* Depth-First Search in Graph:
  + Starting at vertex v, DFS will search deeply into the graph before visiting all v’s neighbors.
  + Summary of recursive DFS(G, v)
    1. Mark the starting vertex v as visited, print v.
    2. Assume ni is a neighbor of v, and the order of these neighbors n1, n2, n3 is defined in the adjacency matrix or adjacency list. Neighbors are connected to v with an edge.
       - For each v’s neighbor ni that has not been visited, perform DFS(G, ni).
  + Implementation:
    - Void DFS (Graph G, int v) {
      * G.setMark(v, VISITED);
      * S.O.P(v);
      * For (Edge w = G.first(v); G.isEdge(w); w = G.next(w))
        + If (G.getMark(G.v2(w)) == UNVISITED)

DFS(G, G.v2(w));

* + - * } // End of DFS method
      * NOTE: This implementation is for a directed graph.
* Note for exam: He may ask us to perform a depth-first traversal, given an adjacency list (the linked list implementation). If so, make sure to think about in terms of the adjacency matrix.