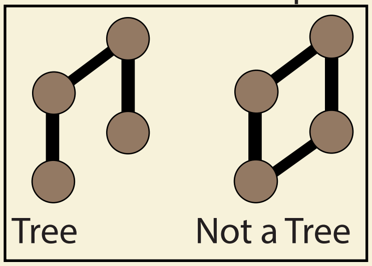
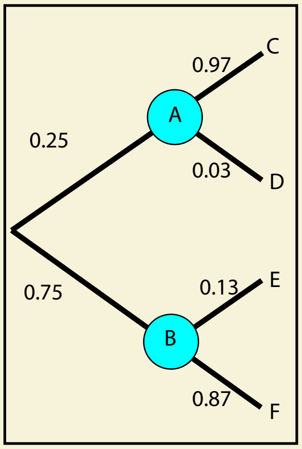
Trees are a type of graph in which there are no cycles. This means that from a certain vertex, there can be no paths that point back to that vertex. Multiple tree graphs that are not connected make up a forest. Connecting a forest with no cycles makes it a tree.



To be considered a tree, the graph must have a unique path between each distinct vertex. Furthermore, if there are at least two vertices on a certain tree, there must be at least two vertices of degree one. As a result, there can only be one path going from each of the vertices. Trees must always have one less edge or path than the number of vertices in the graph. For example, as we saw in the first part of the assignment, a tree with the number of vertices being equal to one, there were zero paths. However, with sixteen vertices, there were fifteen paths which validates the previous statement about how the number of paths is equal to the number of vertices minus one.

In the illustration part of this assignment we were given three tree graphs called arbors, and then we were to create at least three more in the same fashion. We connect a forest of two identical trees at their root vertex, which in turn doubles the number of vertices there are each time we create an arbor. In addition, the number of paths were doubled as well. The creation of an arbor is technically the connection of a forest into a tree. And we can take said "connected forest" or tree and join it with an identical tree to create another arbor. This is the route I took to complete the illustration part of the assignment. When joining the two root vertices, the higher root vertex becomes the new root vertex. Now, let's take A_6 as an example, there are 64 vertices, there must be 63 unique paths. As we can see, when we count the number of paths there is always one path less than the number of vertices in each arbor or tree. In the way we were to make arbors, we could make infinitely many trees, as you keep combining identical trees into one tree, and so on.

Trees can be used for many things, like a visual representation of an if-else statement. The most well known use, is a family tree which maps the generations of a family tree to a certain ancestor, which is the root of the tree. As your parents must be on the tree before you, this makes a family tree a graph with no cycles. When I took AP Statistics in high school we used trees to visualize the probability of something happening. For example, in the picture there is a 25% chance that event 'A' occurs, and when 'A' occurs a 3% chance that event 'D' will happen. If you want to know the probability of 'D' taking place, you multiply down the path to 'D'. So 0.25 * 0.03 will give you .75%. This method of finding the probability of something, displays the data in an easy to read fashion, as well as making use of tree graphs.



In conclusion, tree graphs are very distinct in their characteristics. They must not have vertices with paths back to themselves, so no cycles. Trees must have one path less than the number of vertices (so if thirty-two vertices, thirty-one paths). For the assignment, to make arbors, take two identical trees and connect them at their root vertices. This process can be repeated forever if you desire. Tree graphs have a multitude of real life uses. For example, in probability and statistics to find the probability of a certain event happening. Trees are also used in anthropology in family trees which track current members of a family to the last documented ancestors. Finally, tree graphs are used in computer science in many way; for example, binary trees, which are a data structure.