This homework covers graphs, graph traversal, and BFS- and DFS-based graph algorithms. It is due in class Monday, March 10.

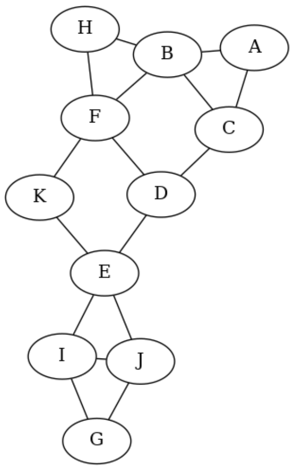
Write your solutions carefully — your work should be neat, readable, and organized.

See the Policies page on the course website for information about revise-and-resubmit, late work, and academic integrity as it applies to homework.

For all problems, use alphabetical order when there is a choice of what vertex to consider next.

1. Find the unweighted shortest path from A to all other vertices for the graph shown below.

   Show your work tracing the unweighted shortest path algorithm — show the contents of the queue at each iteration, indicate the order in which vertices are removed from the queue, and show a table of the `dist` labels as they are updated in each iteration of the `while` loop. (Have a column of the table for each vertex and a row for each iteration.)



2. Is the graph in #1 bipartite? If so, give a coloring; if not, explain why not.

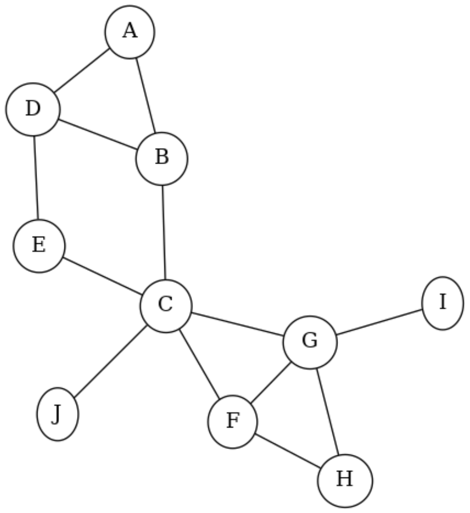
   Show your work tracing breadth-first search (start from vertex A) until either the graph is colored or it is determined not to be bipartite — show the contents of the queue at each iteration, indicate the order in which vertices are removed from the queue, and show a table of the `color` labels as they are assigned in each iteration of the `while` loop. (Have a column of the table for each vertex and a row for each iteration.)

3. Carry out depth-first search on the graph in #1 starting from vertex A.

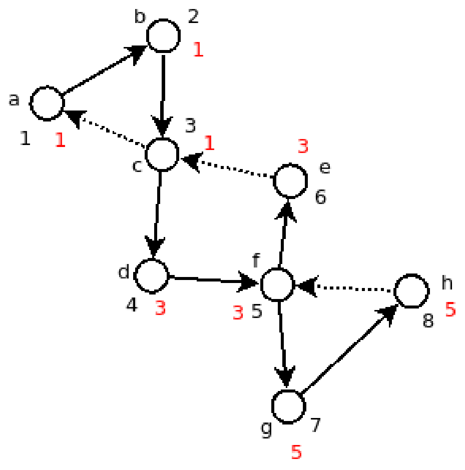
   Indicate the order in which vertices are marked as ”discovered” and as ”processed”. It can be beneficial if you make a mistake to show some additional evidence of your tracing, such as by showing a tree of the recursive calls.

4. Find the cut vertices in the graph shown below.

   Start the DFS from A, and show your work tracing the algorithm by identifying the tree and back edges in the DFS tree and by giving the entry labels and earliest reachable ancestor for each vertex.

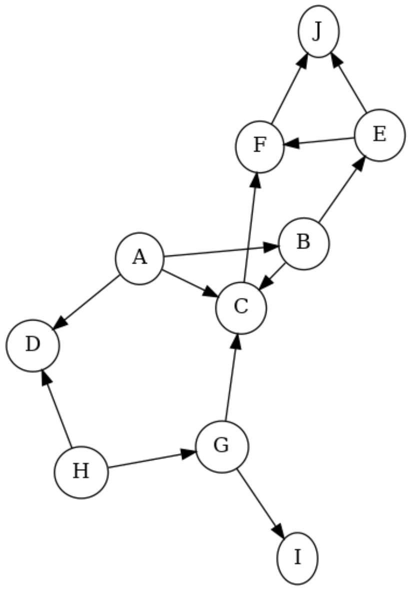


   The example below illustrates a convenient way of showing your work. The entry times are shown in black and the earliest reachable ancestors in red in order to distinguish them. An alternative is to list entry times and earliest reachable ancestors in a table.



5. Perform a topological sort on the graph shown below, giving the resulting order of vertices.

   Show your work tracing the algorithm by giving the exit labels computed for each vertex.


J
F
E
A
B
C
D
G
H
I