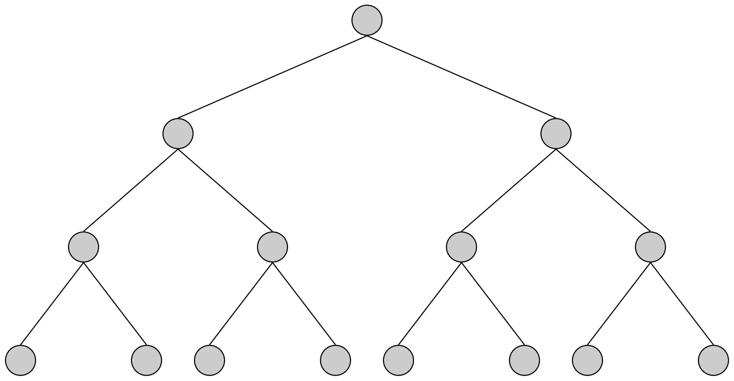
## 11.1 Introduction to Trees

A *tree* is a connected undirected graph with no simple circuits.

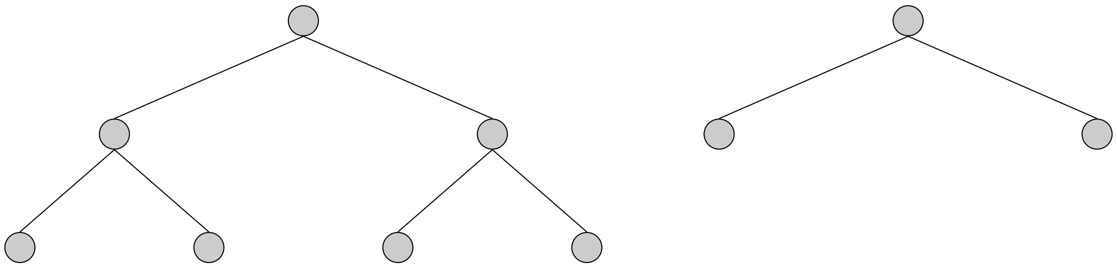


**Theorem 1**

An undirected graph is a tree if and only if there is a unique simple path between any two of its vertices.

**Forest**

A (not-necessarily-connected) undirected graph without simple circuits is called a *forest*.



### Rooted Trees

A *rooted tree* is a tree in which one vertex has been designated as the root and every edge is directed away from the root.

**Parent**

Suppose that $T$ is a rooted tree. If $v$ is a vertex in $T$ other than the root, the *parent* of $v$ is the unique vertex $u$ such that there is a directed edge from $u$ to $v$.

**Child**

If $u$ is the parent of $v$, then $v$ is called a *child* of $u$.

**Siblings**

Vertices with the same parent are called *siblings*.

**Ancestors**

The *ancestors* of a vertex other than the root are the vertices in the path from the root to this vertex, excluding the vertex itself and including the root.

**Descendants**

The *descendants* of a vertex $v$ are those vertices that have $v$ as an ancestor.

**Leaf**

A vertex of a tree is called a *leaf* if it has no children.

**Internal Vertices**

Vertices that have children are called *internal vertices*.

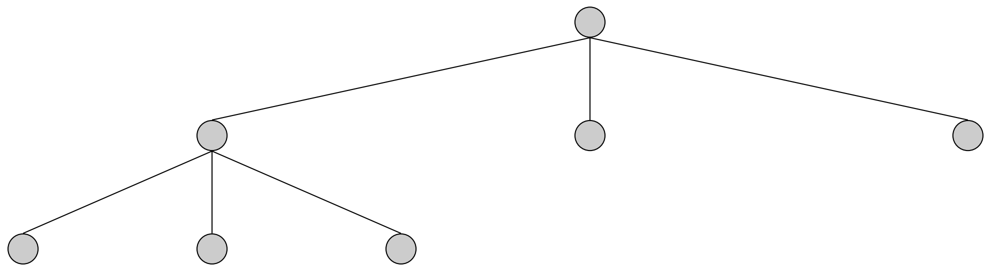
**Subtree**

If $a$ is a vertex in a tree, the *subtree* with $a$ as its root is the subgraph of the tree consisting of $a$ and its descendants and all edges incident to these descendants.

**$m$-ary Tree**

A rooted tree is called an *m-ary tree* if every internal vertex has no more than $m$ children. The tree is called a *full m-ary tree* if every internal vertex has exactly $m$ children.

Example of a full 3-ary tree:

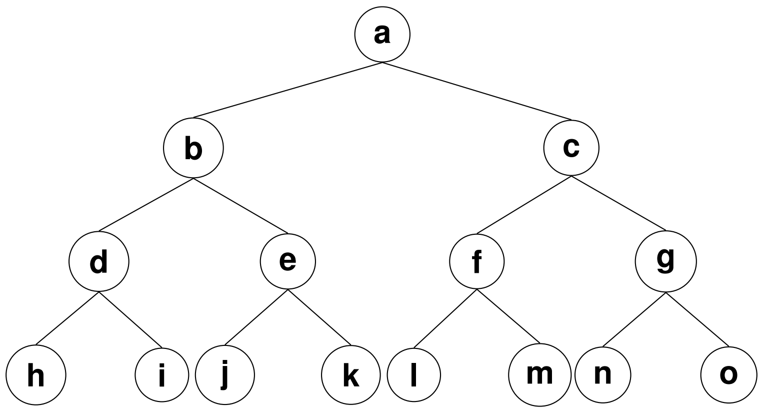


**Binary Tree**

An $m$-ary tree with $m = 2$ is called a *binary tree*.

**Ordered Root Tree**

An *ordered rooted tree* is a rooted tree where the children of each internal vertex are ordered.



**Left and Right Child**

In an ordered binary tree, the first child is called the *left child* and the second child is called the *right child*.

**Left and Right Subtree**

The tree rooted at the left child is called the *left subtree* and the tree rooted at the right child is called the *right subtree*.

**Theorem 2**

A tree with $n$ vertices has $n - 1$ edges.

**Theorem 3**

A full $m$-ary tree with $i$ internal vertices contains $n = mi + 1$ vertices.

**Theorem 4**

A full $m$-ary tree with

1. $n$ vertices has $i = \dfrac{n-1}{m}$ internal vertices and $l = \dfrac{(m-1)n+1}{m}$ leaves.
2. $i$ internal vertices has $n = mi + 1$ vertices and $l = (m-1)i + 1$ leaves.
3. $l$ leaves has $n = \dfrac{ml-1}{m-1}$ vertices and $i = \dfrac{l-1}{m-1}$ internal vertices.

**Level**

The *level* of a vertex $v$ in a rooted tree is the length of the unique path from the root to this vertex.

**Height**

The *height* of a rooted tree is the maximum of the levels of vertices. In other words, the height of a rooted tree is the length of the longest path from the root to any vertex.

**Balanced**

A rooted $m$-ary tree of height $h$ is *balanced* if all leaves are at levels $h$ or $h - 1$.

**Theorem 5**

There are at most $m^h$ leaves in an $m$-ary tree of height $h$.

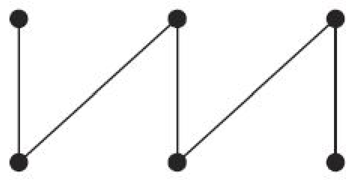
**Corollary 1**

If an $m$-ary tree of height $h$ has $l$ leaves, then $h \geq \lceil \log_m l \rceil$. If the $m$-ary tree is full and balanced, then $h = \lceil \log_m l \rceil$.

**11.1 pg. 775 # 1**

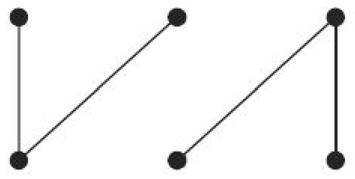
Which of these graphs are trees?

a )



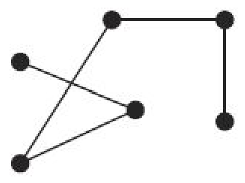
This graph is a tree because it is connected has no simple circuits.

b )



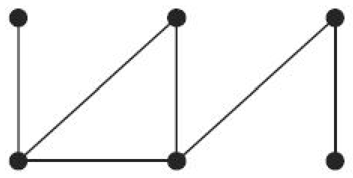
Not a tree because it is not connected.

c )



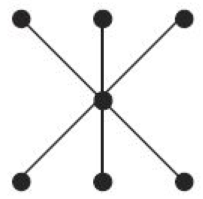
This graph is a tree because it is connected has no simple circuits.

d )



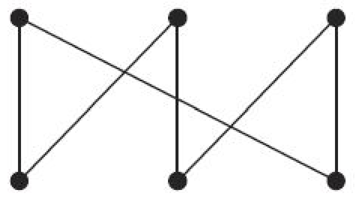
Not a tree because it has a simple circuit.

e )



This graph is a tree because it is connected has no simple circuits.

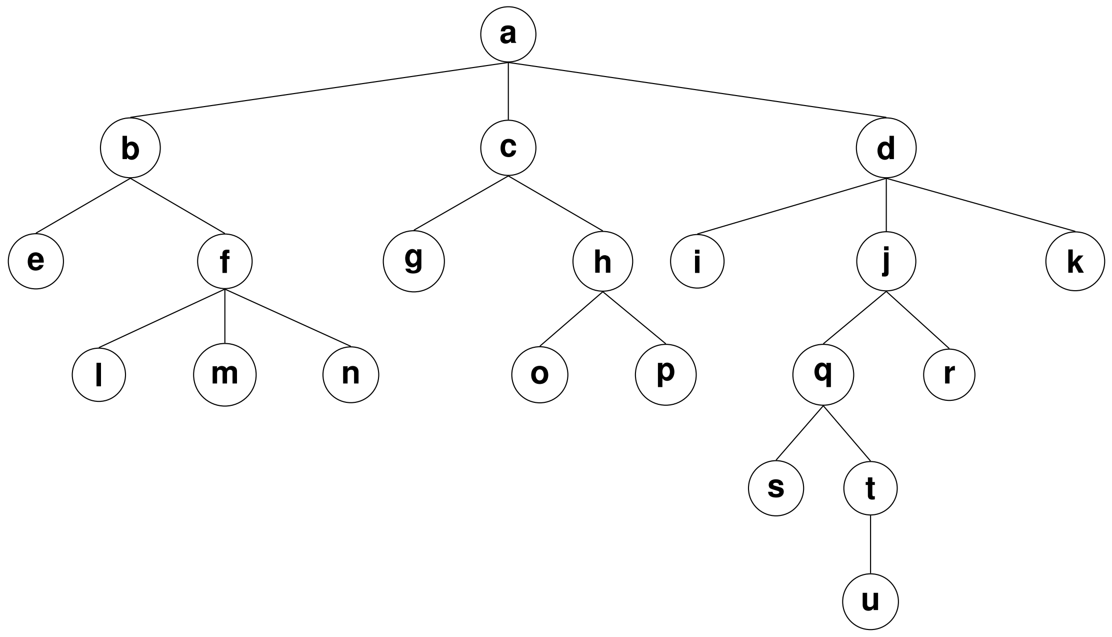
f )



Not a tree because it has a simple circuit.

**11.1 pg. 775 # 3**

Answer these questions about the rooted tree illustrated



a) Which vertex is a root?

   $a$ is the root of the tree.

b) Which vertices are internal?

   The internal vertices are $a, b, c, d, f, h, j, q, t$.

c) Which vertices are leaves?

   The leaves are $e, g, i, k, l, m, n, o, p, r, s, u$.

d) Which vertices are children of $j$?

   The children of $j$ are $q$ and $r$.

e) Which vertex is the the parent of $h$?

   The parent of $h$ is $c$.

f) Which vertices are siblings of $o$?

   The sibling of $o$ is $p$.

g) Which vertices are ancestors of $m$?

   The ancestors of $m$ are $f, b, a$.

h) Which vertices are descendants of $b$?

   The descendants of $b$ are $e, f, l, m, n$.

**11.1 pg. 755 # 5**

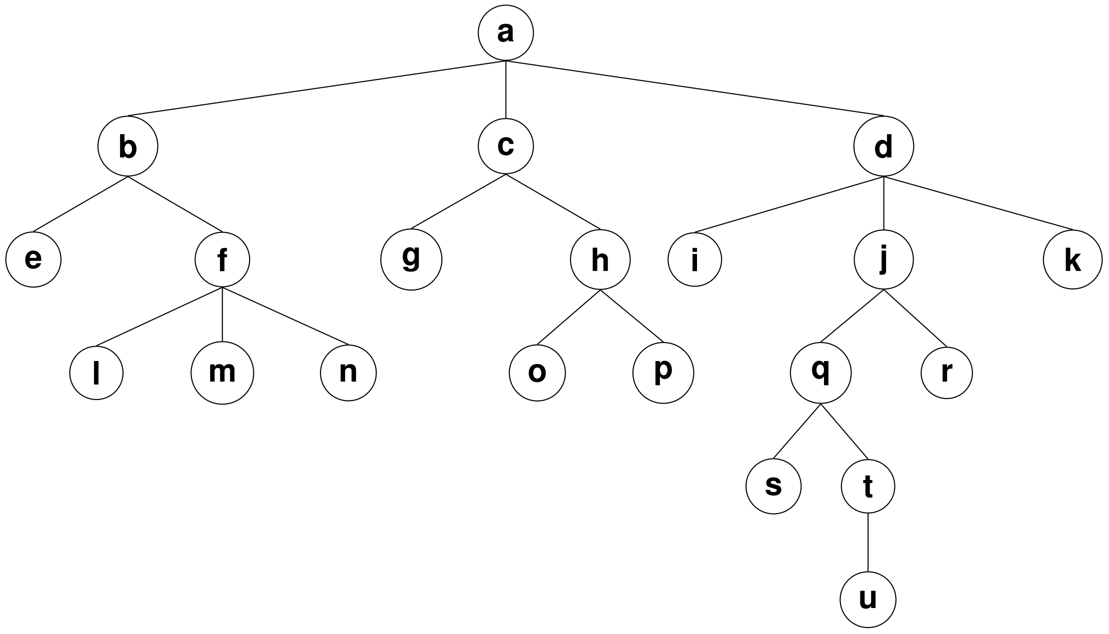
Is the rooted tree in Exercise 3 a full $m$-ary tree for some positive integer $m$?

This $m$-ary tree is valid for all $m \geq 3$. However, the tree is not a full $m$-ary tree because it has vertices that have 3, 2, or 1 children.

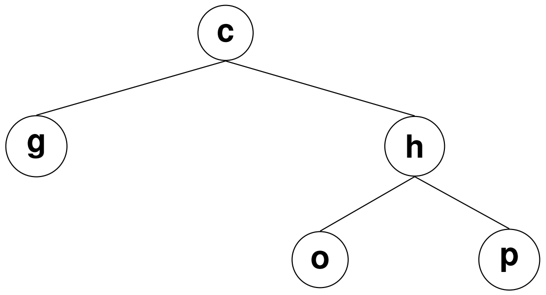
**11.1 pg. 755 # 9**

Draw the subtree of the tree in Exercise 3 that is rooted at

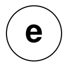
a) $a$.



b) $c$.



c) $e$.



**11.1 pg. 756 # 17**

How many edges does a tree with $10000$ vertices have?

Use theorem 2. A tree with $n$ vertices has $n - 1$ edges.
$10000 - 1 = 9999$ edges.

**11.1 pg. 756 # 19**

How many edges does a full binary tree with $1000$ internal vertices have?

A full binary tree has two edges for each internal vertex. So we'll just multiply the number of internal vertices by the number of edges.
$1000 \cdot 2 = 2000$ edges

**11.1 pg. 756 # 21**

Suppose 1000 people enter a chess tournament. Use a rooted tree model of the tournament to determine how many games must be played to determine a champion, if a player is eliminated after one loss and games are played until only one entrant has not lost. (Assume there are no ties.)

We can model this tournament with a full binary tree. We know we have 1000 leaves for this tree because we have 1000 people. Each internal vertex will represent the winner of the game played by its children. The root will be the winner of the tournament. By Theorem 4(3) with $m = 2$ and $l = 1000$. We know that:

$$i = \frac{l-1}{m-1}$$
$$= \frac{1000-1}{2-1}$$
$$= \frac{999}{1}$$
$$= 999$$

We have 999 internal vertices, so we know 999 games must be played to determine the champion.