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?

**Solution 1.** A B C E D J G H I K

**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?

**Solution 2.** A B E C D H I J K G

**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?

2
**Algorithm 4:** FullDFS($G = (V,E)$)

**Input:** A graph $G$ of vertices $V$ and edges $E$

while $V$ contains an undiscovered vertex do

Pick an undiscovered vertex $v \in V$

DFS($G,v$)

return

**Solution 3.** Change Algorithm 2 to Algorithm 4.

**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?

**Solution 4.** H I K J D C G E B A

**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.

```
A
|  |
|--|--
|  |
C

B
|  |
|--|--
|  |
D

E
```

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

**Solution 5.** The visit operation adds the visited vertex to the front of a designated linked list. In addition, the visit operation should occur after a vertex’s children are processed.

**Algorithm 5:** TS($G = (V,E)$)

**Input:** A DAG $G$ of vertices $V$ and edges $E$

**Output:** A topological sort of $G$

// The visit function puts vertices into the front of $L$
// A vertex is visited after its children

Let $L$ be an empty linked list

FullDFS($G$)

return $L$