Consider the graph below.

For the following problems, assume that vertices are selected in alphabetical order (i.e. A before any other vertex from the set of all vertices, D before E from the neighbors of H, etc) from a set when applicable.
Problem 1. Consider the usual iterative breadth-first search algorithm given below.

Algorithm 1: BFS\((G = (V, E))\)

**Input:** A graph \(G\) of vertices \(V\) and edges \(E\)
Let \(Q\) be an empty queue
Pick an undiscovered vertex \(v \in V\)
Put \(v\) into \(Q\)
while \(Q\) is not empty do
    Pull \(v\) from \(Q\)
    if \(v\) is discovered then
        Continue
    Mark \(v\) as discovered
    Visit \(v\)
    foreach neighbor \(u\) of \(v\) do
        Put \(u\) into \(Q\)
return

If the visit operation prints out the name of the vertex visited, what is the output of BFS\((G)\) where \(G\) is the graph given above?

Problem 2. Consider the usual recursive depth-first search algorithm given below.

Algorithm 2: DFS\((G = (V, E))\)

**Input:** A graph \(G\) of vertices \(V\) and edges \(E\)
Pick a vertex \(v \in V\)
DFS\((G, v)\)
return

Algorithm 3: DFS\((G = (V, E), v)\)

**Input:** A graph \(G\) of vertices \(V\) and edges \(E\) with a start vertex \(v \in V\)
if \(v\) is discovered then
    return
Mark \(v\) as discovered
Visit \(v\)
foreach neighbor \(u\) of \(v\) do
    DFS\((G, u)\)
return

If the visit operation prints out the name of the vertex visited, what is the output of DFS\((G, A)\) where \(G\) is the graph given above?

Problem 3. In the previous problem, vertex F did not appear in the output. How would you modify Algorithm 2 and/or Algorithm 3 to guarantee every vertex is visited by a DFS?

Problem 4. In Algorithm 3, suppose the visit operation appeared after the foreach loop instead. What would the output of DFS\((G, A)\) be then when \(G\) is the graph given above?
Problem 5. Given a directed, acyclic graph $G = (V, E)$, a topological sort of $G$ is an ordering of $V$ so that if vertex $v$ is reachable from vertex $u$, then $u$ appears before $v$. For example, in the graph below, the following is a topological sort: A C B D E.

Propose an algorithm to generate a topological sort. (Hint: Craft a useful visit operation for a search)