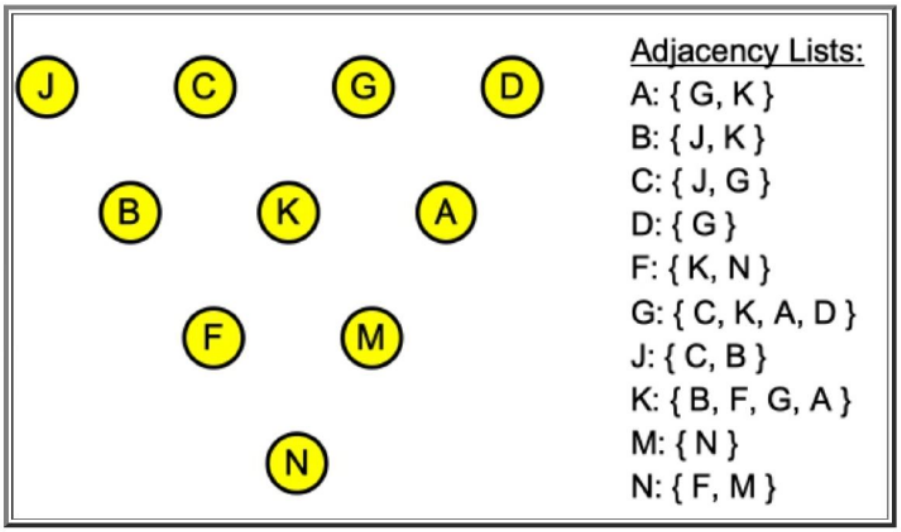
Clicker Question on Doing DFS with a Stack

In the following picture, I give you the adjacency lists of an undirected graph, plus a suggested layout of the nodes.



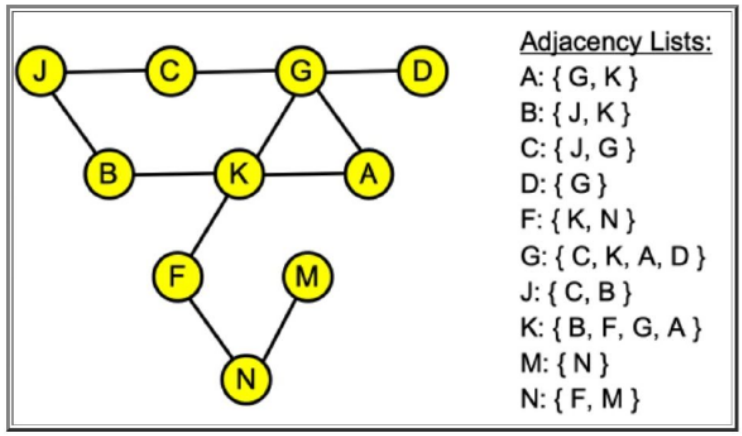
Here are two exam questions that I could see myself asking:

1. Suppose I call DFS(J). Which node is visited first: K or B? (Hint: It's not a bad idea to draw the graph here.)

2. Suppose I print out the name of each node, the first time that it is visited after I call **DFS(J)**. What will be printed (print a 10-character word, with no spaces, like "JBFNCKMGAD")? I suggest that you maintain a stack and a "visited" vector, and then instead of doing recursion, you push nodes from the adjacency list in reverse order. You don't even have to draw the graph for this problem.

Answer to the Clicker Questions

Here's the graph with the edges drawn:



1. Suppose I call DFS(J). Which node is visited first: K or B?

Answer: The simplest thing is to draw in the edges as above. Then, you can see from adjacency lists that J will visit C first. And you can see that the DFS will wind its way from C to B via node K. So the answer is K. See the DFS below if you need to walk through it more slowly.

2. Suppose I print out the name of each node, the first time that it is visited after I call **DFS(J)**. What will be printed (print a 10-character word, with no spaces, like "JBFNCKMGAD")? I suggest that you maintain a stack and a "visited" vector, and then instead of doing recursion, you push nodes from the adjacency list in reverse order. You don't even have to draw the graph for this problem.

Answer: The cleanest way to work through the DFS is to maintain a stack, and we'll push our children on in reverse order. We'll also maintain a vector of visited nodes. You can see the process in the video of class:

*The answer is **JCGKBFNMAD**.*