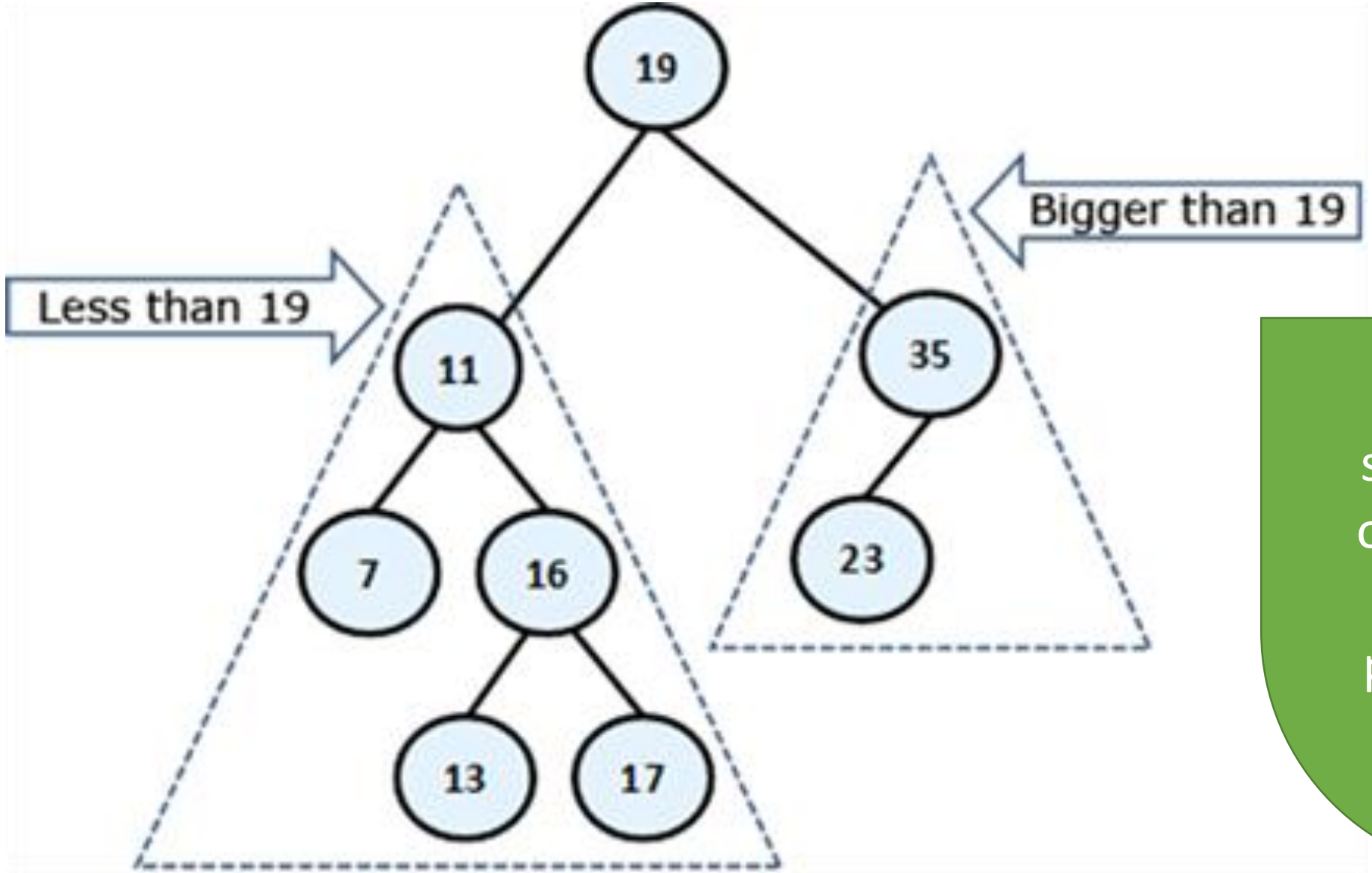

Graphs
Trees
Binary Trees
Binary Search Trees


Root node
= node
= leaf node
Siblings
Parent
Child
A subtree

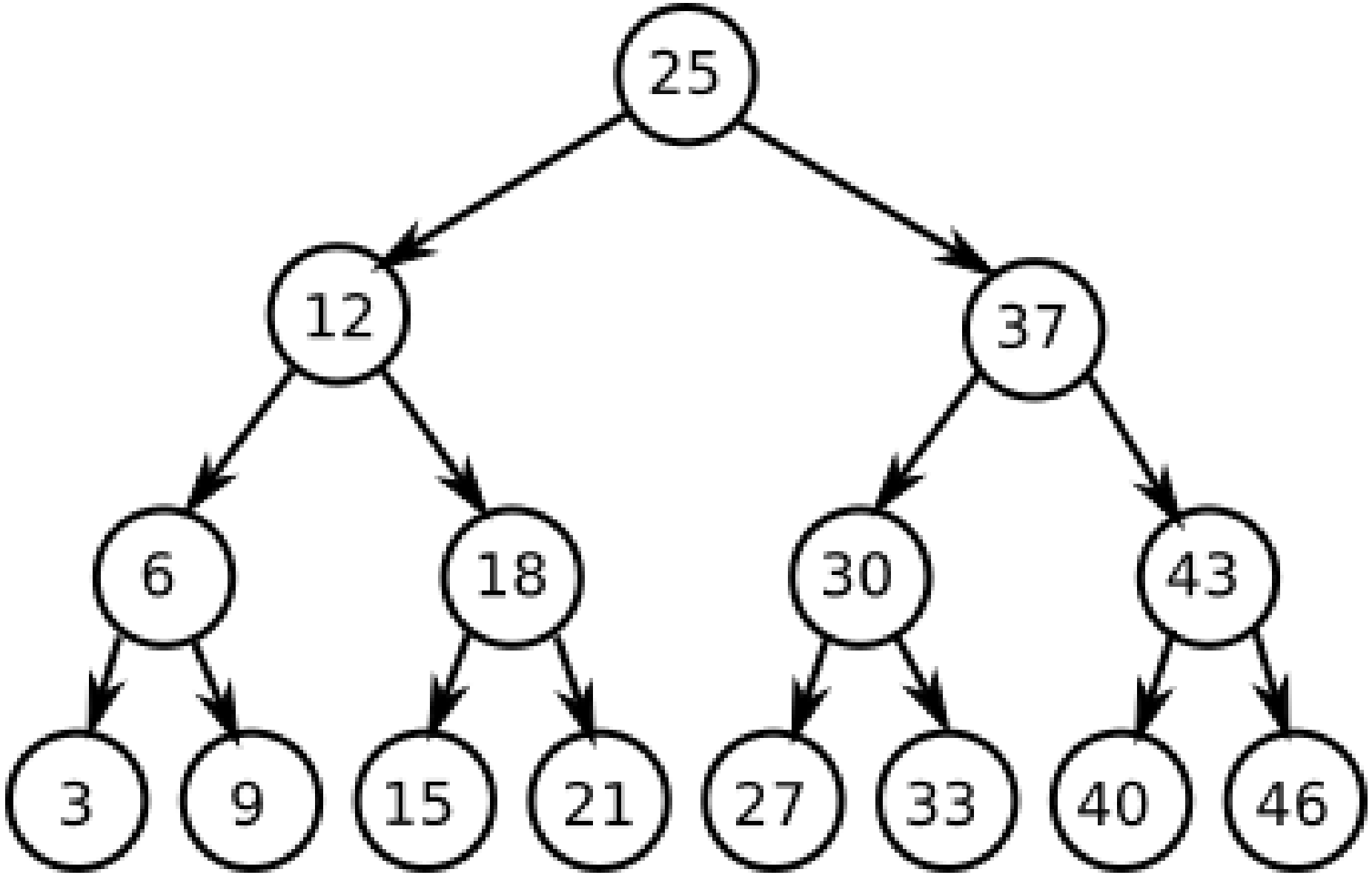

Root node
Left subtree
Right child
Right child
Left child
17
9
15
6
5
8
10

A full binary tree with i parent vertices, has 2i+1 vertics

Is this a tree?



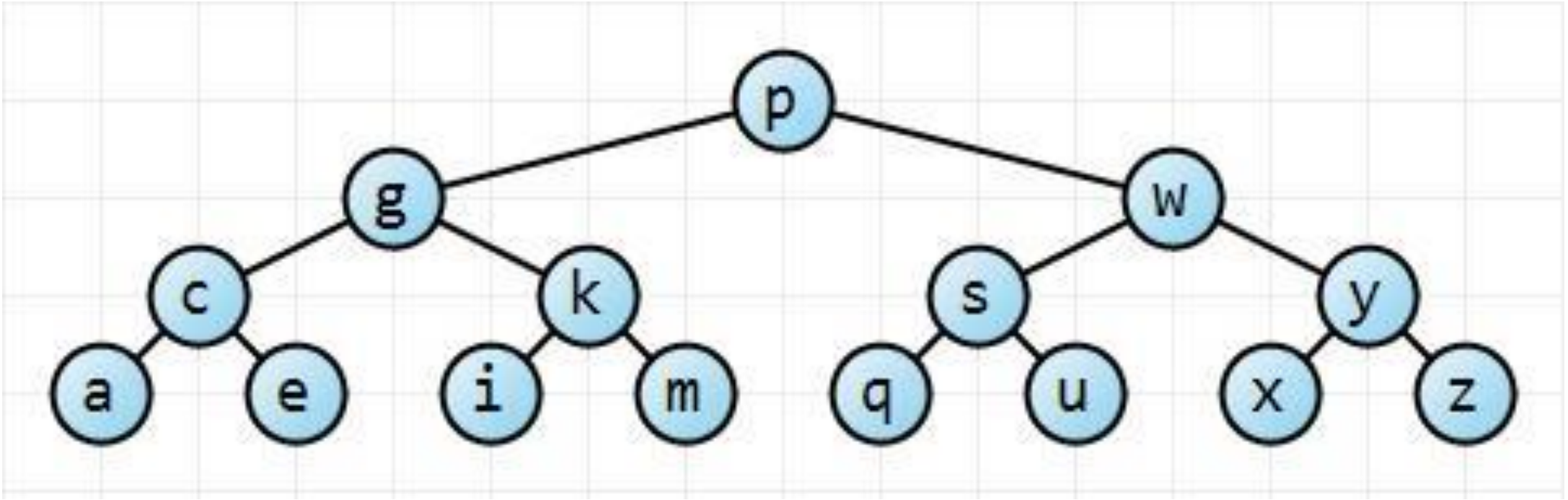
In a Binary search Tree, all of the nodes to the left of the parent are less than it, to the right, greater.



A tree?

A binary tree?

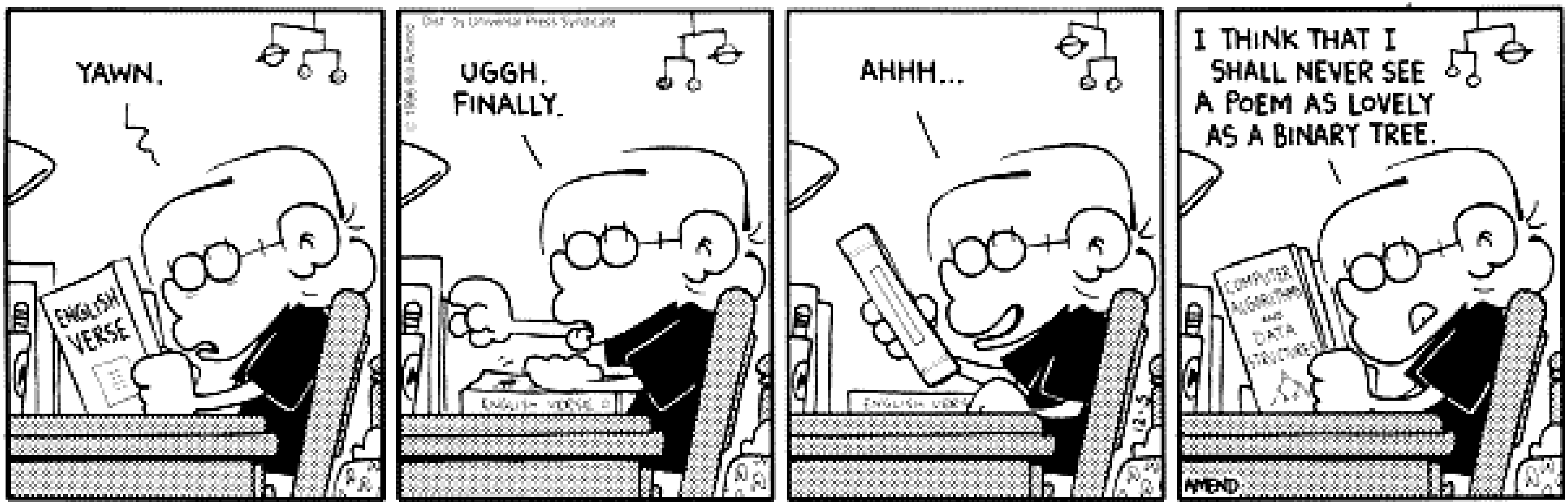
A binary search tree?



A tree?

A binary tree?

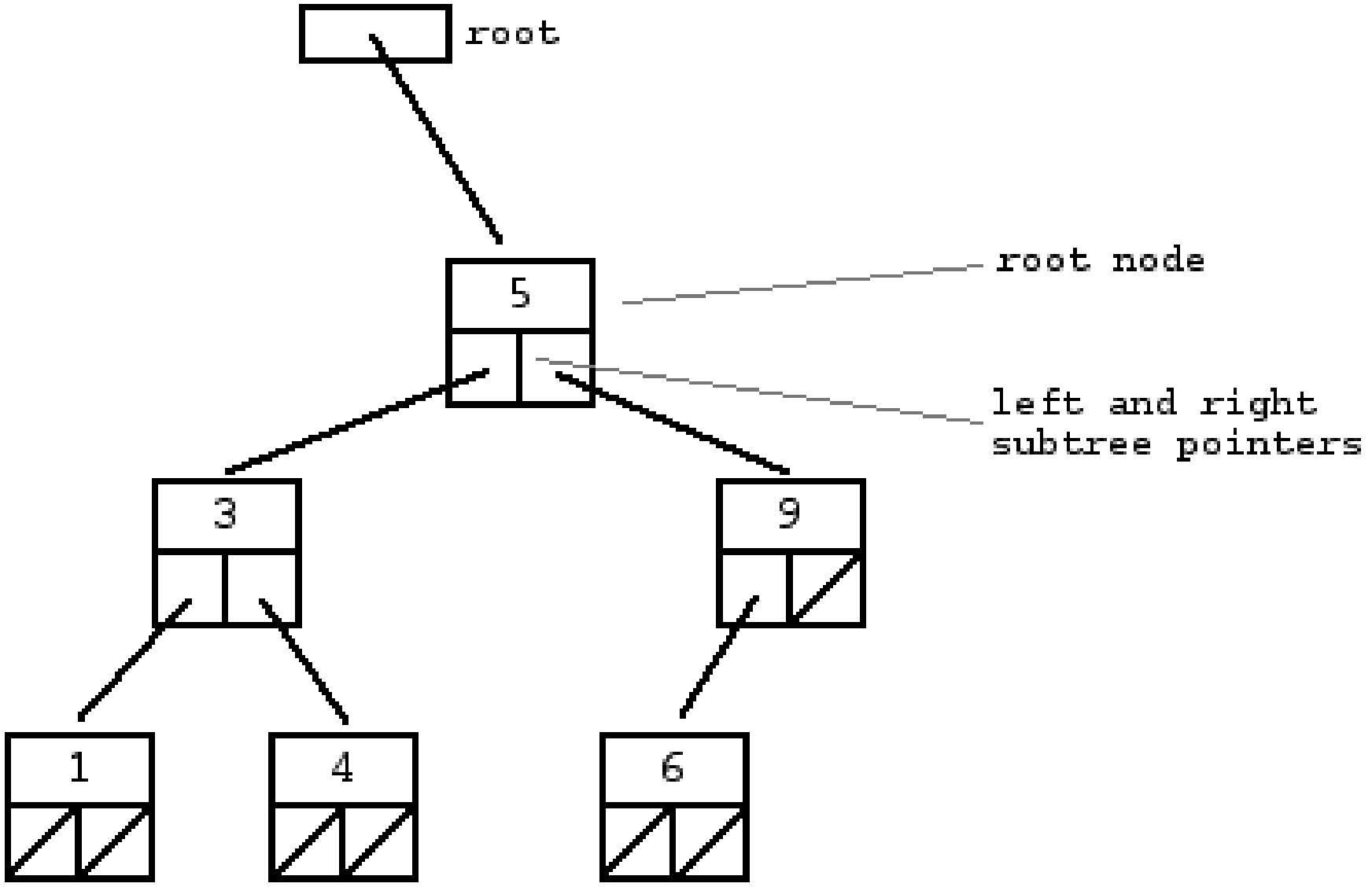
A binary search tree?

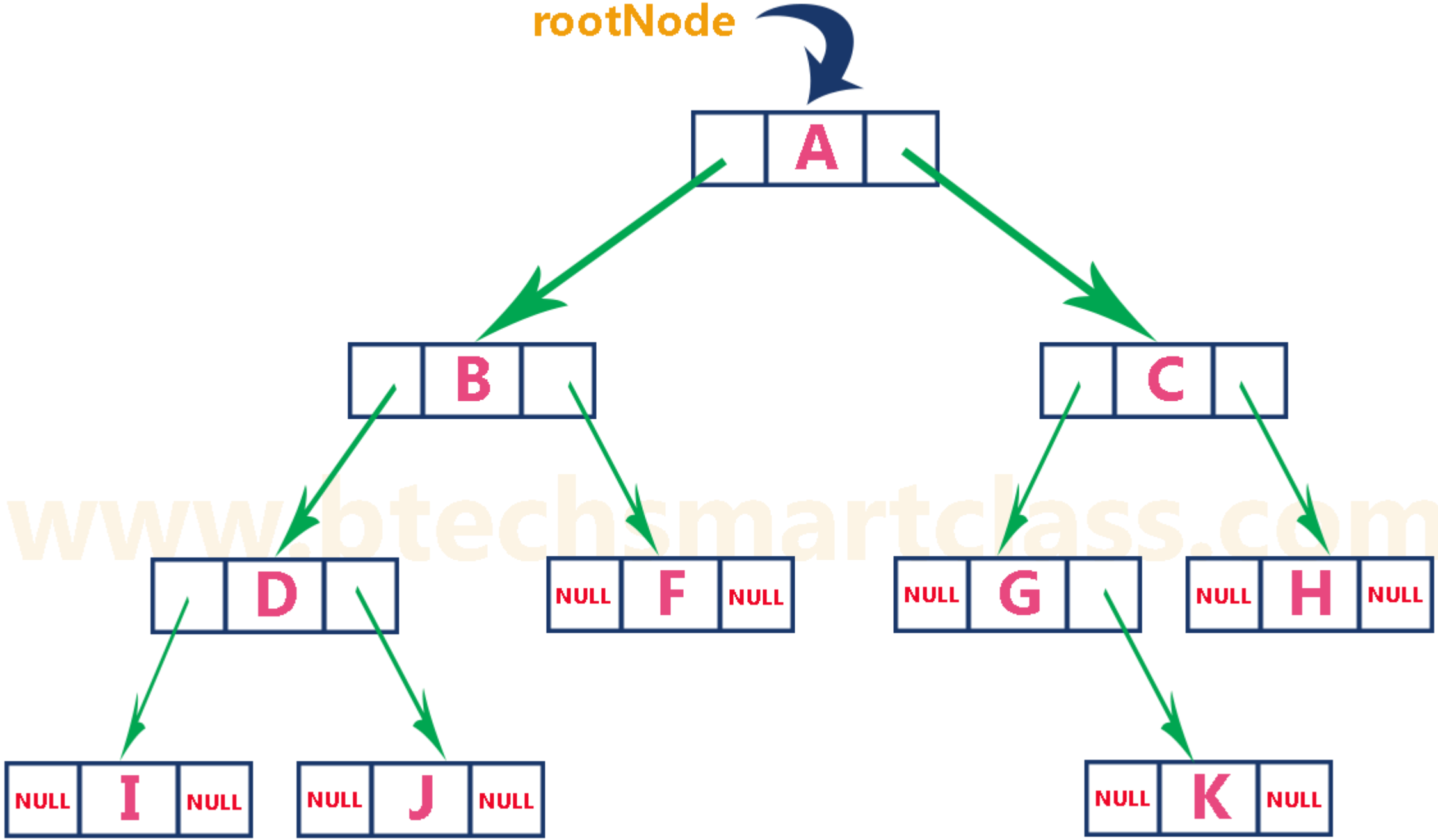

YAWN.
ENGLISH VERSE
UGGH. FINALLY.
AHHH...
I THINK THAT I SHALL NEVER SEE A POEM AS LOVELY AS A BINARY TREE.
AMEND

```
public class Node {
    String key;
    Node left, right;

   public Node (String item) {
      key = item;
      left = right = null;
   }

   public String toString () {
      return key;
   }
}
```

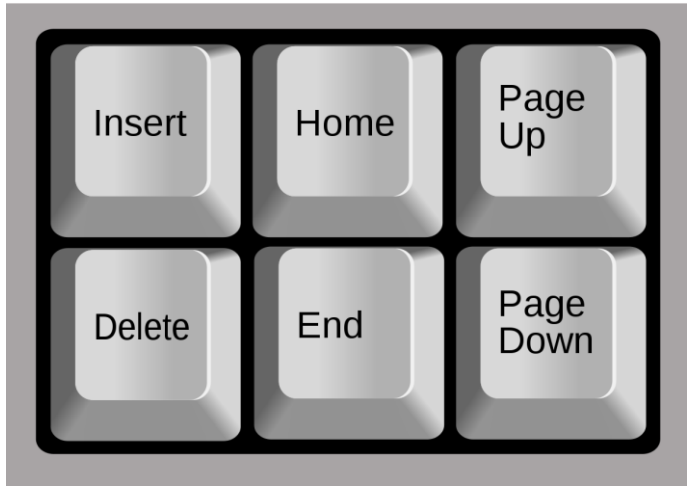

root
5
root node
left and right
subtree pointers
3
9
1
4
6



```
public class BinarySearchTree {
    private Node root;

    public BinarySearchTree () {
        root = null;
    }
```

Insert



If there is no node,

    Insert it there.

If it is less than the node,

    Go left. Try again.

If it is greater than the node,

    Go right. Try again.

```
BinarySearchTree t = new BinarySearchTree();
t.insert(25);
t.insert(15);
t.insert(60);
t.insert(30);
t.insert(61);
t.insert(42);
t.insert(10);
t.insert(17);
t.insert(20);
```

Draw this binary search tree.

Insert

If there is no node,
    Insert it there.
If it is less than the node,
    Go left. Try again.
If it is greater than the node,
    Go right. Try again.

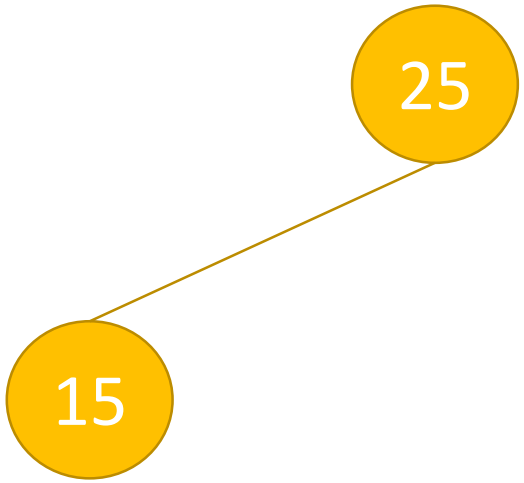
```
BinarySearchTree t = new BinarySearchTree();
t.insert(25);
t.insert(15);
t.insert(60);
t.insert(30);
t.insert(61);
t.insert(42);
t.insert(10);
t.insert(17);
t.insert(20);
```

Draw this binary search tree.

Insert

If there is no node,
    Insert it there.
If it is less than the node,
    Go left. Try again.
If it is greater than the node,
    Go right. Try again.

```
BinarySearchTree t = new BinarySearchTree();
t.insert(25);
t.insert(15);
t.insert(60);
t.insert(30);
t.insert(61);
t.insert(42);
t.insert(10);
t.insert(17);
t.insert(20);
```

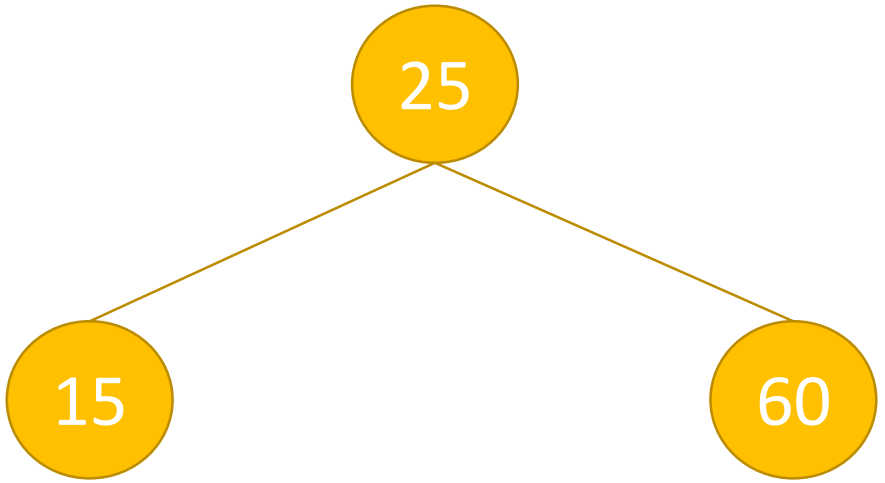


Draw this binary search tree.

Insert

If there is no node,
    Insert it there.
If it is less than the node,
    Go left. Try again.
If it is greater than the node,
    Go right. Try again.

```
BinarySearchTree t = new BinarySearchTree();
t.insert(25);
t.insert(15);
t.insert(60);
t.insert(30);
t.insert(61);
t.insert(42);
t.insert(10);
t.insert(17);
t.insert(20);
```

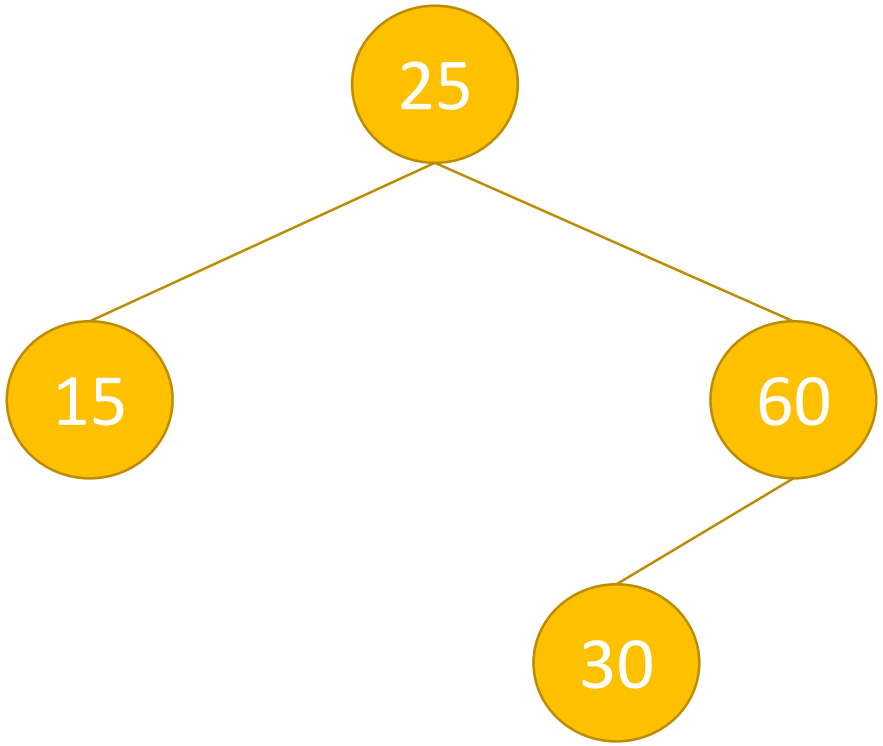


Draw this binary search tree.

Insert

If there is no node,
    Insert it there.
If it is less than the node,
    Go left. Try again.
If it is greater than the node,
    Go right. Try again.

```
BinarySearchTree t = new BinarySearchTree();
t.insert(25);
t.insert(15);
t.insert(60);
t.insert(30);
t.insert(61);
t.insert(42);
t.insert(10);
t.insert(17);
t.insert(20);
```

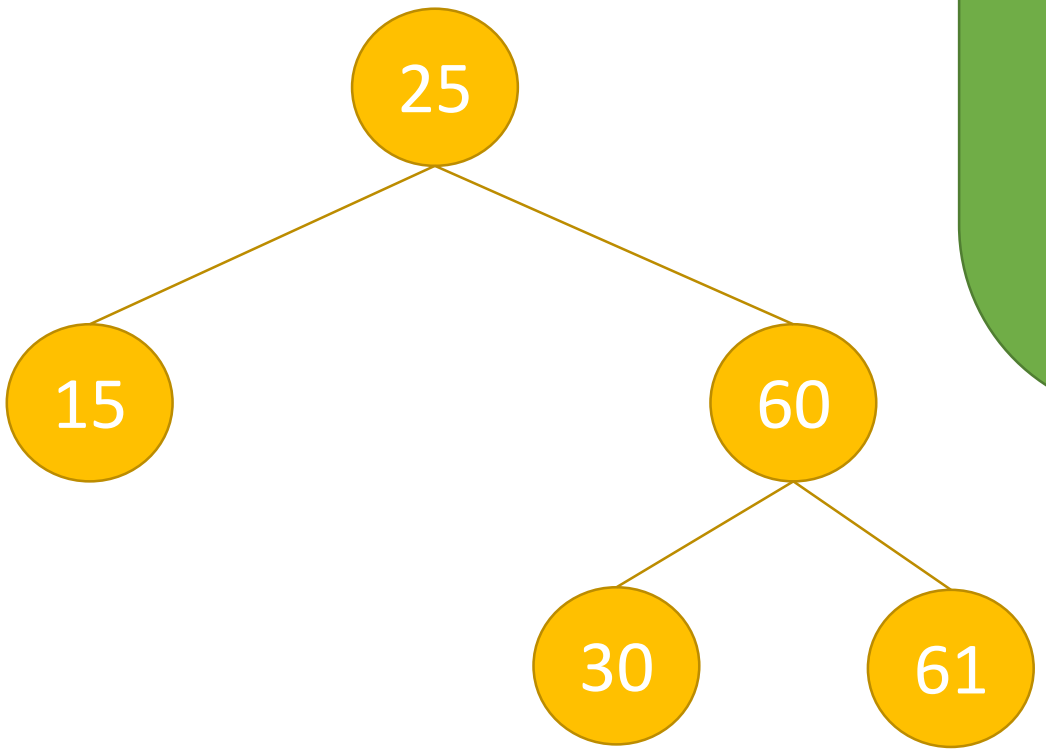


Draw this binary search tree.

Insert

If there is no node,
    Insert it there.
If it is less than the node,
    Go left. Try again.
If it is greater than the node,
    Go right. Try again.

```
BinarySearchTree t = new BinarySearchTree();
t.insert(25);
t.insert(15);
t.insert(60);
t.insert(30);
t.insert(61);
t.insert(42);
t.insert(10);
t.insert(17);
t.insert(20);
```



Draw this binary search tree.

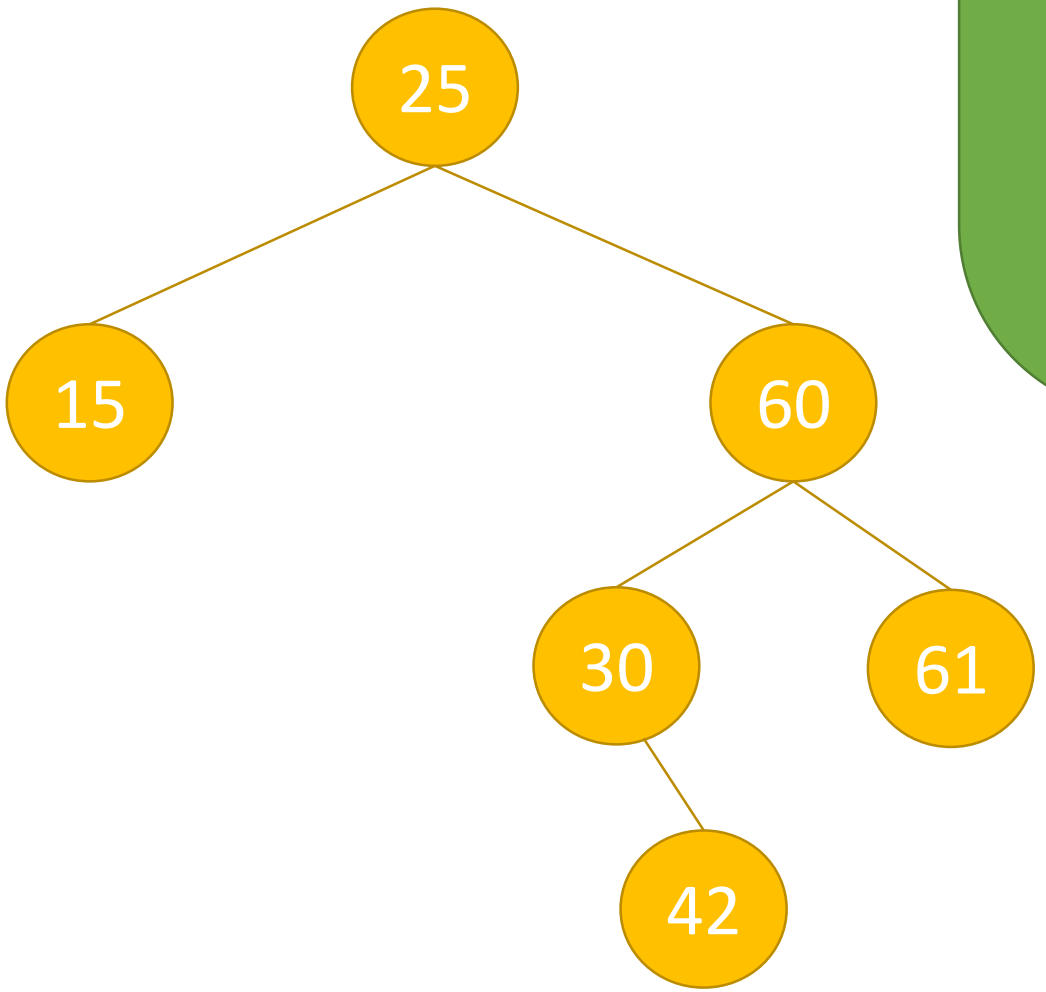
Insert

If there is no node,
    Insert it there.
If it is less than the node,
    Go left. Try again.
If it is greater than the node,
    Go right. Try again.

```
BinarySearchTree t = new BinarySearchTree();
t.insert(25);
t.insert(15);
t.insert(60);
t.insert(30);
t.insert(61);
t.insert(42);
t.insert(10);
t.insert(17);
t.insert(20);
```

Insert

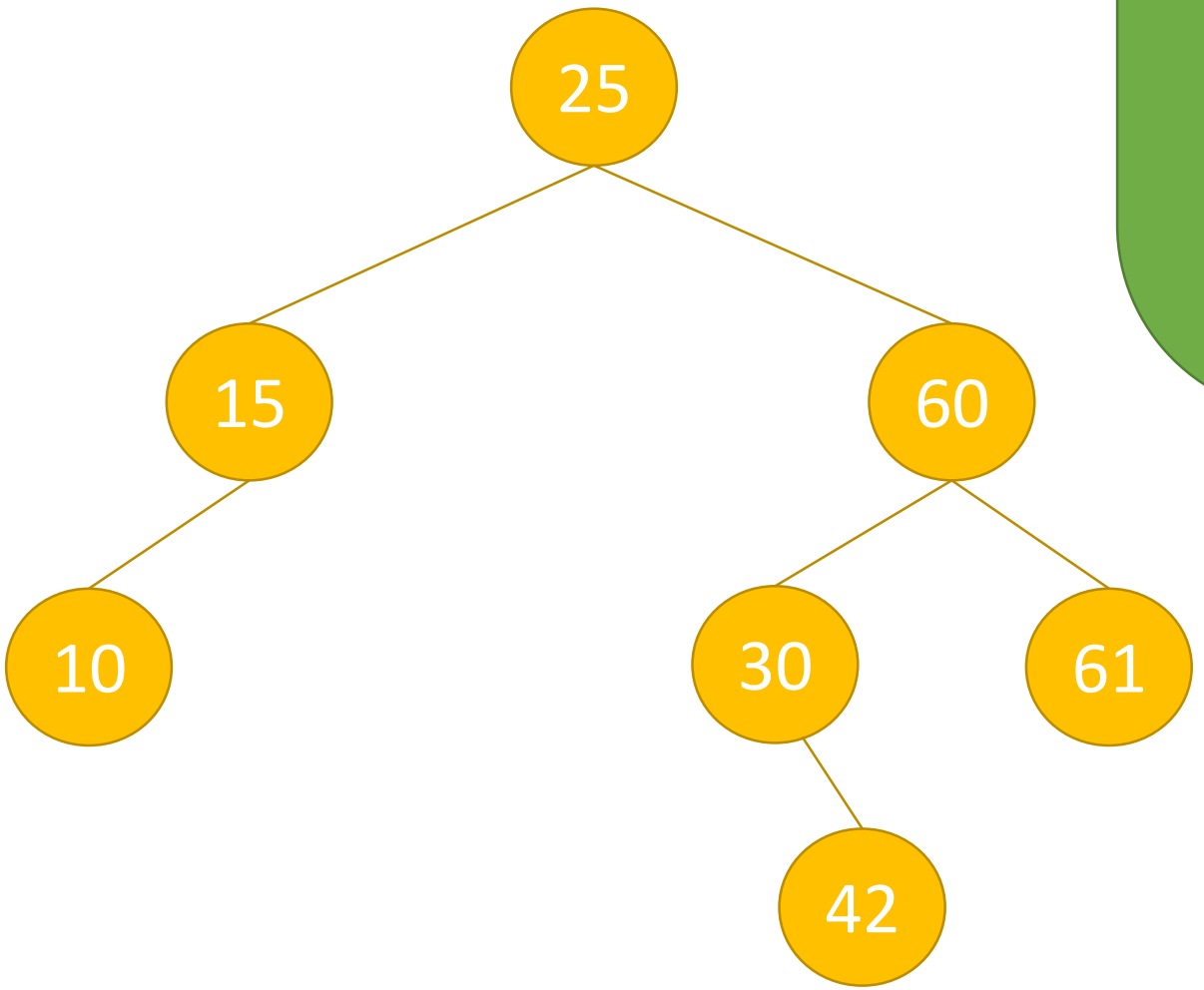
If there is no node,
    Insert it there.
If it is less than the node,
    Go left. Try again.
If it is greater than the node,
    Go right. Try again.



Draw this binary search tree.

```
BinarySearchTree t = new BinarySearchTree();
t.insert(25);
t.insert(15);
t.insert(60);
t.insert(30);
t.insert(61);
t.insert(42);
t.insert(10);
t.insert(17);
t.insert(20);
```

Draw this binary search tree.

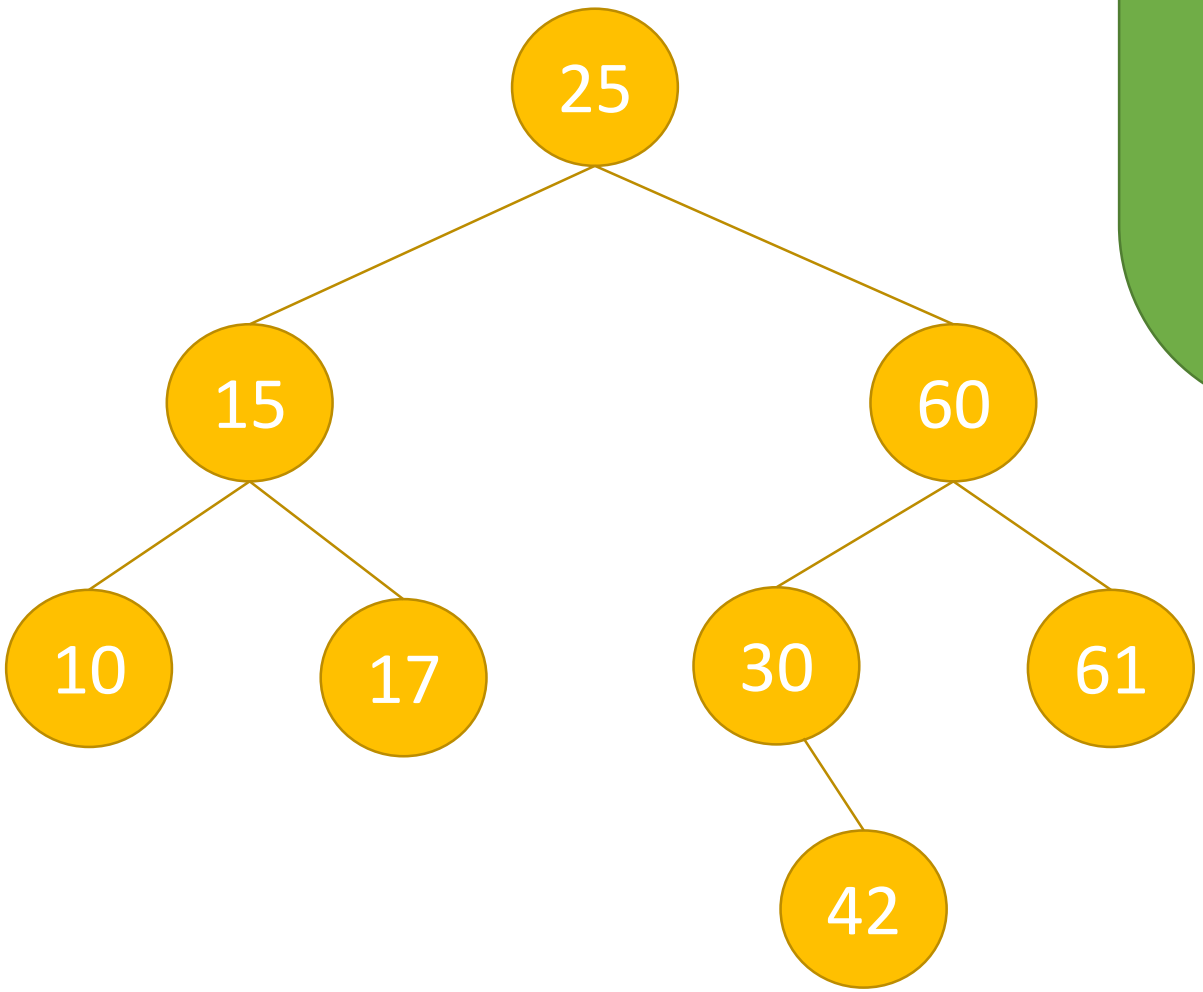


Insert

If there is no node,
    Insert it there.
If it is less than the node,
    Go left. Try again.
If it is greater than the node,
    Go right. Try again.

```
BinarySearchTree t = new BinarySearchTree();
t.insert(25);
t.insert(15);
t.insert(60);
t.insert(30);
t.insert(61);
t.insert(42);
t.insert(10);
t.insert(17);
t.insert(20);
```

Draw this binary search tree.

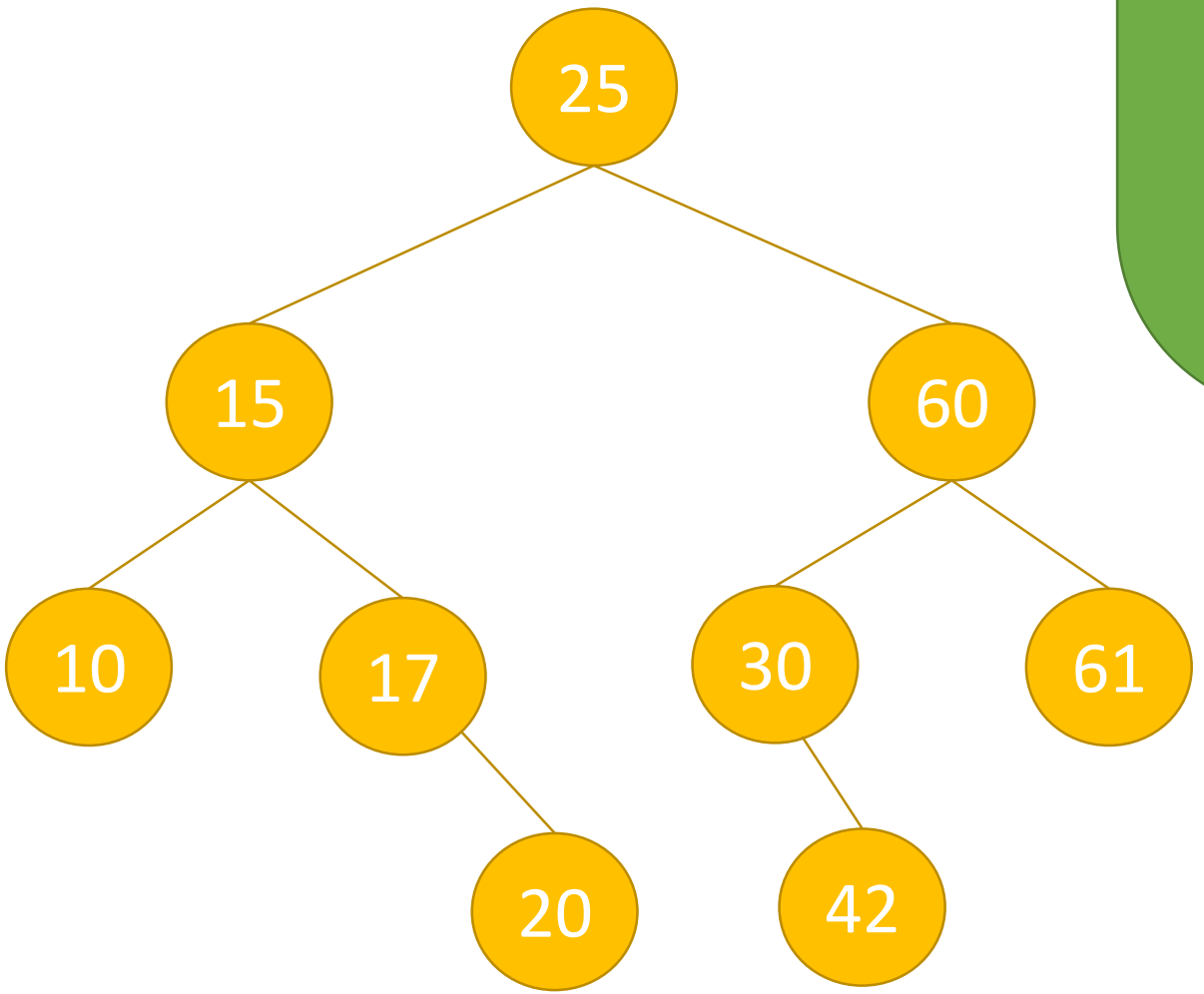


Insert

If there is no node,
    Insert it there.
If it is less than the node,
    Go left. Try again.
If it is greater than the node,
    Go right. Try again.

```
BinarySearchTree t = new BinarySearchTree();
t.insert(25);
t.insert(15);
t.insert(60);
t.insert(30);
t.insert(61);
t.insert(42);
t.insert(10);
t.insert(17);
t.insert(20);
```

Draw this binary search tree.



Insert

If there is no node,
    Insert it there.
If it is less than the node,
    Go left. Try again.
If it is greater than the node,
    Go right. Try again.

```
BinarySearchTree t = new BinarySearchTree();
t.insert("m");
t.insert("k");
t.insert("c");
t.insert("e");
t.insert("f");
t.insert("a");
t.insert("w");
t.insert("z");
t.insert("p");
```

Draw this binary search tree.

Insert

If there is no node,
    Insert it there.
If it is less than the node,
    Go left. Try again.
If it is greater than the node,
    Go right. Try again.

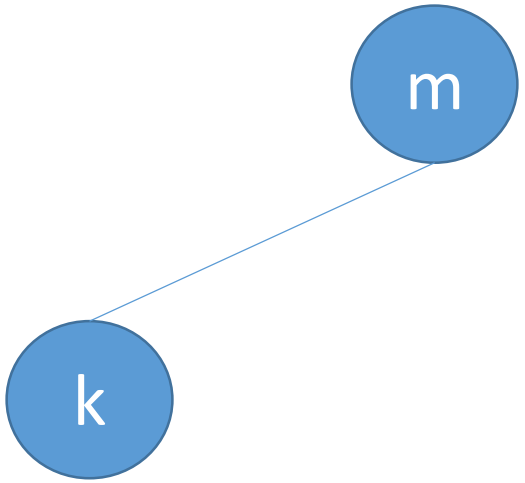
```
BinarySearchTree t = new BinarySearchTree();
t.insert("m");
t.insert("k");
t.insert("c");
t.insert("e");
t.insert("f");
t.insert("a");
t.insert("w");
t.insert("z");
t.insert("p");
```

Draw this binary search tree.

Insert

If there is no node,
    Insert it there.
If it is less than the node,
    Go left. Try again.
If it is greater than the node,
    Go right. Try again.

```
BinarySearchTree t = new BinarySearchTree();
t.insert("m");
t.insert("k");
t.insert("c");
t.insert("e");
t.insert("f");
t.insert("a");
t.insert("w");
t.insert("z");
t.insert("p");
```

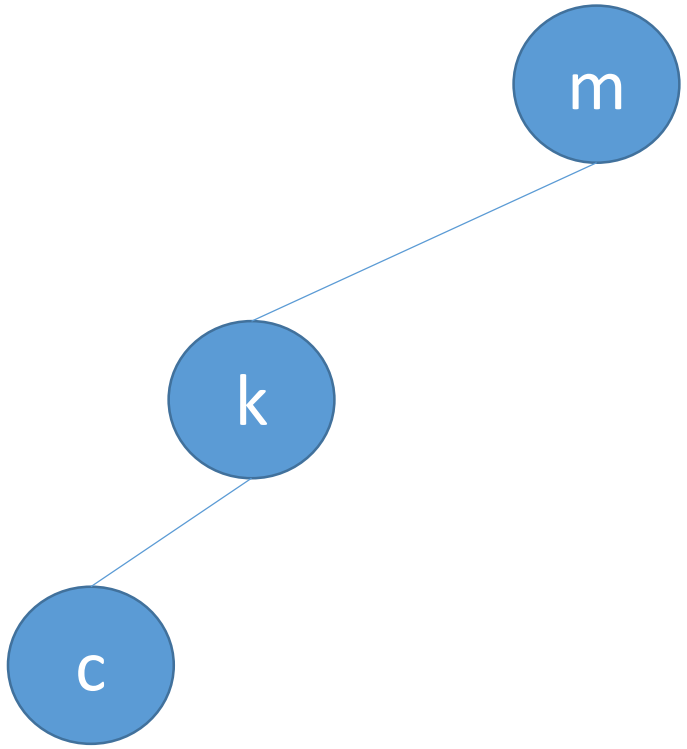


Draw this binary search tree.

Insert

If there is no node,
    Insert it there.
If it is less than the node,
    Go left. Try again.
If it is greater than the node,
    Go right. Try again.

```
BinarySearchTree t = new BinarySearchTree();
t.insert("m");
t.insert("k");
t.insert("c");
t.insert("e");
t.insert("f");
t.insert("a");
t.insert("w");
t.insert("z");
t.insert("p");
```

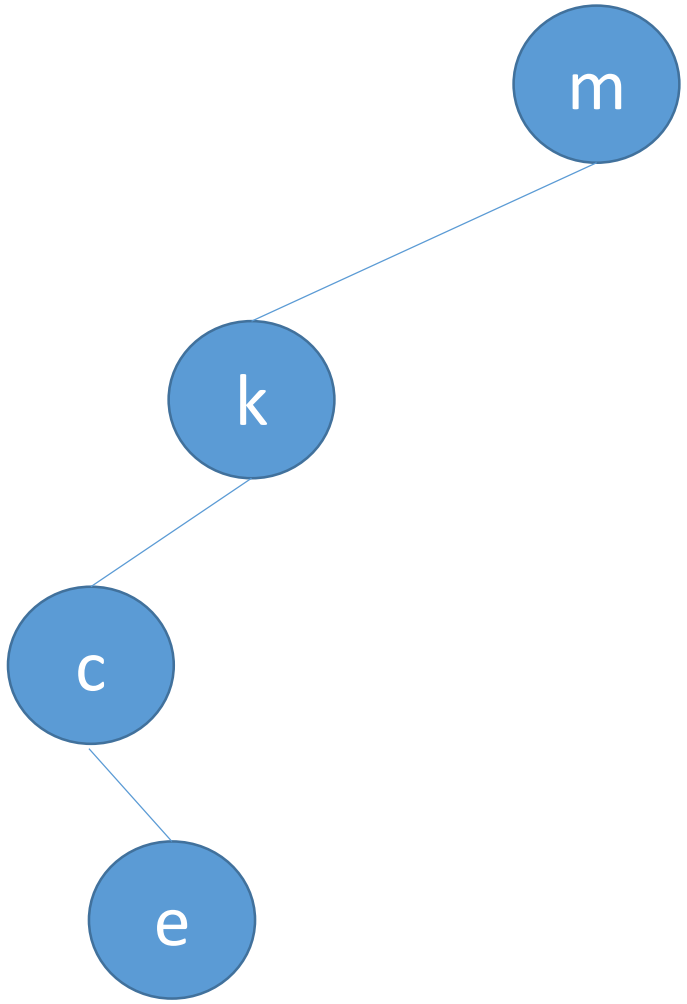


Draw this binary search tree.

Insert

If there is no node,
    Insert it there.
If it is less than the node,
    Go left. Try again.
If it is greater than the node,
    Go right. Try again.

```
BinarySearchTree t = new BinarySearchTree();
t.insert("m");
t.insert("k");
t.insert("c");
t.insert("e");
t.insert("f");
t.insert("a");
t.insert("w");
t.insert("z");
t.insert("p");
```

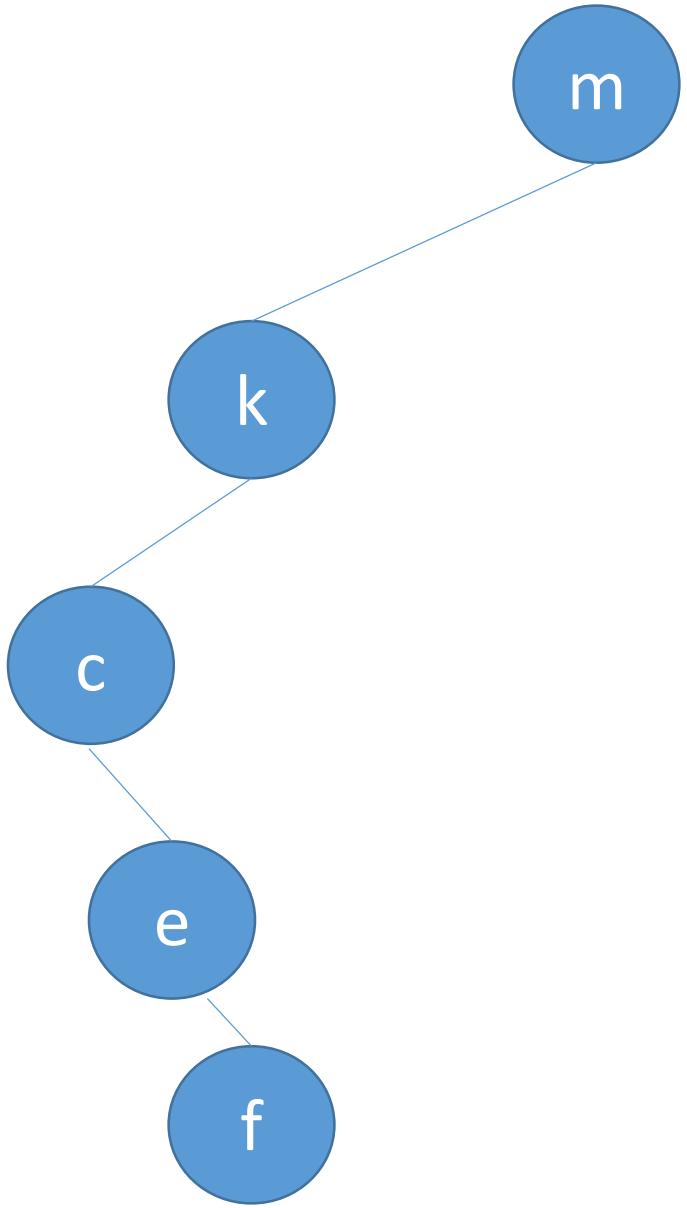


Draw this binary search tree.

Insert

If there is no node,
    Insert it there.
If it is less than the node,
    Go left. Try again.
If it is greater than the node,
    Go right. Try again.

```
BinarySearchTree t = new BinarySearchTree();
t.insert("m");
t.insert("k");
t.insert("c");
t.insert("e");
t.insert("f");
t.insert("a");
t.insert("w");
t.insert("z");
t.insert("p");
```

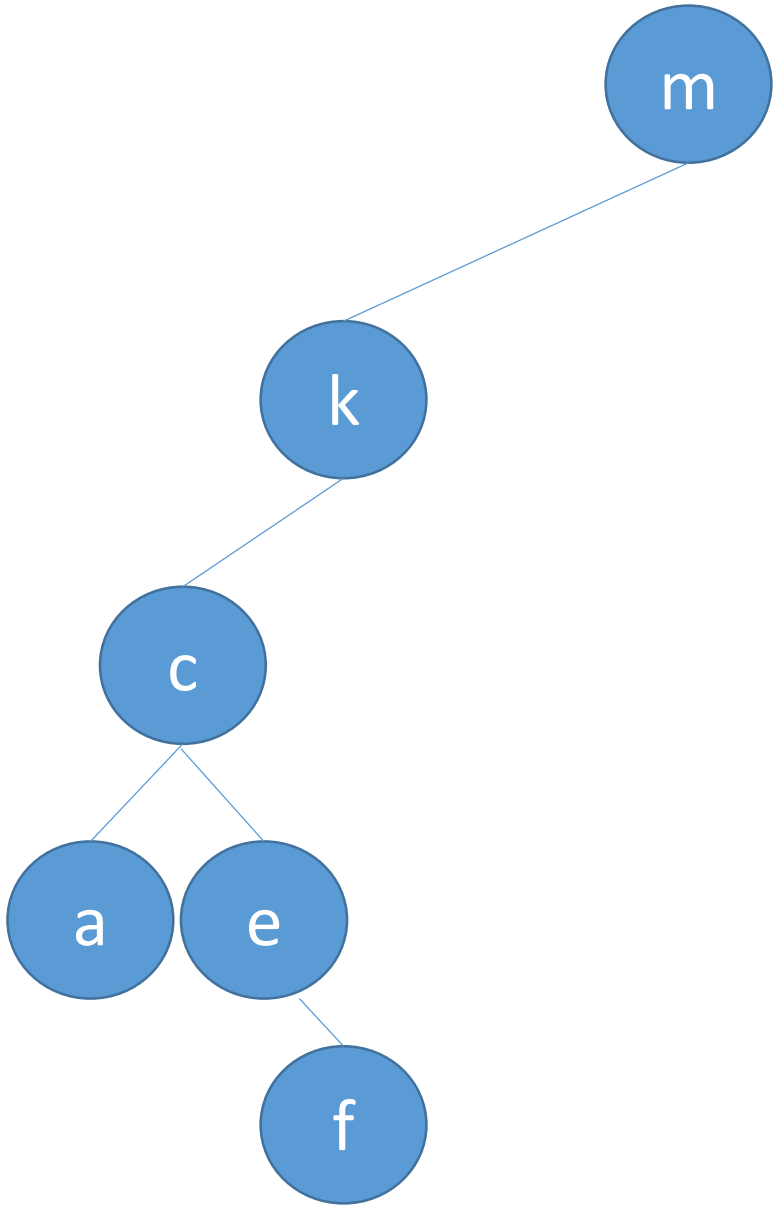


Draw this binary search tree.

Insert

If there is no node,
    Insert it there.
If it is less than the node,
    Go left. Try again.
If it is greater than the node,
    Go right. Try again.

```
BinarySearchTree t = new BinarySearchTree();
t.insert("m");
t.insert("k");
t.insert("c");
t.insert("e");
t.insert("f");
t.insert("a");
t.insert("w");
t.insert("z");
t.insert("p");
```

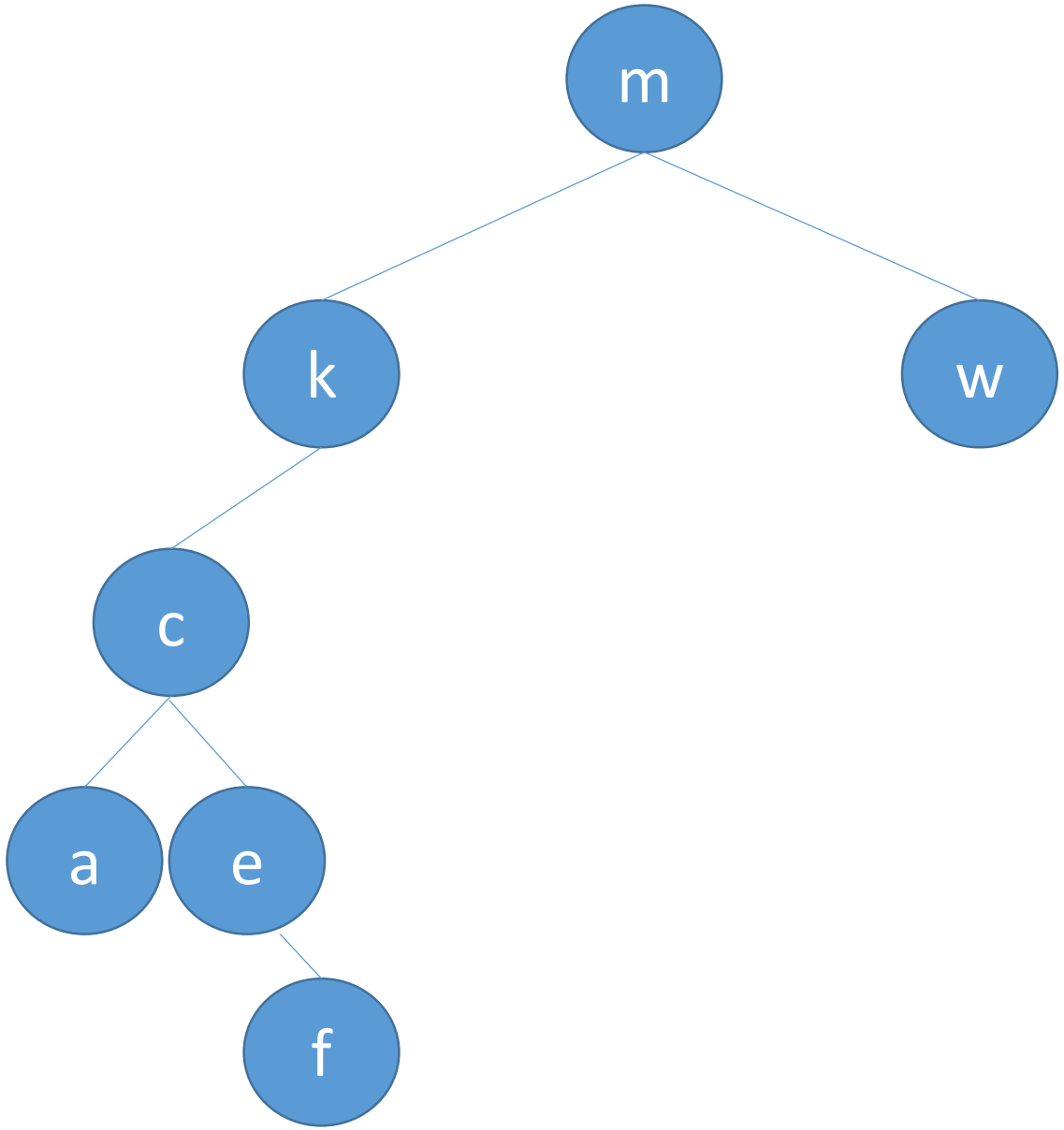


Draw this binary search tree.

Insert

If there is no node,
    Insert it there.
If it is less than the node,
    Go left. Try again.
If it is greater than the node,
    Go right. Try again.

```
BinarySearchTree t = new BinarySearchTree();
t.insert("m");
t.insert("k");
t.insert("c");
t.insert("e");
t.insert("f");
t.insert("a");
t.insert("w");
t.insert("z");
t.insert("p");
```

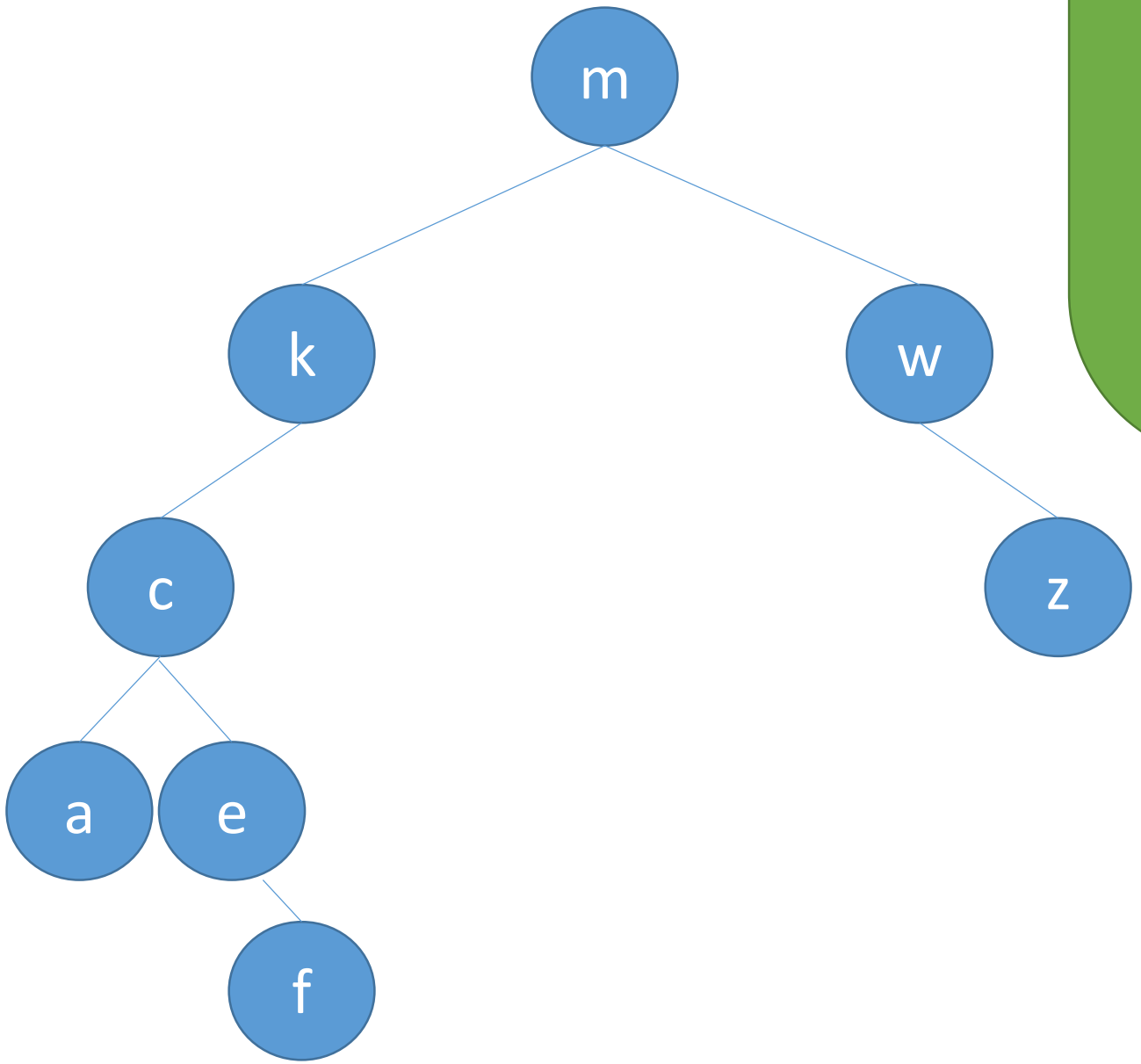


Draw this binary search tree.

Insert

If there is no node,
    Insert it there.
If it is less than the node,
    Go left. Try again.
If it is greater than the node,
    Go right. Try again.

```
BinarySearchTree t = new BinarySearchTree();
t.insert("m");
t.insert("k");
t.insert("c");
t.insert("e");
t.insert("f");
t.insert("a");
t.insert("w");
t.insert("z");
t.insert("p");
```

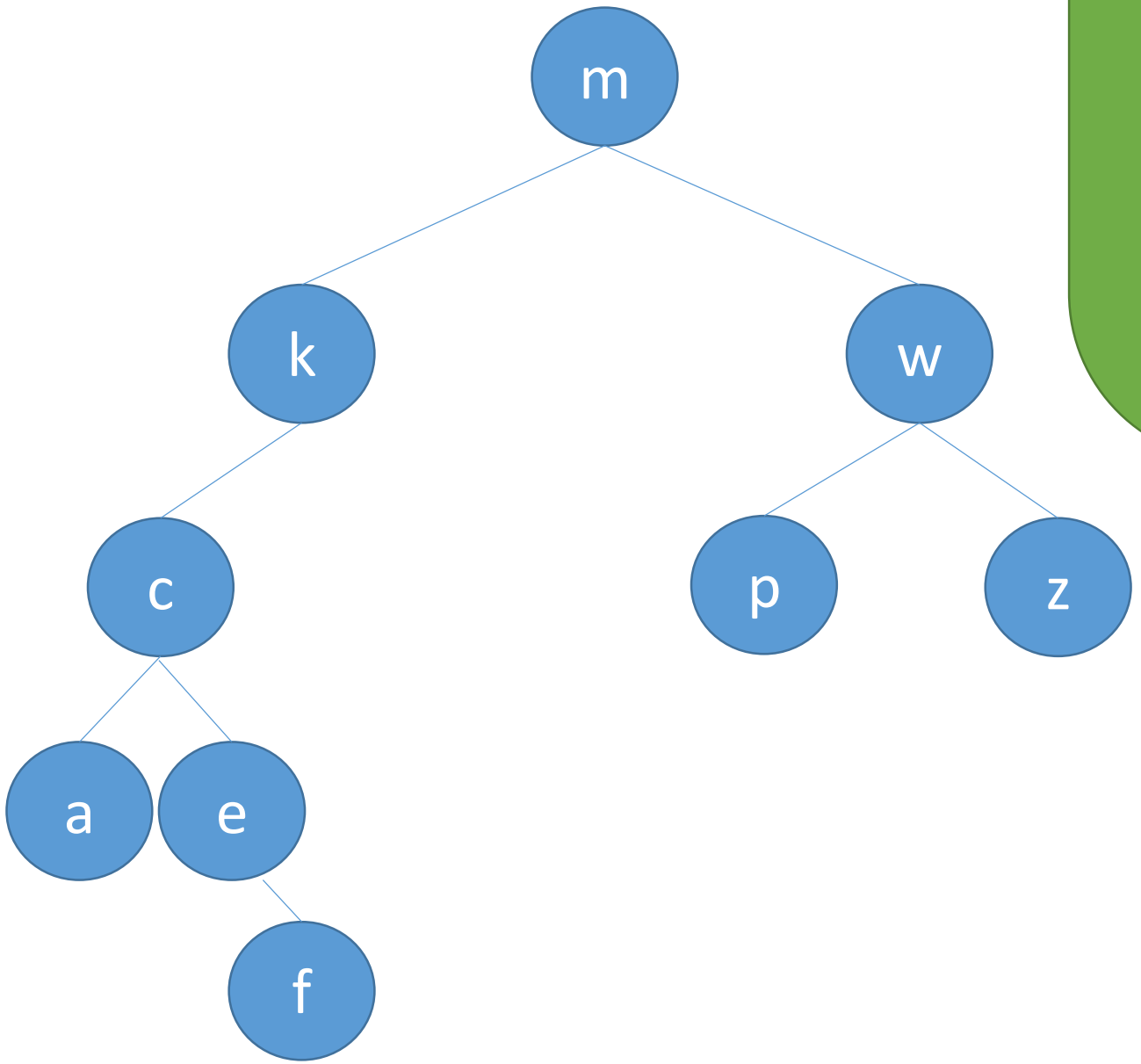


Draw this binary search tree.

Insert

If there is no node,
    Insert it there.
If it is less than the node,
    Go left. Try again.
If it is greater than the node,
    Go right. Try again.

```
BinarySearchTree t = new BinarySearchTree();
t.insert("m");
t.insert("k");
t.insert("c");
t.insert("e");
t.insert("f");
t.insert("a");
t.insert("w");
t.insert("z");
t.insert("p");
```



Draw this binary search tree.

Insert

If there is no node,
    Insert it there.
If it is less than the node,
    Go left. Try again.
If it is greater than the node,
    Go right. Try again.

```
private Node insertRec (Node root, String key) {
    if (root == null) {
      root = new Node (key);
      return root;
    }
    else if (key.compareTo (root.key) < 0)
       root.left = insertRec (root.left, key);
    else if (key.compareTo (root.key) > 0)
      root.right = insertRec (root.right, key);
    return root;
}
```

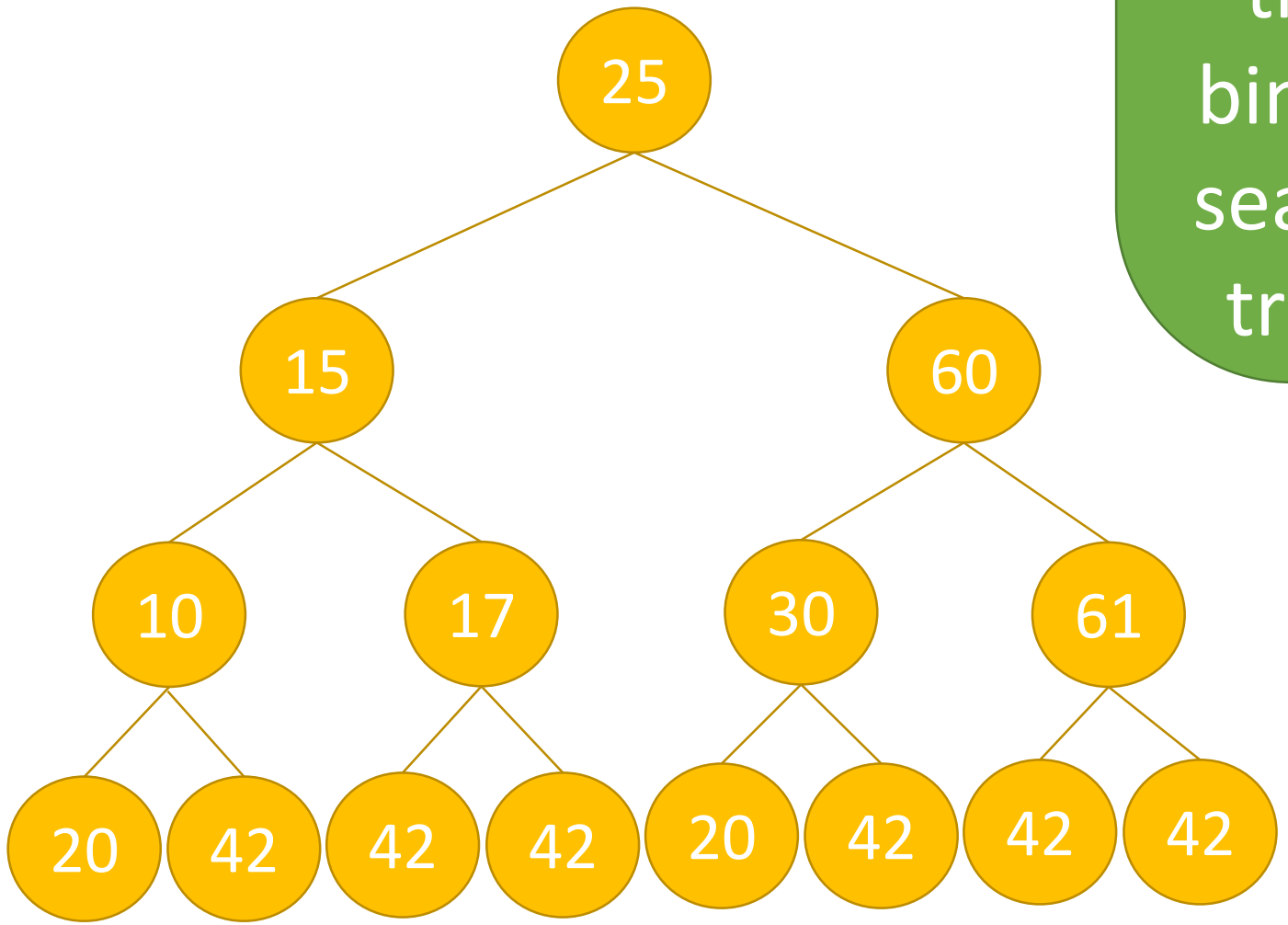
Why is a binary search tree useful?

- They are sorted and their order is useful
- They are very adaptable; easy to insert and delete
- They are FAST. Most functions: search, insert, delete, min, max are O(log n)

- Used to “index” databases so searching is quick
- Used to store dictionaries
- IP address indexing
- Expressing arithmetic expressions

```
BinarySearchTree t = new BinarySearchTree();
t.insert(25);
t.insert(15);
t.insert(60);
t.insert(30);
t.insert(61);
t.insert(42);
t.insert(10);
t.insert(17);
t.insert(20);
```

Draw this binary search tree.



Insert

If there is no node,
    Insert it there.
If it is less than the node,
    Go left. Try again.
If it is greater than the node,
    Go right. Try again.