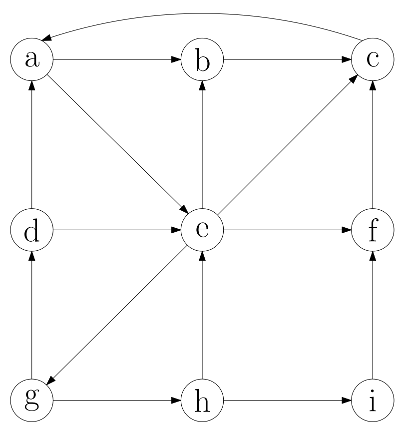
H7.

- The assignment is due at Gradescope on March 16 at 1:35pm. Late assignments will not be accepted. Submit early and often.
- You are permitted to study with friends and discuss the problems; however, *you must write up you own solutions, in your own words*. Do not submit anything you cannot explain. If you do collaborate with any of the other students on any problem, please do list all your collaborators in your submission for each problem.
- Finding solutions to homework problems on the web, or by asking students not enrolled in the class is strictly prohibited.
- We require that all homework submissions are prepared in Latex. If you need to draw any diagrams, however, you may draw them with your hand.

problem 1 *Depth first search*



Consider running depth first search on the above graph starting from vertex a. When there are multiple choices for the neighbor to visit next, assume that the algorithm follows the alphabetical order (from a to z).

(a) Label every edge as either tree, forward, backward, or cross edge.

Solution:

(b) Give the postorder numbers of all vertices.

Solution:

PROBLEM 2 *Mario*

Mario is on a journey to rescue princess Peach. The world consists of n castles $1, 2, \ldots, n$ and m roads connecting them. Let V be the set of castles and E be the set of roads. Castle 1 is Toad Town, Mario's starting location. As we know from the side-scrollers, all roads can only be traversed in one direction and interestingly, all paths eventually lead to Bowser's castle, castle n (castle n can be reached from all other castles). Moreover, there is no directed cycle in the world (the world is a *directed acyclic graph*). At castle i, there are c_i gold coins. Mario needs your help to figure out the way to get to castle n while collecting the maximum number of gold coins. Armed with deep knowledge of dynamic programming, you are eager to help Mario out.

(a) Let $f(i)$ be the maximum number of gold coins Mario can collect if he starts from castle i and goes to castle n. Give a recurrence to compute $f(i)$ based on $f(j)$ $\forall (i, j) \in E$ and the base case $f(n)$.

Solution:

(b) Use an algorithm we learned to compute an ordering to compute all $f(i)$. That is, we want an ordering of $1, 2, \ldots, n$ so that when we want to compute $f(i)$, all the values needed for your recurrence above are already computed.

Solution:

(c) Give pseudocode to implement your algorithm and analyze its running time. You can call algorithms we have learned in the course.

Solution:

PROBLEM 3 *Patrol*

Inspector Gadget wants to patrol all streets in the city. He has decided that the proper way to patrol is to walk on each side of the street exactly once, going up the street in one direction and down in the other direction. Penny is away on vacation and he needs your help to design the patrol route. Assume that the city is modeled as a graph $G = (V, E)$ where the vertices are intersections and the edges are the streets connecting the intersections. Design an algorithm based on depth first search to compute the patrol route. Give its pseudocode and prove its correctness. Your algorithm should run in $O(V + E)$ time.

Solution:

PROBLEM 4 *Pairing up the odd ones*

Consider an undirected graph $G = (V, E)$. The *degree* of a vertex v is the number of edges adjacent to v.

(a) Prove that the sum of the degrees of the vertices is equal to $2 \cdot E$.

Solution:

(b) Prove that there are an even number of vertices with odd degrees. Hint: if the sum of odd numbers is even, what can we say about the number of terms?

Solution:

(c) Consider a vertex v with odd degree. Prove that there is a path connecting v with another vertex u with odd degree.

Solution: