THE UNIVERSITY OF VERMONT
COLLEGE OF ENGINEERING &
MATHEMATICAL SCIENCES

TREES

AN INTRODUCTION

CS 124 / Department of Computer Science

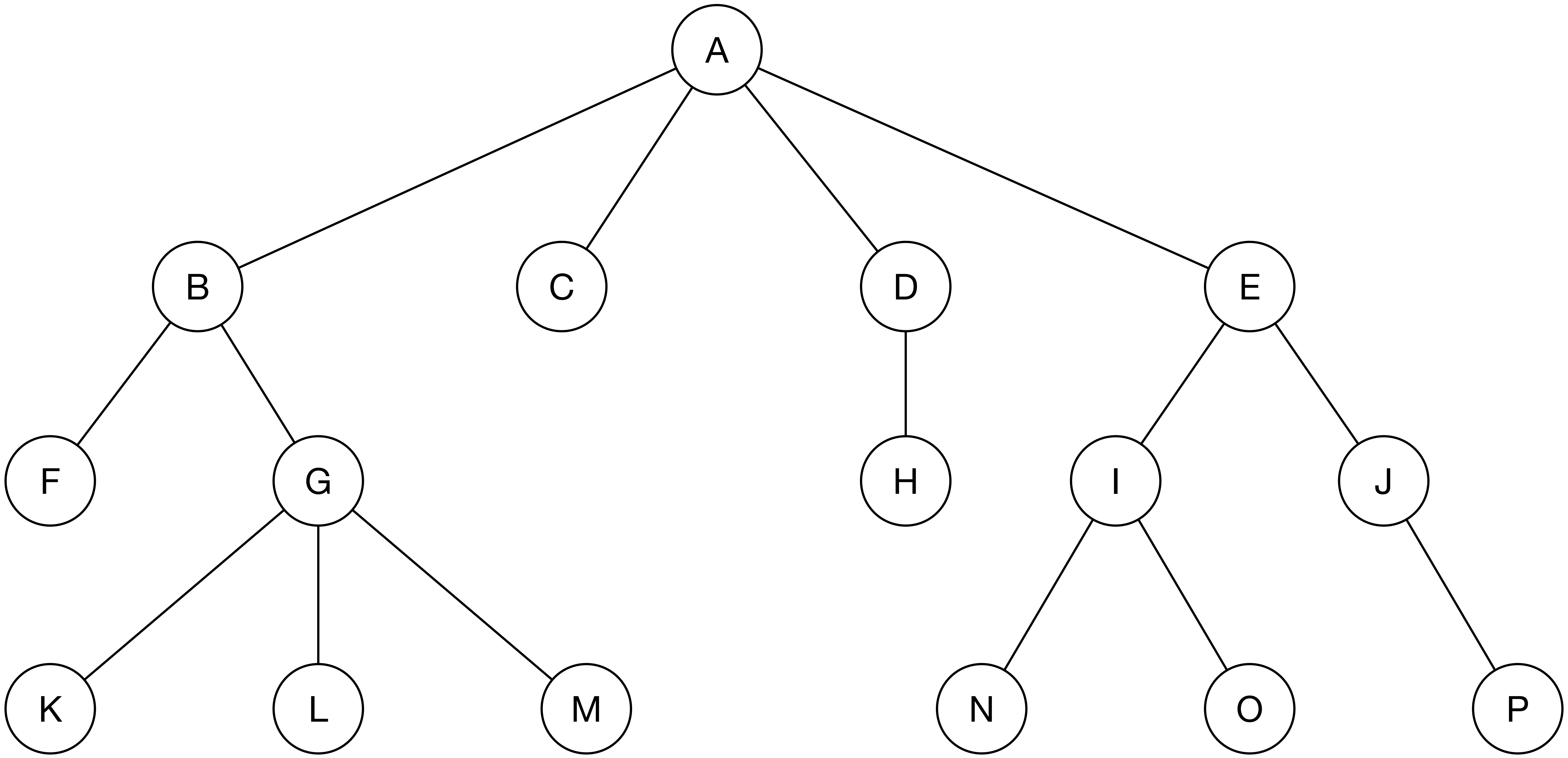
Trees

Trees are ubiquitous in programming and computer science. They are an essential data structure used in many applications. Moreover, there are many kinds of trees, each kind with special properties. These properties are either structural, or with regard to the kind of data that trees can store.

Before we get into the details of applications or the varieties of trees, we need to understand the basics.

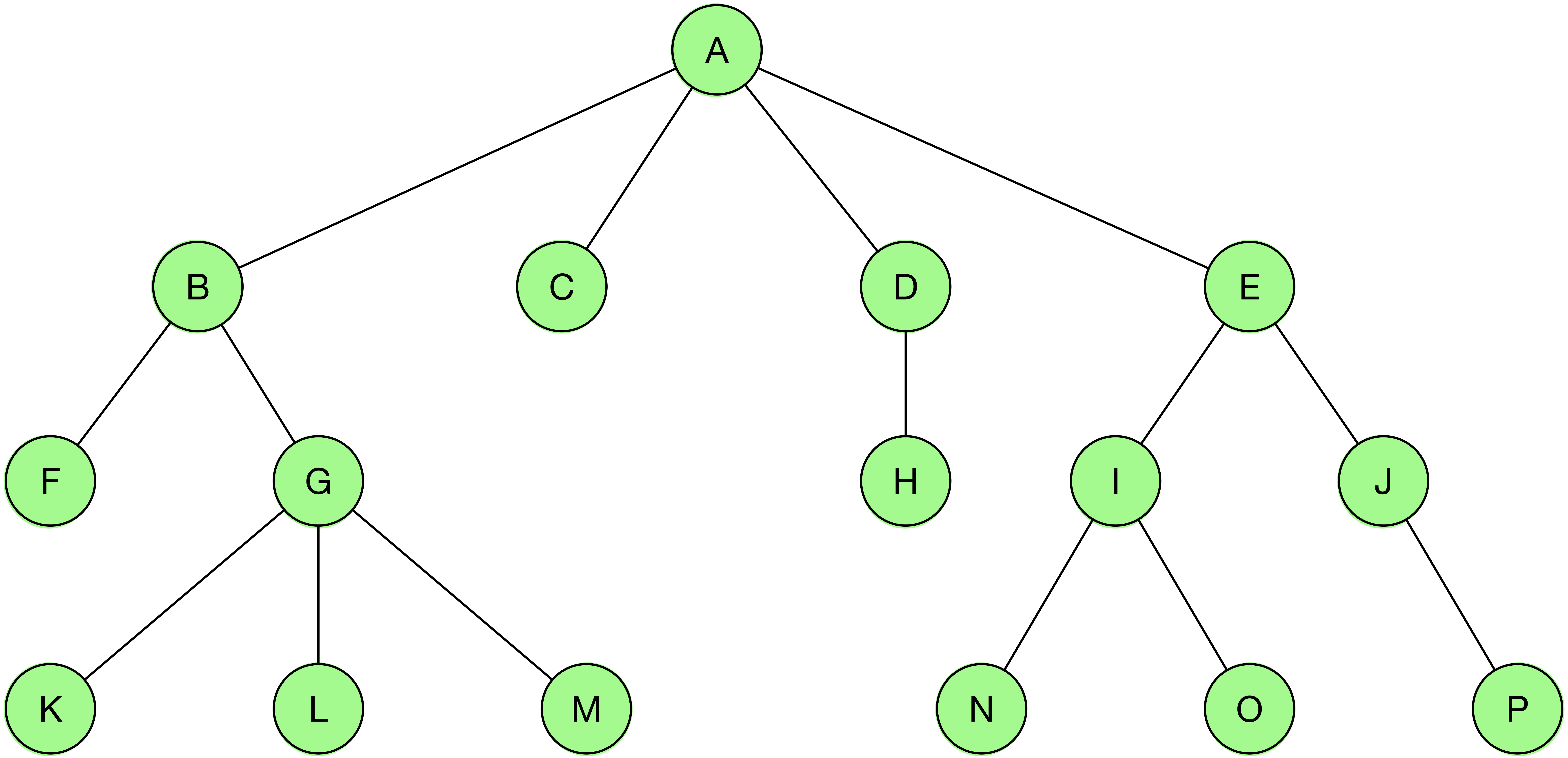
What is a tree?

A tree is a structure consisting of *nodes* (a.k.a. vertices) and *edges*.



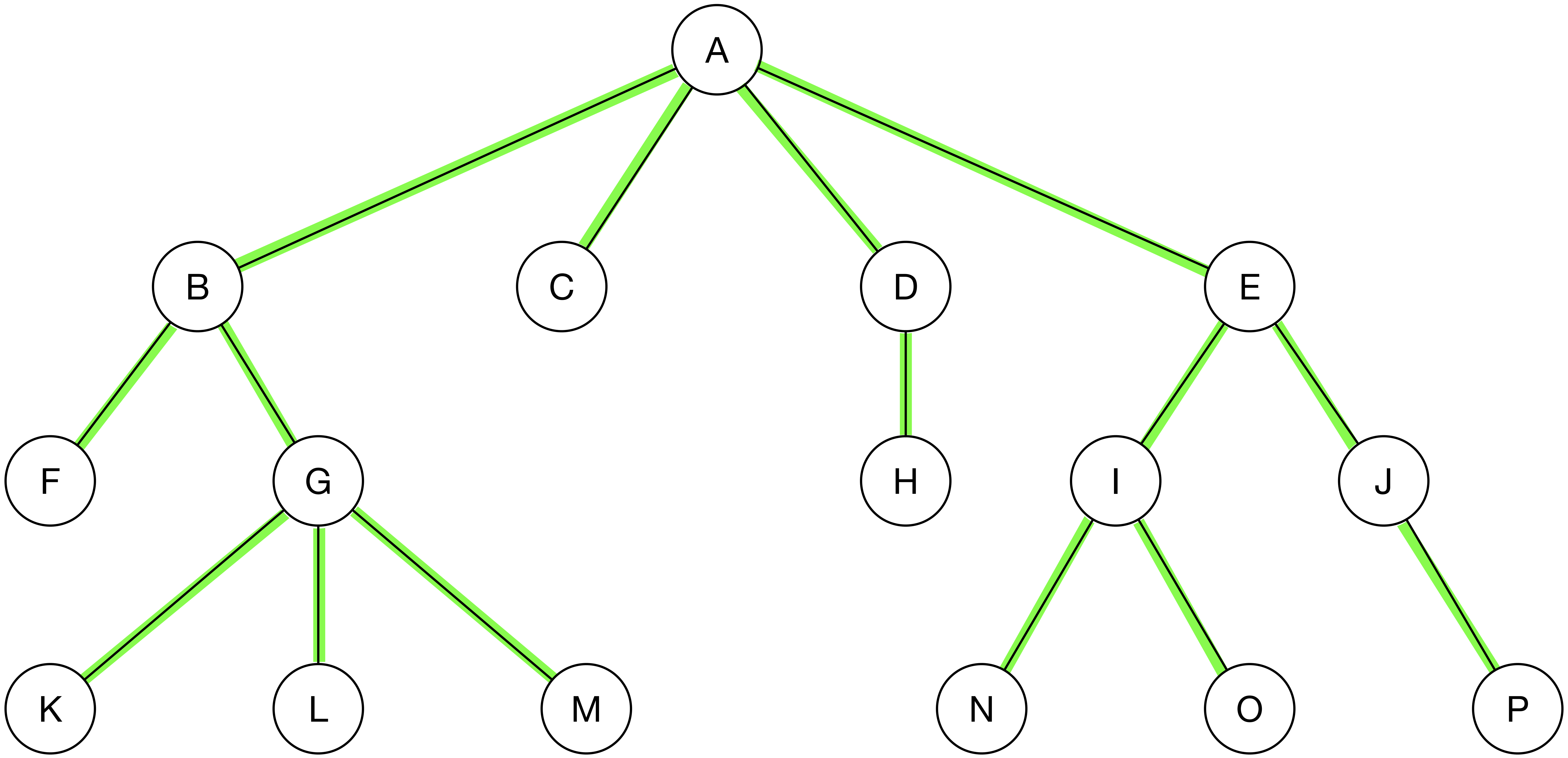
What is a tree?

These are the nodes...



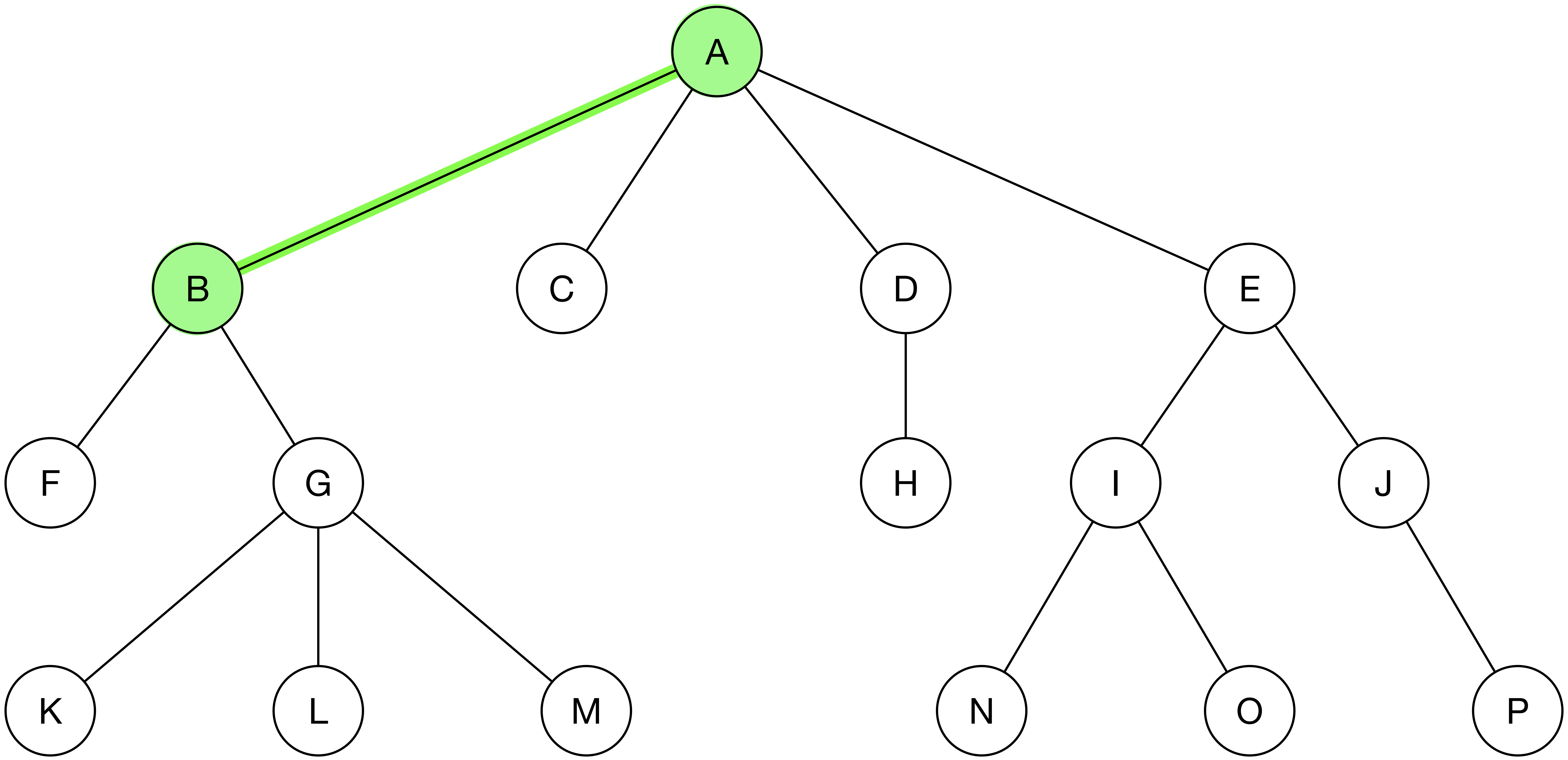
What is a tree?

...and these are the edges.



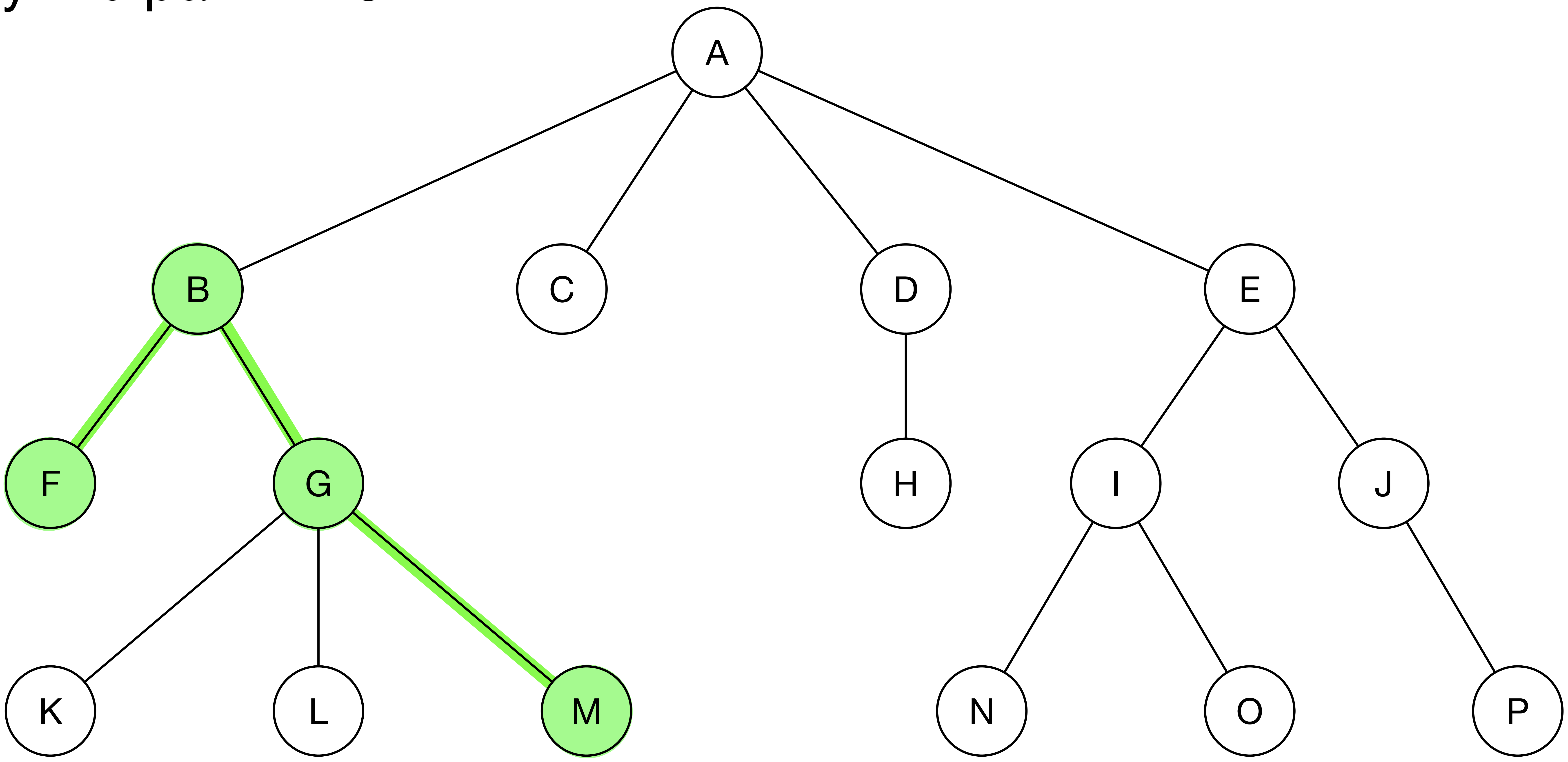
What is a tree?

An edge connects two nodes. Every edge has exactly two endpoints (nodes).



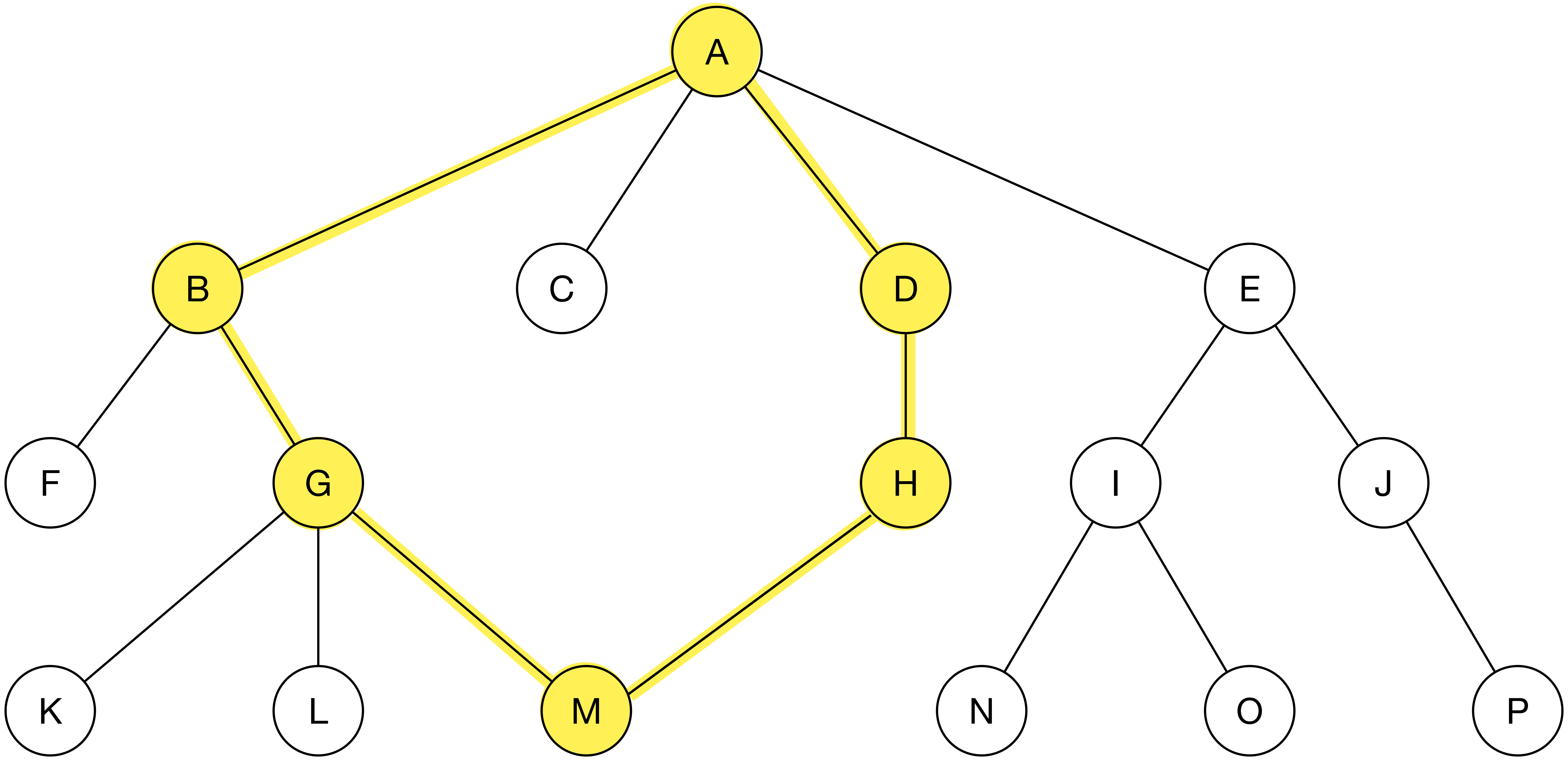
What is a tree?

A *path* is a collection of edges that joins two nodes. Here the nodes F and M are joined by the path FBGM.



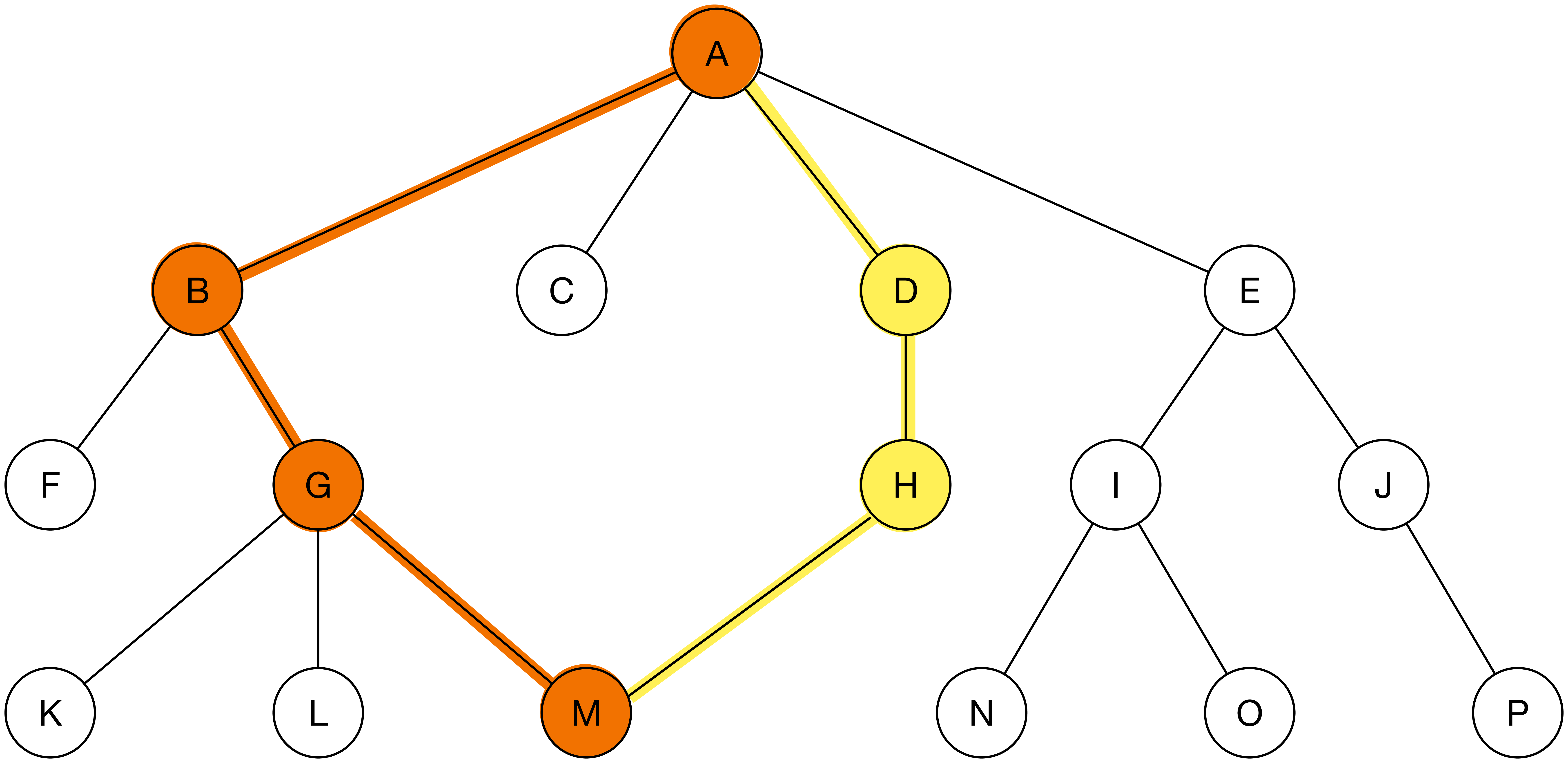
What is a tree?

However, there must be only one path between any pair of nodes.



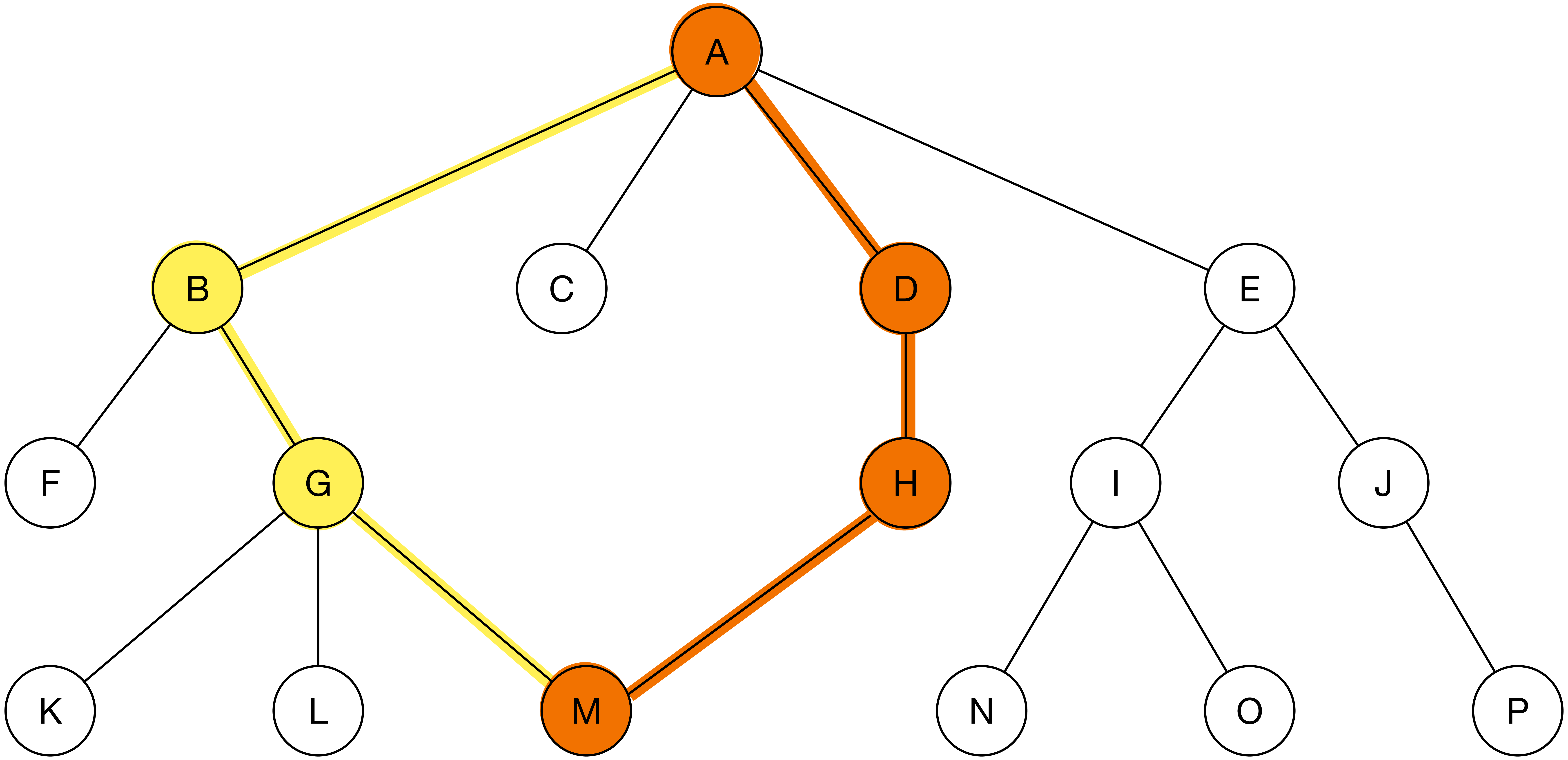
What is a tree?

Here's one path from A to M passing through B and G...



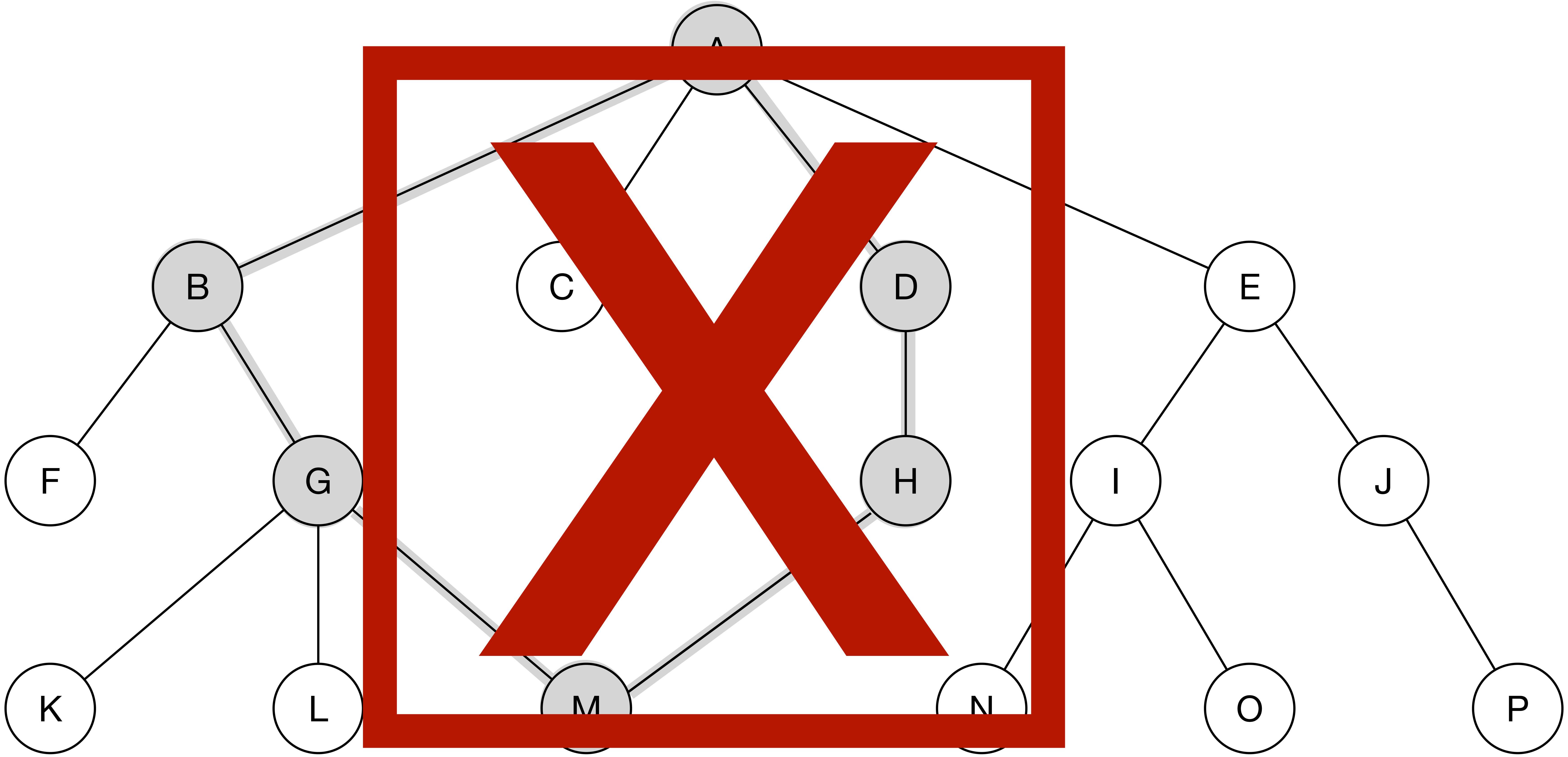
What is a tree?

...and here's another passing through D and H.



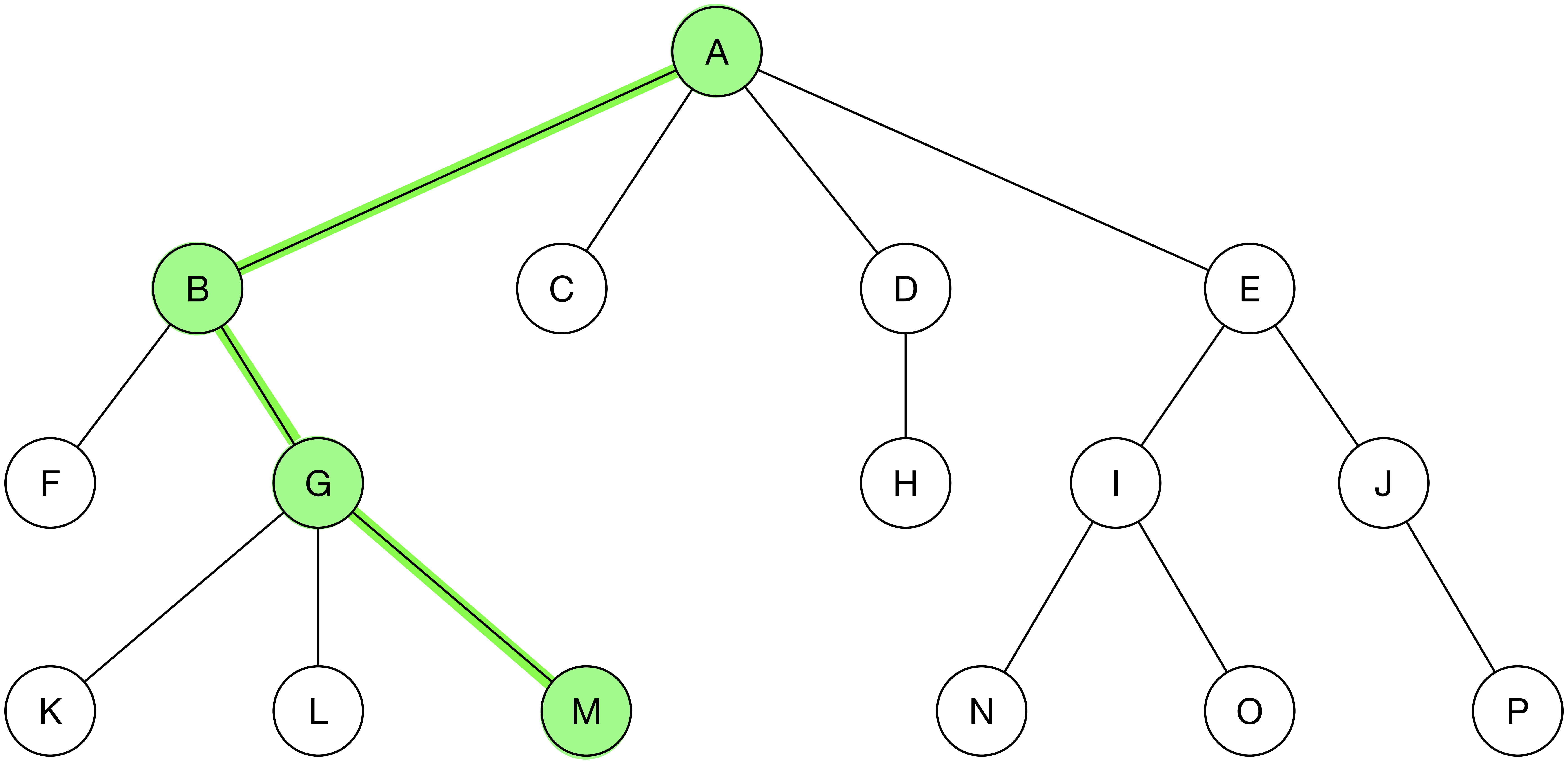
What is a tree?

However, there must be only one path between any pair of nodes.



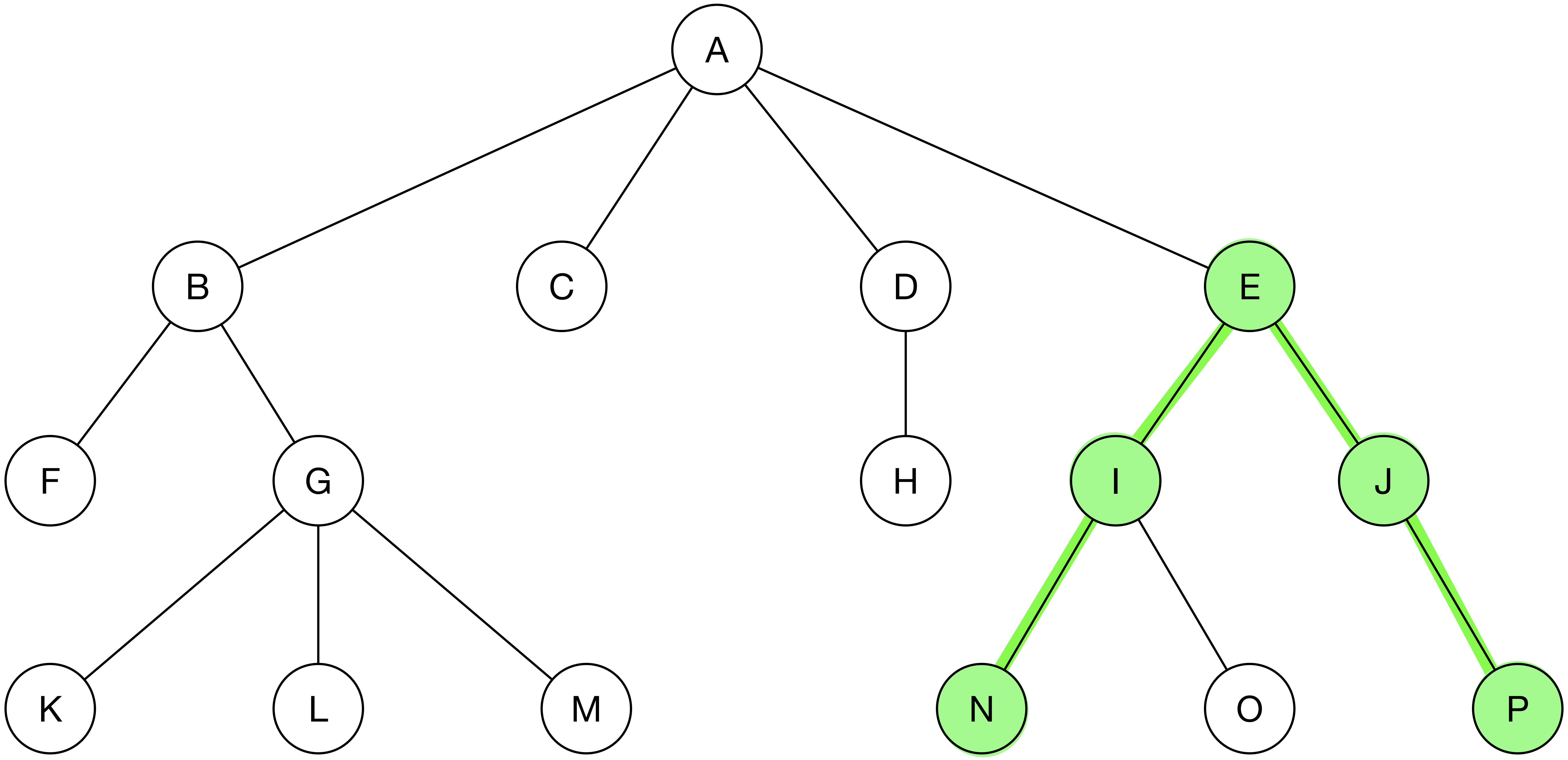
What is a tree?

Trees are *connected*. This means that a unique path exists between every pair of nodes in the tree.



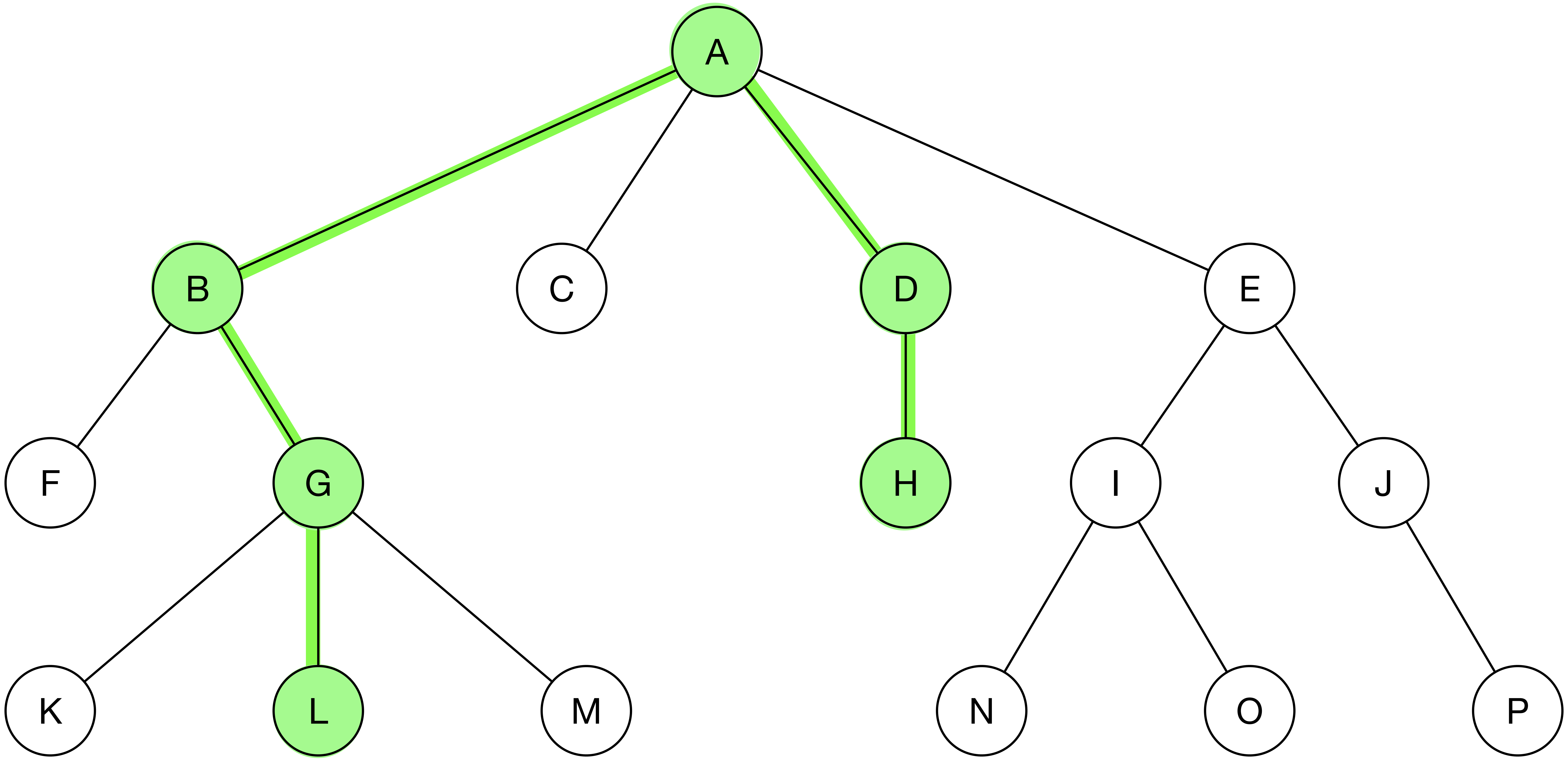
What is a tree?

Trees are *connected*. This means that a path exists between every pair of nodes in the tree.



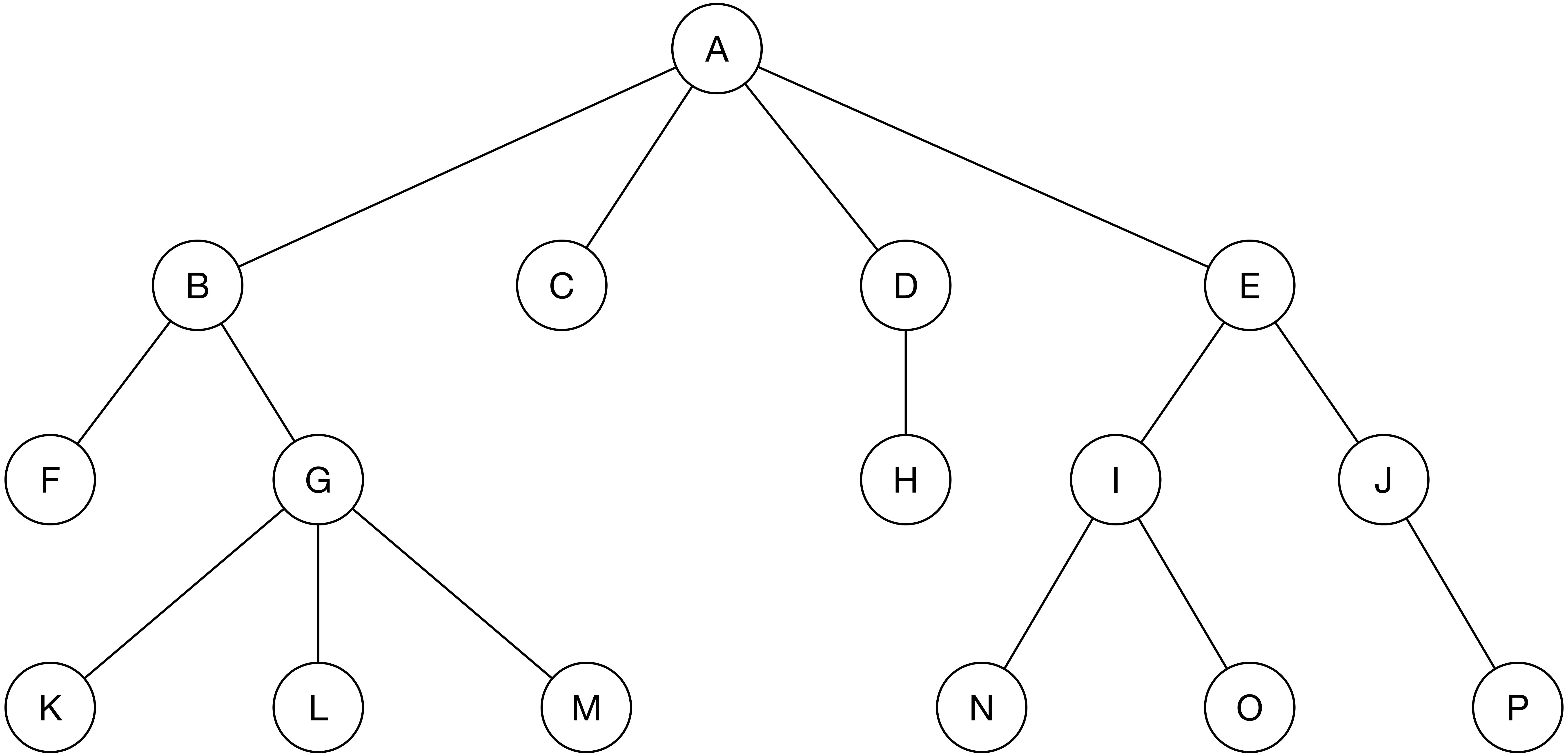
What is a tree?

Trees are *connected*. This means that a path exists between every pair of nodes in the tree.



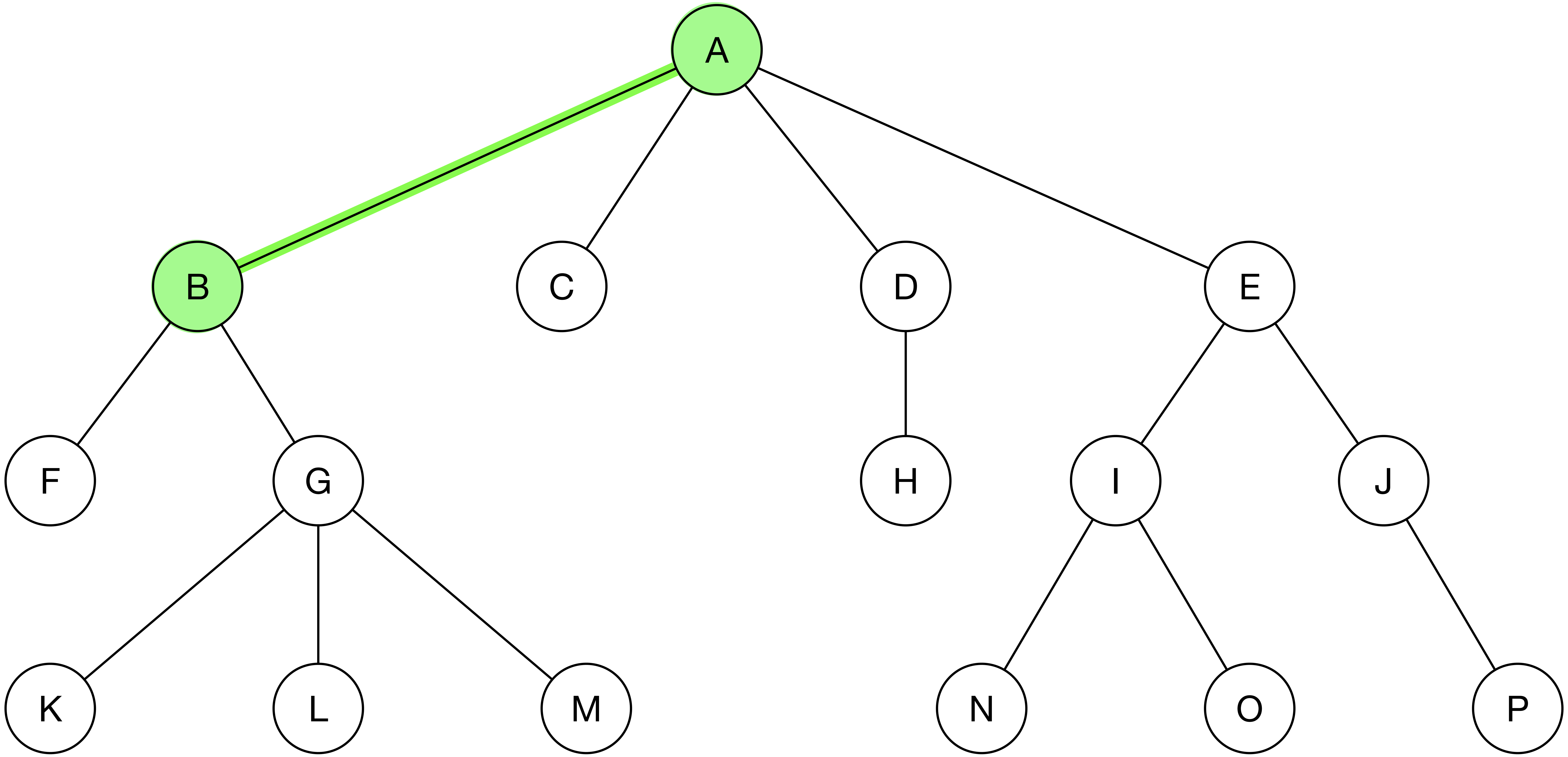
Distance

The *distance* between two nodes is the number of edges we must traverse to get from one to the other. Similarly we may speak of the *length* of a path.



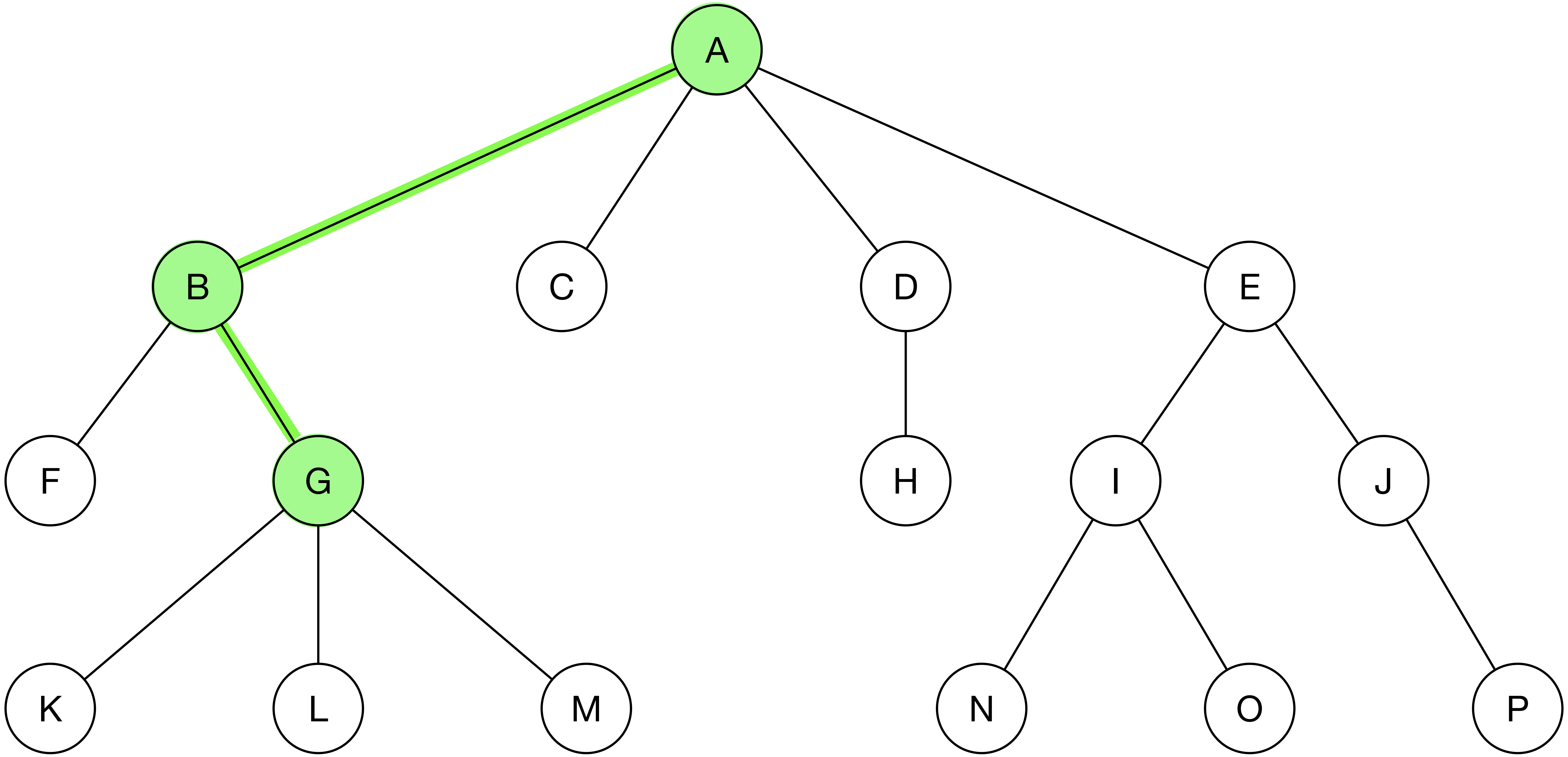
Distance

The distance from A to B, written d(A, B) = 1



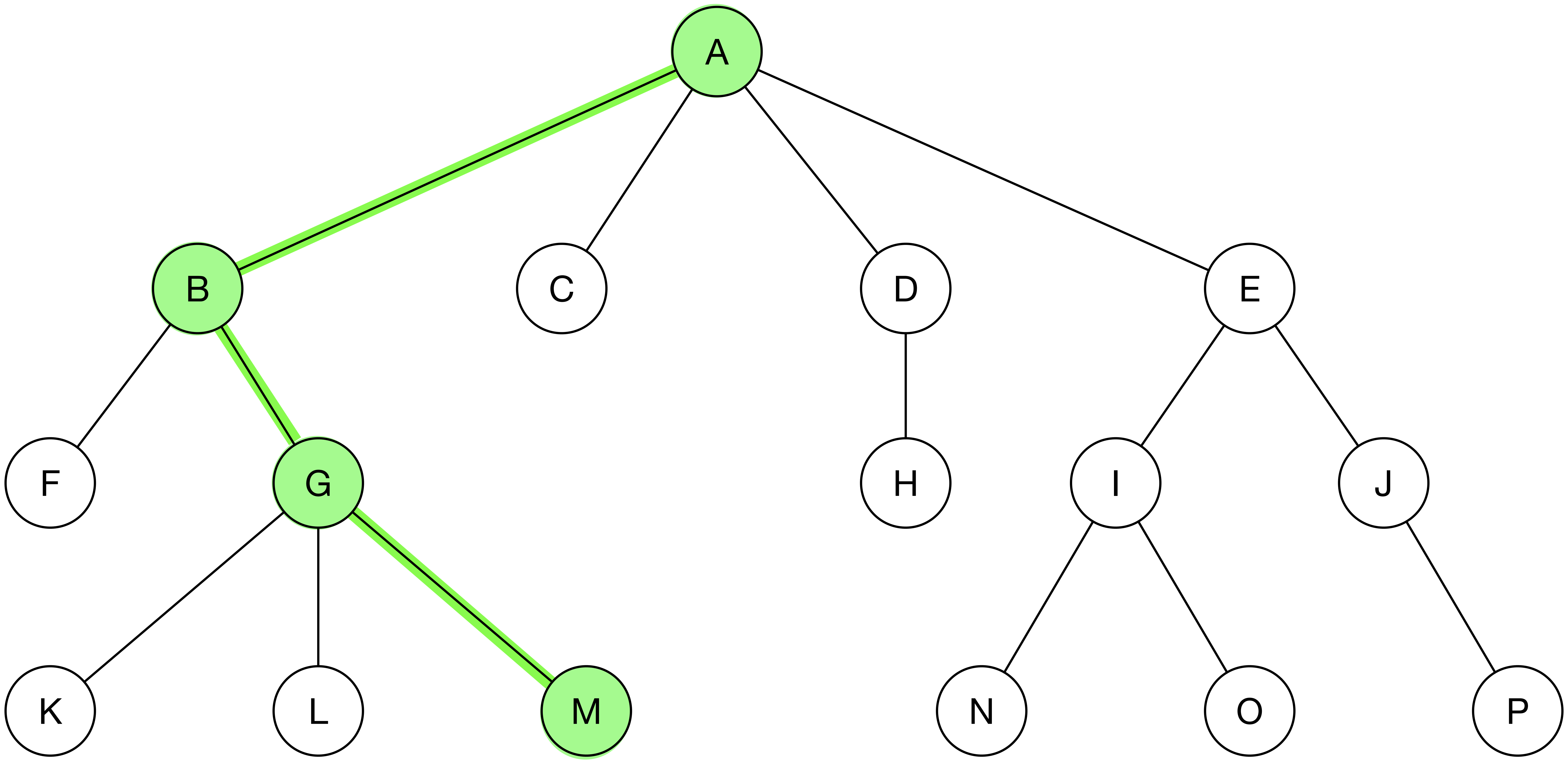
Distance

d(A, G) = 2



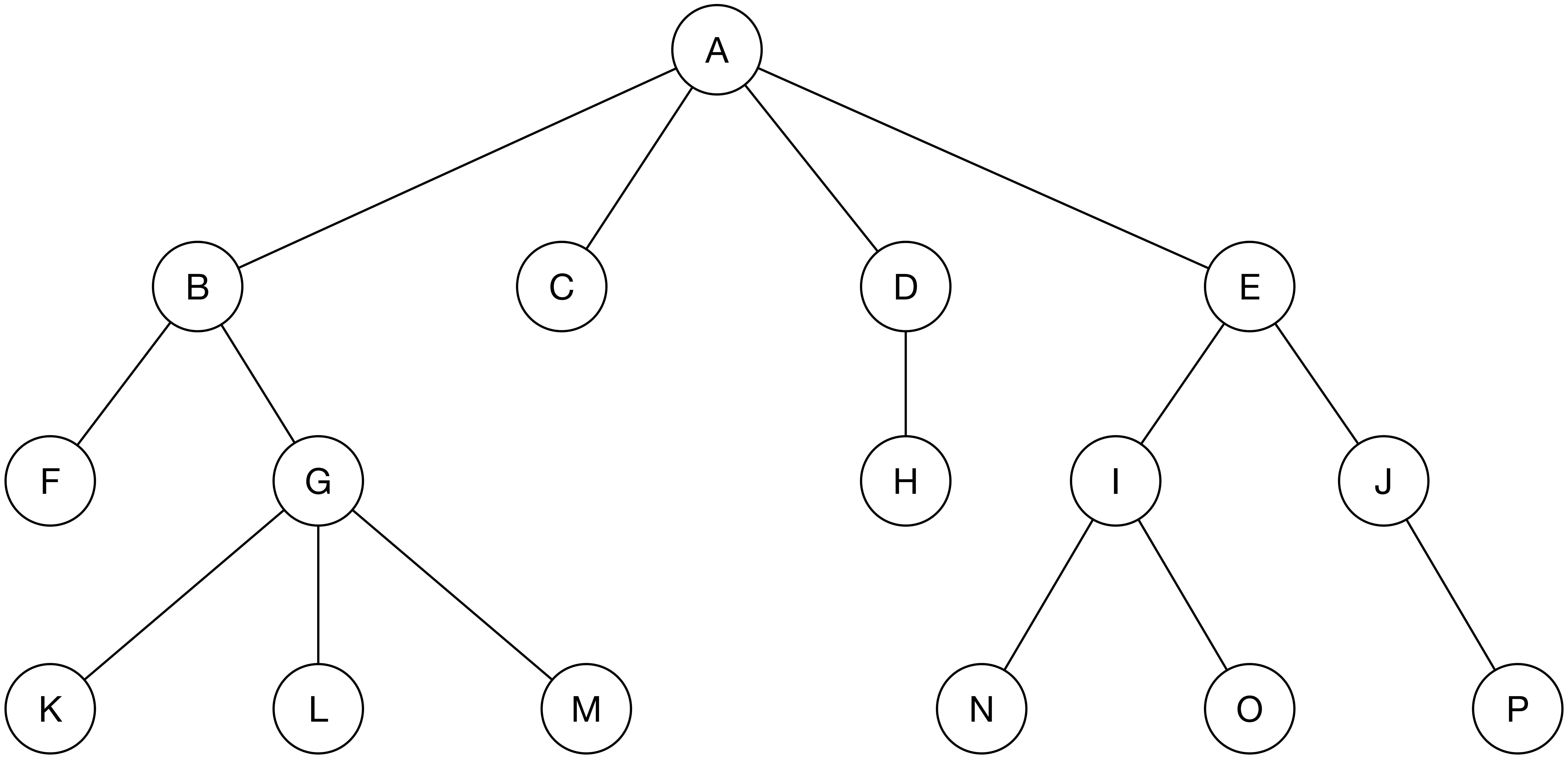
Distance

$d(A, M) = 3$



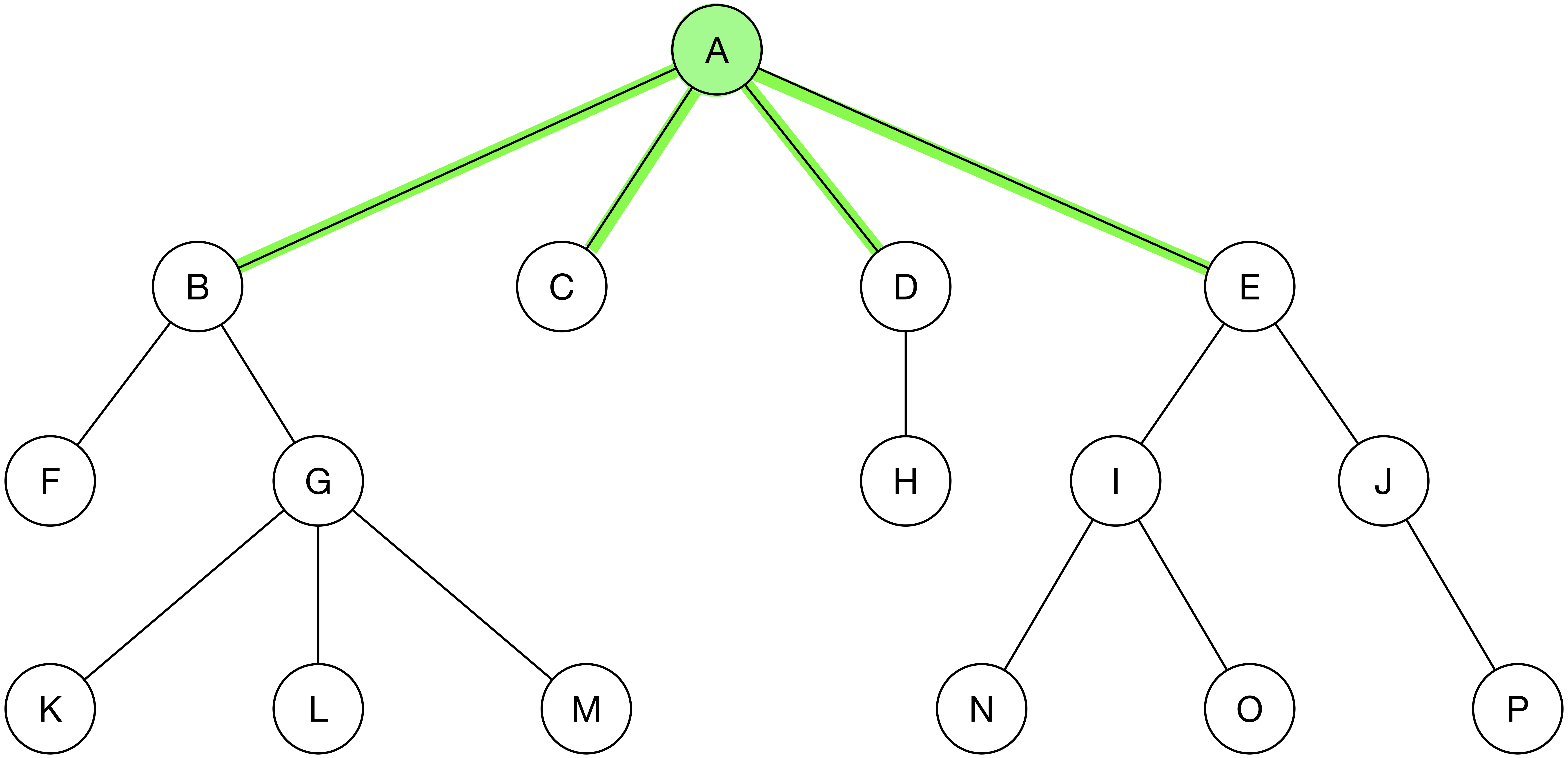
Some terminology

The *degree* of a node is the number of edges that are *incident* to it.



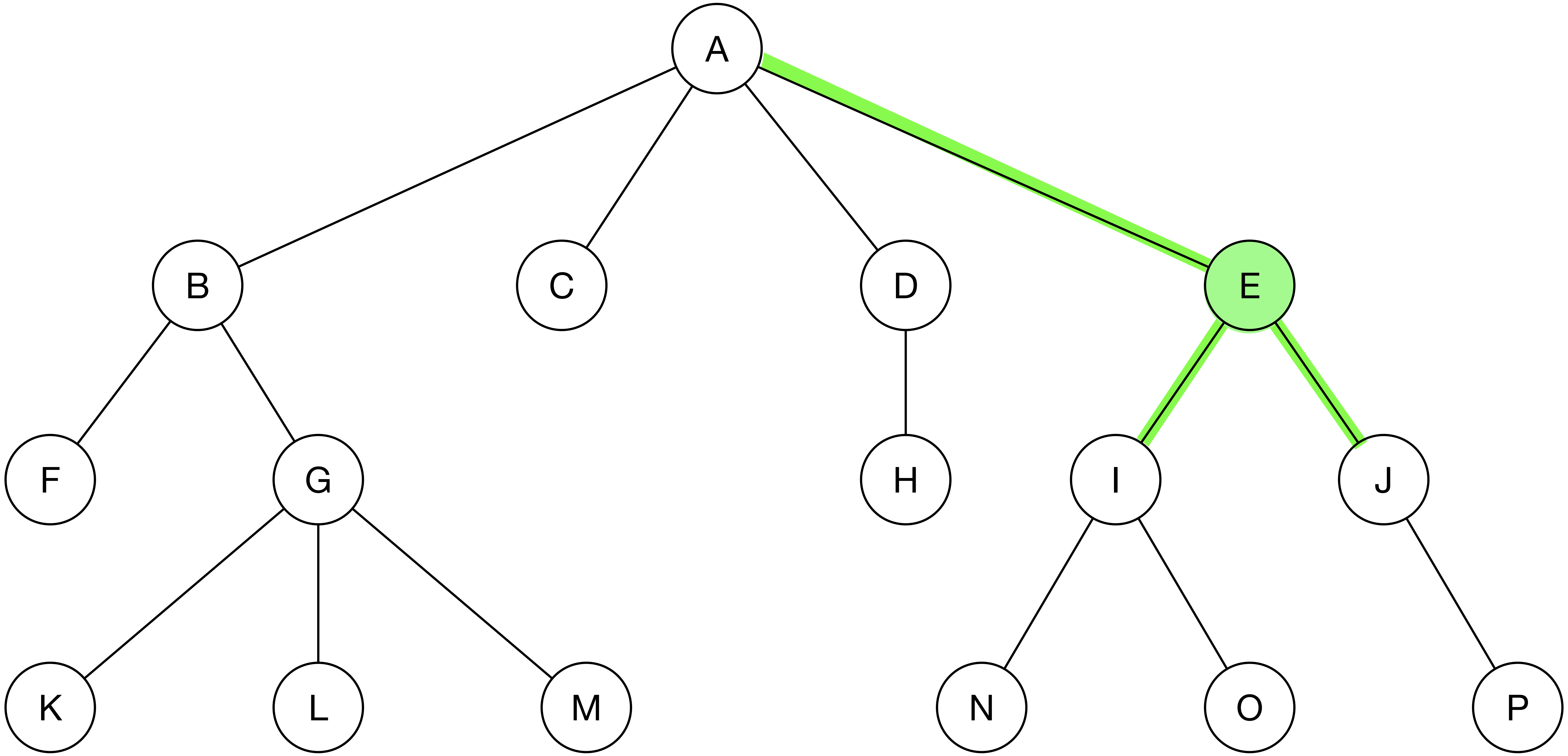
Some terminology

This node has degree 4



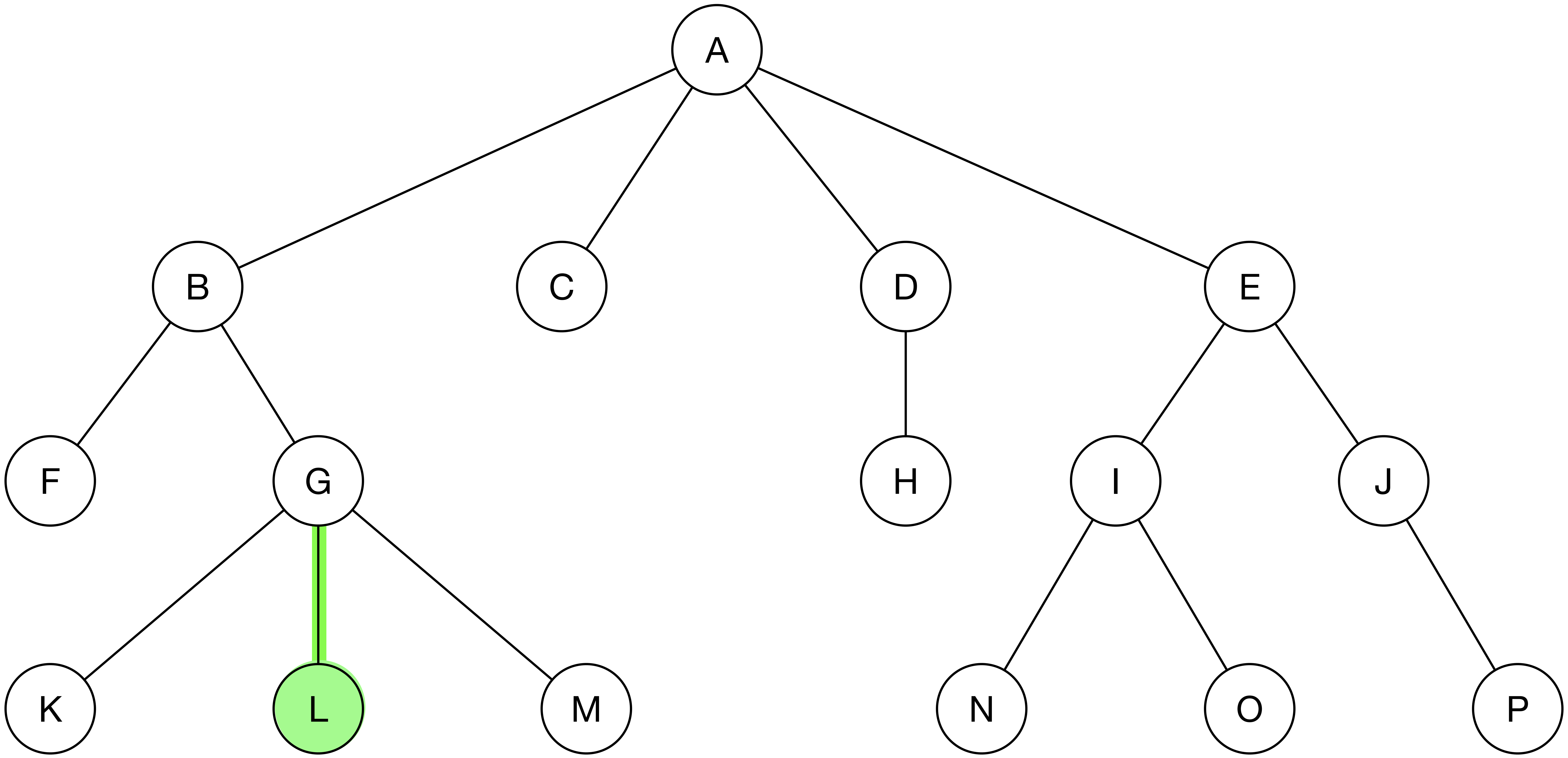
Some terminology

This node has degree 3



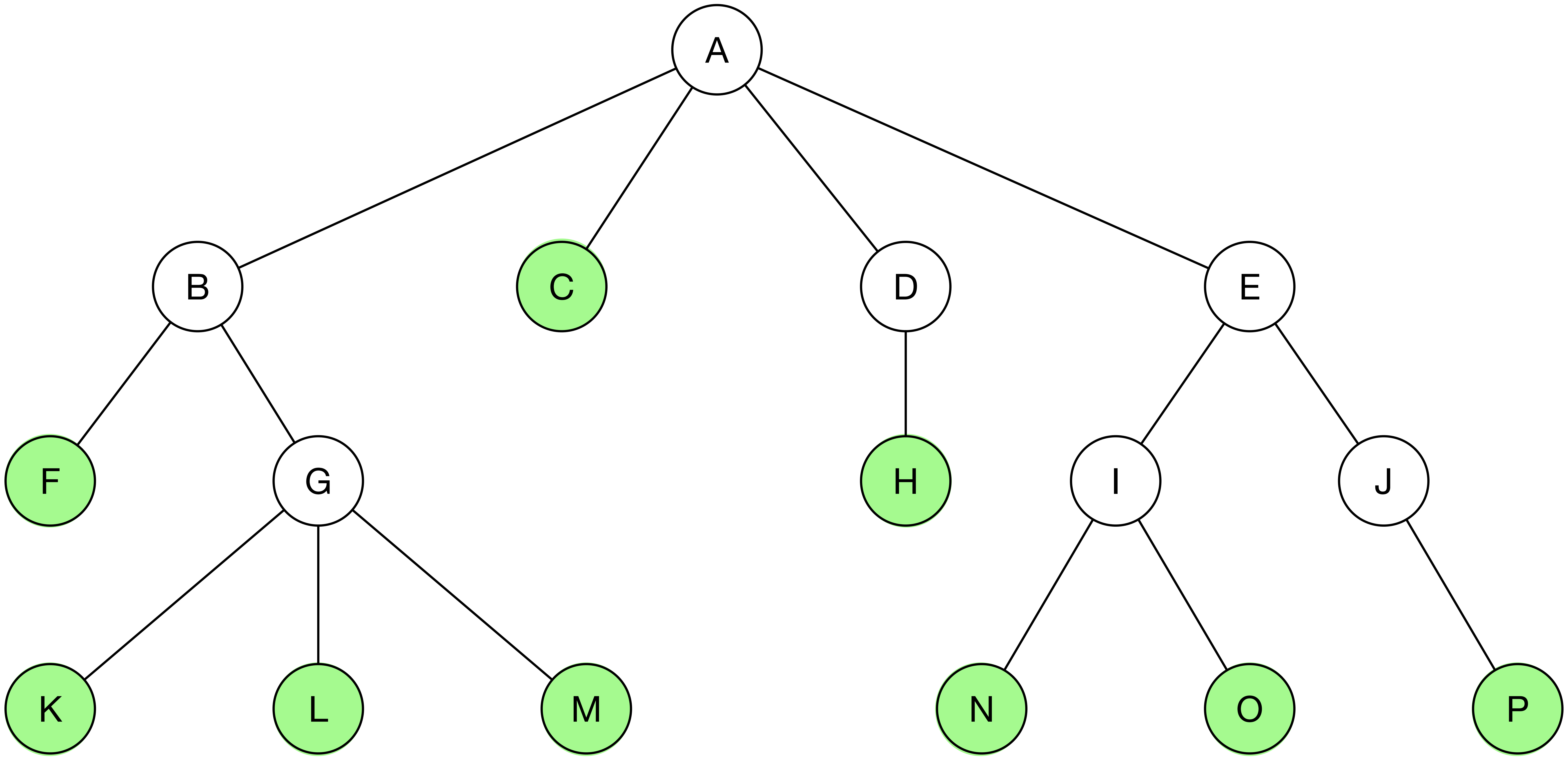
Some terminology

This node has degree 1



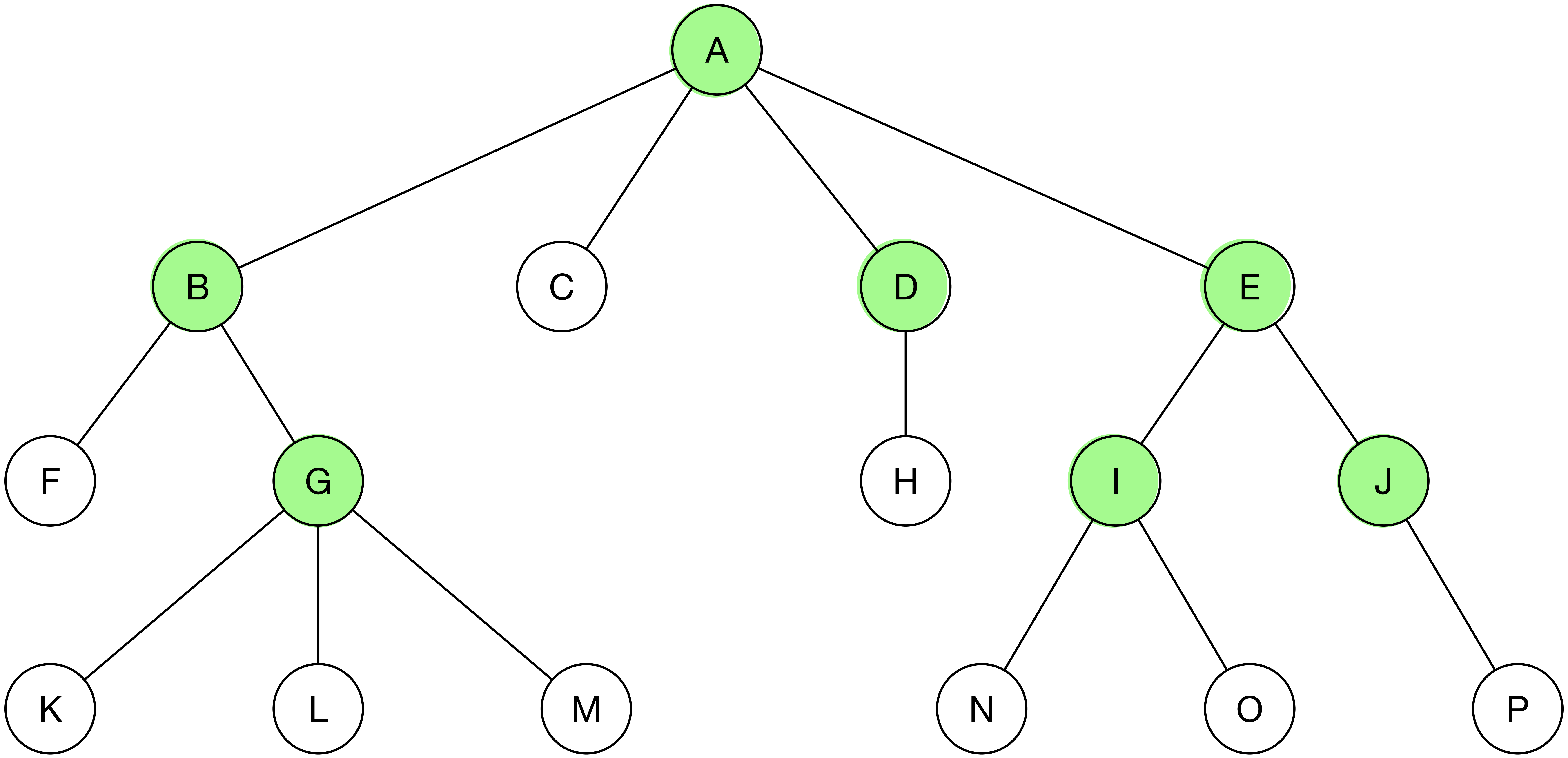
Some terminology

Nodes with degree 1 are called *leaves*.



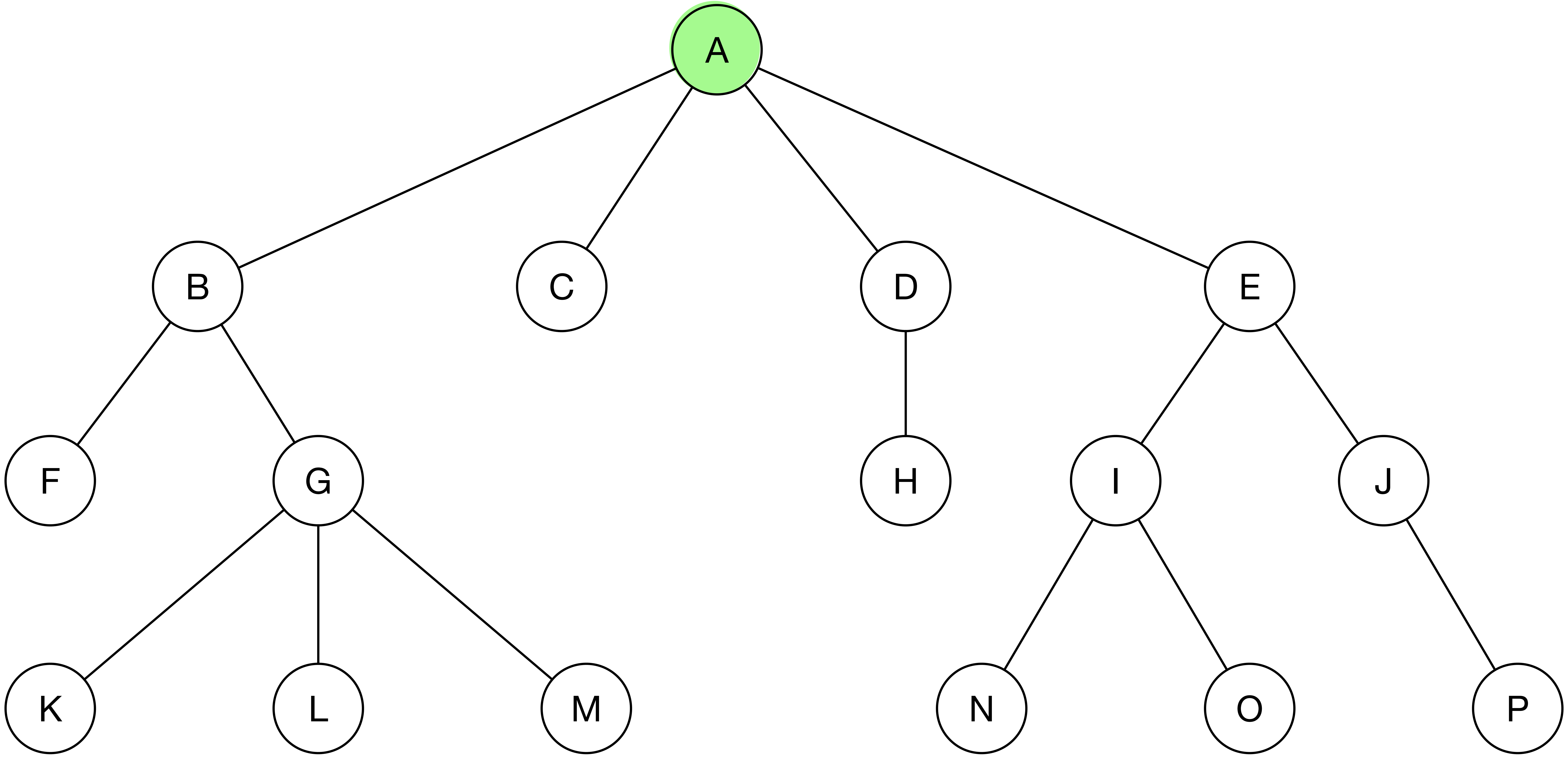
Some terminology

Nodes that are not leaves are called *interior nodes*.



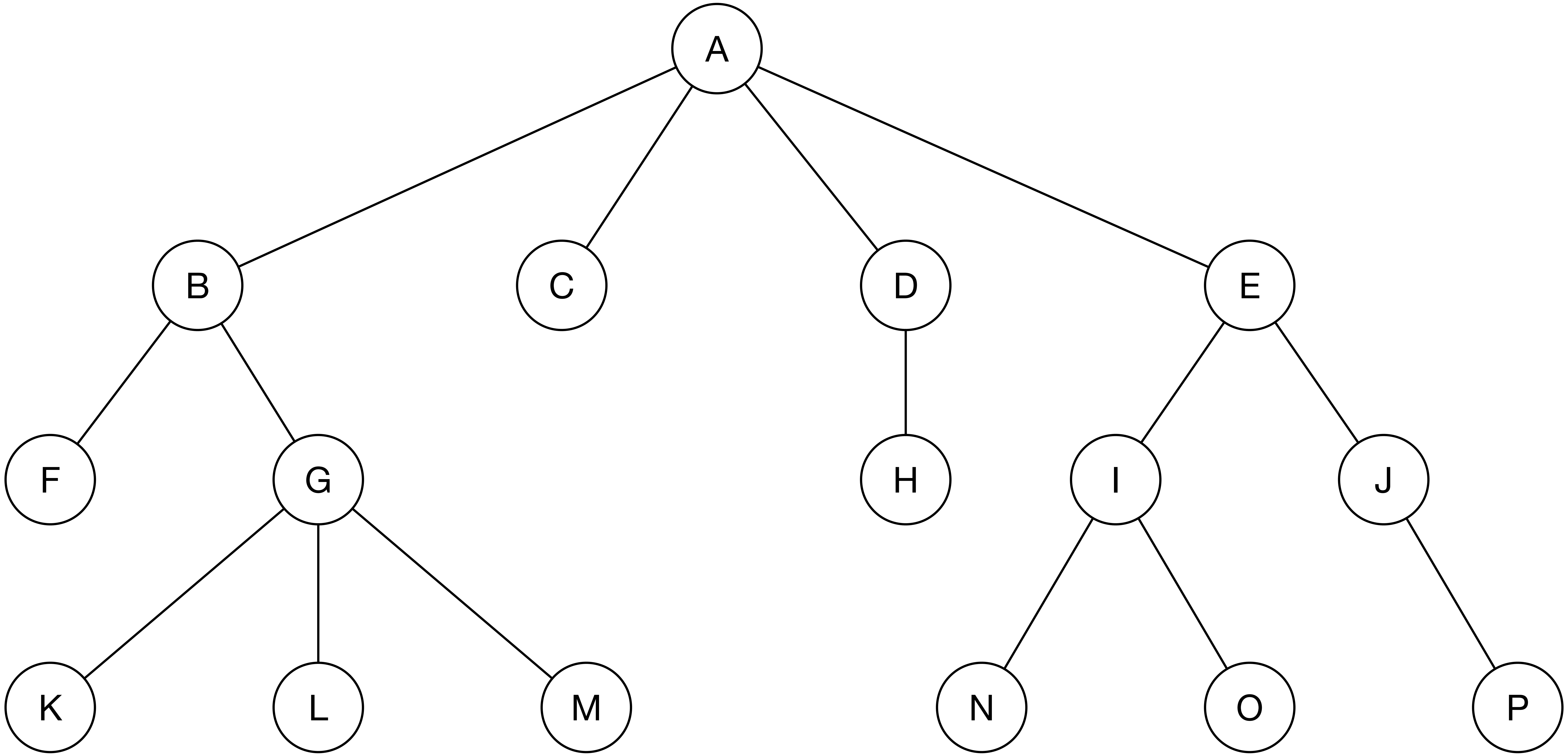
Some terminology

One node has special status. We call that the *root node*.



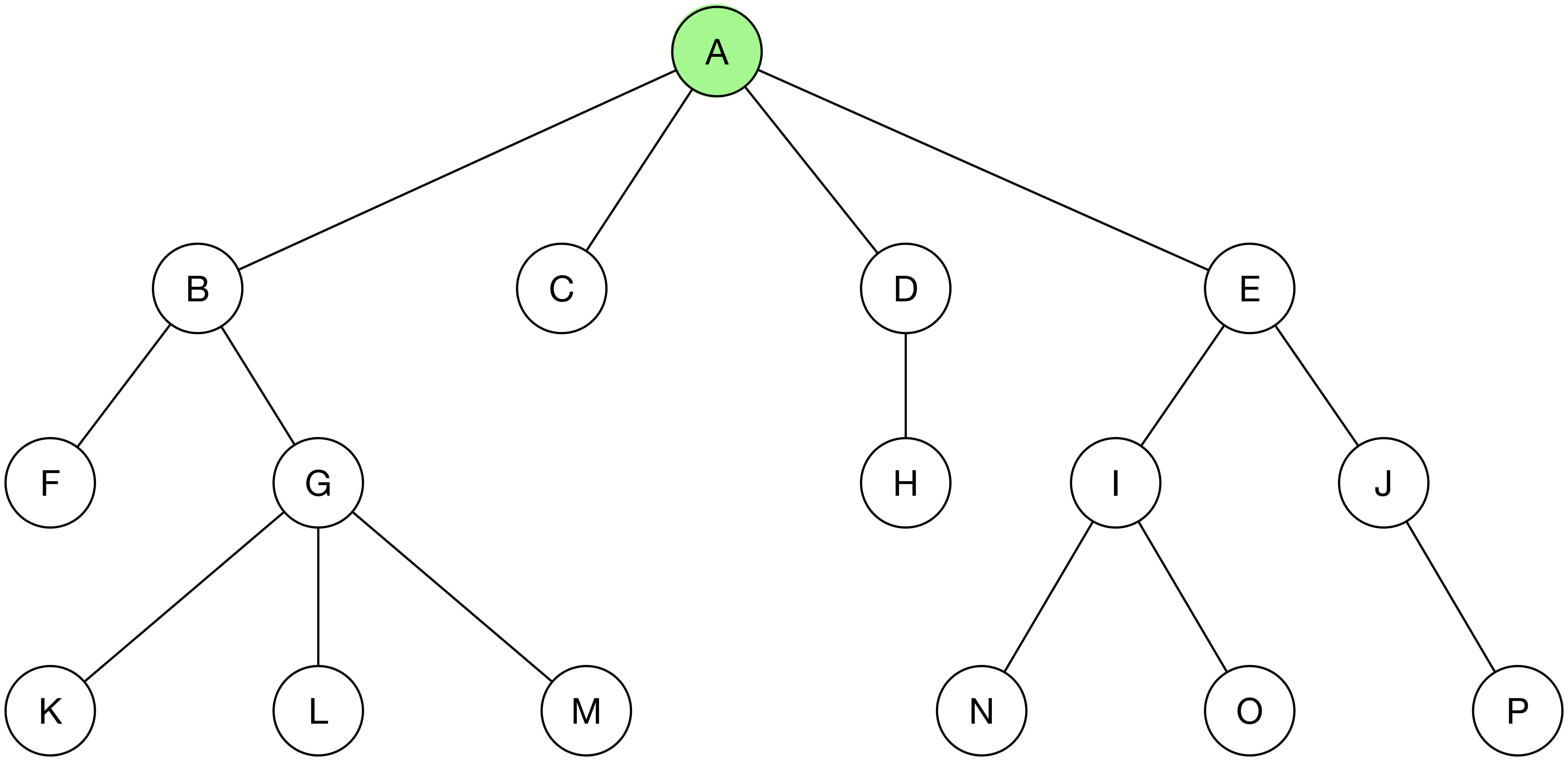
Some terminology

Once a root node is specified we can divide the tree into *levels* based on how far nodes are from the root node.



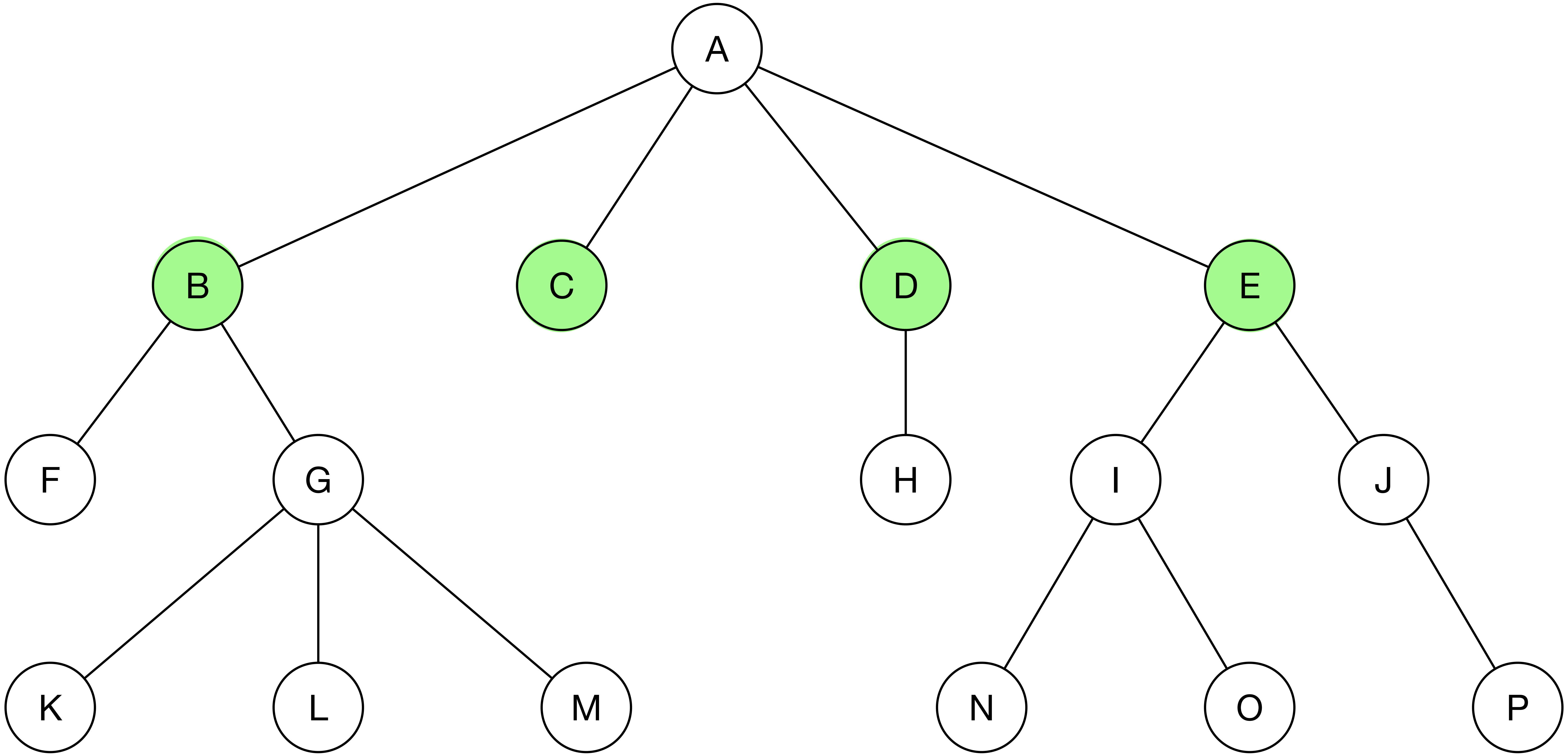
Levels of a tree

The root node is at level 0 (since it is at distance 0 from itself).



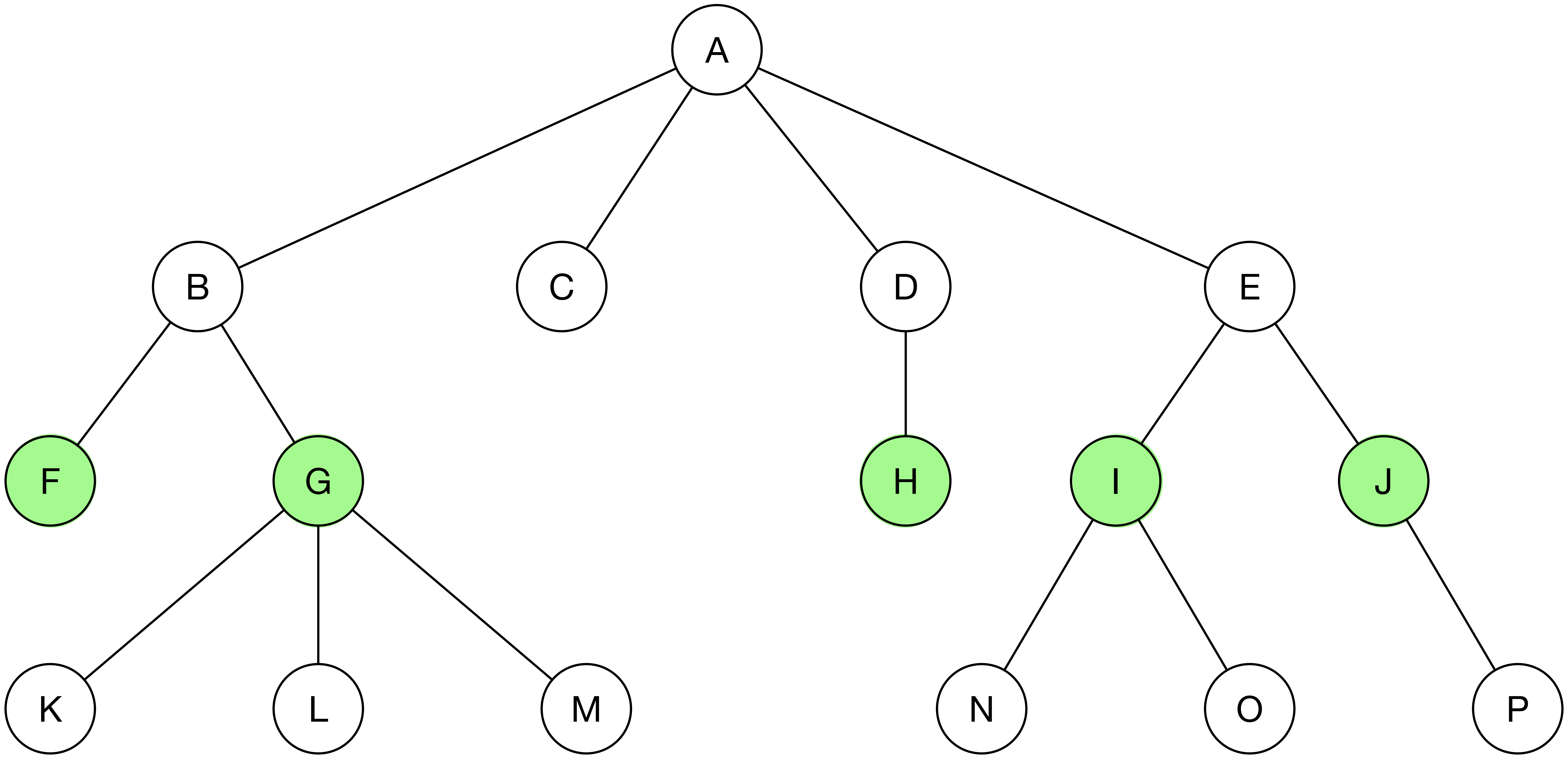
Levels of a tree

B, C, D, and E are at level 1 (being distance 1 from the root)



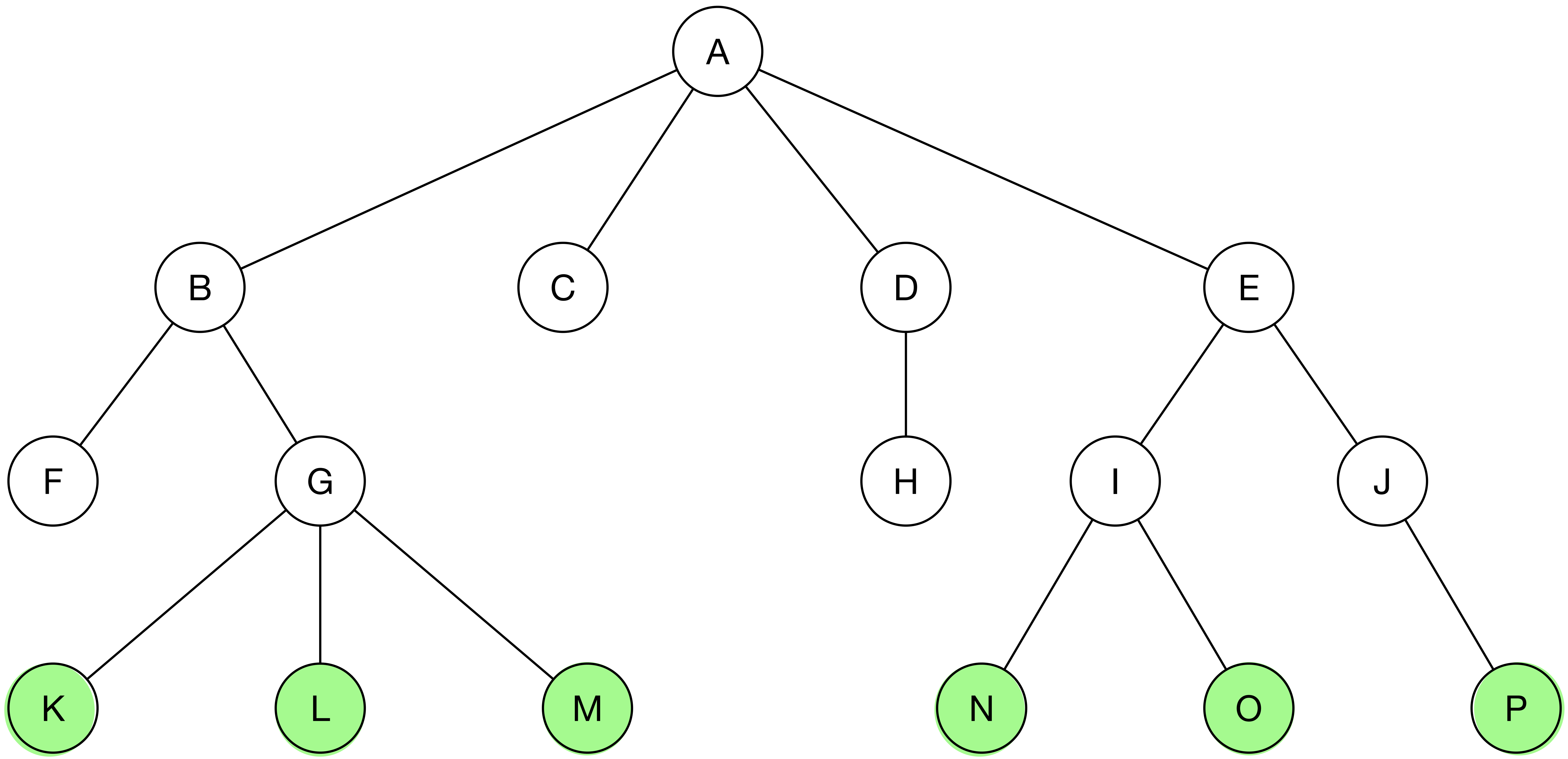
Levels of a tree

F, G, H, I, and J are at level 2 (being distance 2 from the root)



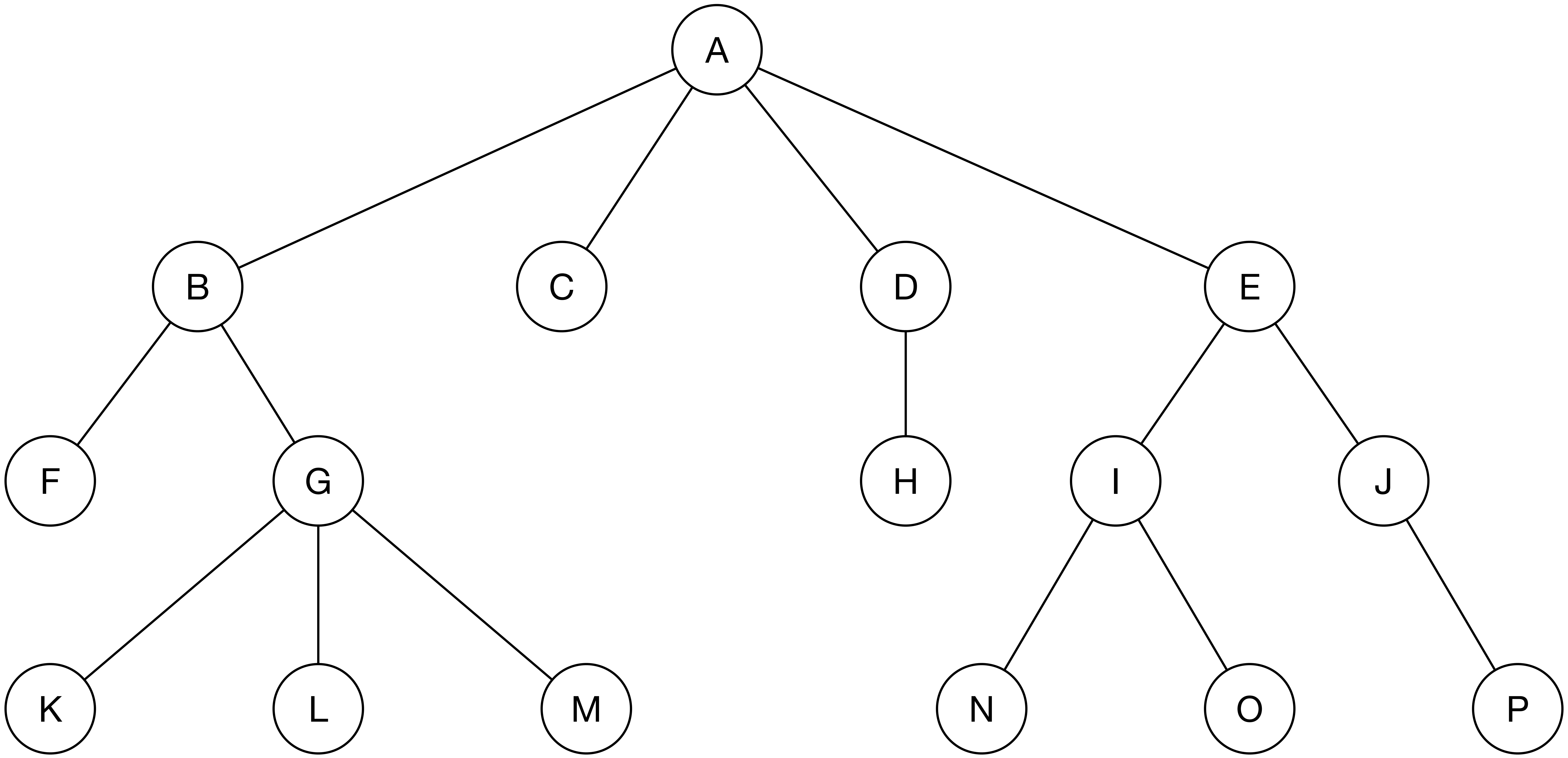
Levels of a tree

K, L, M, N, O, and P are at level 3.



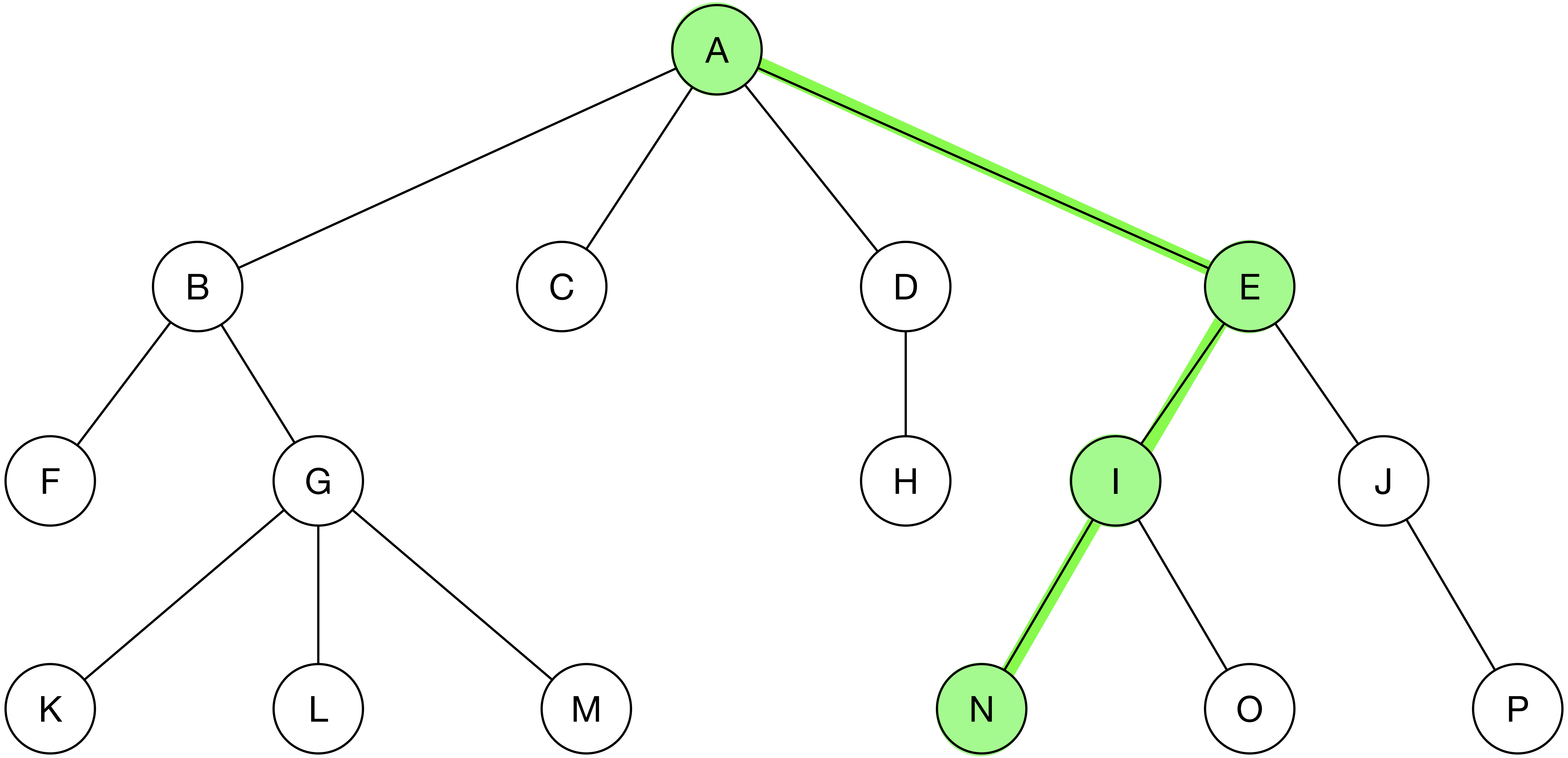
How to draw a tree

Generally we draw a tree with the root at the top and the lowest level at the bottom (though this can become difficult with big trees).



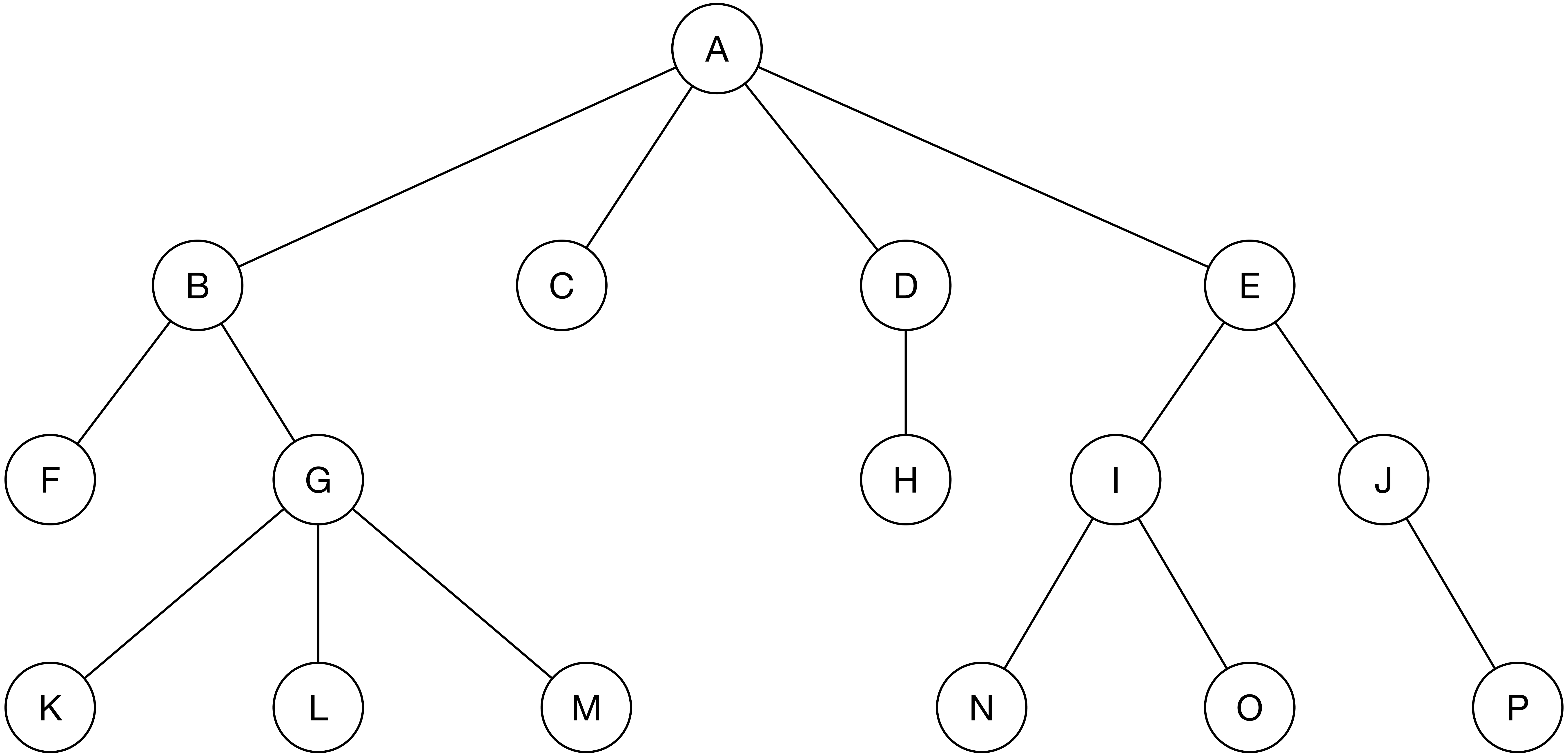
Some properties of trees

The height of a tree, h, is the length of the longest path in the tree from the root to any leaf. In this example, h = 3.



Some properties of trees

A tree with *n* nodes will have *n* - 1 edges. Always.



What's Next

Next we'll learn more about trees, and then how to

- represent a tree in C++
- traverse and search a tree.