In this question we will try out some of the graph algorithms we have seen in class. Draw the annotated graphs that result from applying each of the following algorithms. You do not need to show the running states of the algorithms (i.e. the stacks, queues, etc) just the final result. Whenever you have a choice of visiting two vertices, visit them in alphabetic order (i.e. \textit{a} before \textit{c}, or \textit{u} before \textit{v}). This also means that each of the algorithms should begin at \textit{a}.

(a) (10 pts) \textsc{Test-Bipartiteness}(G_1) to obtain the BFS tree as shown in class. Colour the vertices depending on whether they are at an even or odd level (black and white is fine) and draw the non-tree edges as dashed lines. What is the bipartite partitioning?

(b) (10 pts) \textsc{Top-Sort}(G_2) to obtain the DFS tree. Number the vertices and draw the non-tree edges as dashed lines. What is the topological ordering?

2. (10 pts) Give an O(n + m) time algorithm to decide whether an undirected graph \( G \) contains a cycle of odd length, and, if so, outputs one. Then, argue that your algorithm (i) terminates, (ii) is correct, and (iii) has the claimed running time.
(Hint: What algorithm from class determines whether an undirected graph contains a cycle of odd length? Modify the algorithm to output the cycle.)

3. (10 pts) The reverse of a directed graph \( G = (V, E) \) is another directed graph \( G^R = (V, E^R) \) on the same vertex set, but with all edges reversed; that is, \( E^R = \{(v, u) : (u, v) \in E\} \).

Give a linear-time algorithm for computing the reverse of a graph in adjacency list format. Then, argue that your algorithm (i) terminates, (ii) is correct, and (iii) has the claimed running time.

4. Suppose that you are trapped with nothing but chalk in an enchanted labyrinth. The labyrinth consists of rooms connected by doors. One room contains the exit.

   (a) (1 pt) How can you use the chalk to implement DFS?
   (b) (2 pts) If you could not mark the doors of the labyrinth, could you still implement DFS by only marking rooms?
   (c) (2 pts) If you could not mark the rooms of the labyrinth, could you implement DFS by only marking doors?
   (d) (5 pts) Finally, suppose that doors between rooms can appear and disappear whenever you pass through a door. However, the doors do not change arbitrarily—there is always a path of unexplored rooms from your current position in the labyrinth to the exit (and you can see the mark of a room before passing through a door). Argue that DFS will still reach the exit.
   (e) (BONUS: 5 pts) How long does DFS take in 4(d)?