HW#8 due on Sunday April 16 at 11:59pm **ECE 374, Spring 2023**

Instructions

- please check HW1-3 for detailed instructions (they remain the same)

22 (100 PTS.) Bane studies graph theory.

In the Dark Knight Rises, Bane traps the world's dumbest police force in Gotham's underground tunnel system by waiting for every officer to march into the tunnels, and then sealing the exits. The tunnel system consists of the straight tunnels (represented by edges) and the connection points where multiple tunnels intersect and are connected together (represented by vertices).

However, a deleted scene shows Bane's true genius. Bane couldn't just seal the entrances because then if Batman digs through a single entrance, all the cops will get out. He could just blow up every single tunnel but that's too expensive. After all he's not Batman!

Bane realizes he can save some money and selectively destroy tunnels(e)/intersections(v) resulting in two or more disjoint tunnel sub-systems. Therefore, if the entrance to one subsystem is breached, only the cops in that sub-system will get out. We call these tunnels/intersections *critical*.

22.A. (30 PTS.) Design a linear time algorithm to check if a tunnel is *critical*. In other words check if the removal of this tunnel results in two disjoint sub-systems.

22.B. (50 PTS.) Design a linear-time algorithm which identifies every critical tunnel.

22.C. (20 PTS.) Bane theorizes it might be more cost effective to destroy intersection points rather than entire tunnels (this would fragment the system more effectively as well). Develop a linear time algorithm that will identify every critical intersection.

23 (100 PTS.) Poke-vertices: Gotta visit them all

Let G be a directed acyclic graph with a unique source s and unique sink t.

23.A. (20 PTS.) A Hamiltonian path in G is a directed path that contains every vertex in G. Describe an algorithm to determine whether G has a Hamiltonian path.

23.B. (30 PTS.) Suppose the edges of G have weights. Describe an efficient algorithm to find the path from s to t with the maximum **average** weight. Here, for a path π with k edges, and total weight of the edges being w, the average weight is w/k.

23.C. (20 PTS.) Suppose some of the vertices of G are designated as ***special***. A special vertex v has a positive weight $w(v) > 0$ associated with it. A non-special vertex has weight zero. Describe an algorithm, as fast as possible, that computes the maximum weight path from s to t.

23.D. (30 PTS.) Describe an algorithm that returns if the number of different paths from s to t in G is odd or even. To earn any points, your algorithm should not compute the number of such paths (because the number of such paths is potentially exponential, and working with such big numbers is expensive).

24 (100 PTS.) The revolution will not be televised, it will be a question on the homework.

There are n rebels that are currently planning, well, a "party". A rebel v can send messages to only one other rebel, designated as contact(v). Here, if a rebel is given a message, they will send it to their contact, and it would be propagated in this fashion to everyone reachable.

24.A. (20 PTS.) Describe an algorithm, as fast as possible, that computes a rebel v, such that if you send a message to v, the message gets propagated to all the rebels. If there is not such vertex v, the algorithm should output "no solution".

24.B. (20 PTS.) Describe an algorithm, as fast as possible, that computes the minimal number of rebels that needs to be sent directly a message, before the message can be propagated to all the rebels.

24.C. (30 PTS.) A more realistic situation is that a rebel v can contact a group of rebels $R(v)$. Describe an algorithm, as fast as possible, that decides if there is a rebel such that if you send them a message, they can propagate it to everyone. Let $m = \sum_{v=1}^{n} |R(v)|$ be the total size of the input. You can safely assume in analyzing the input that $m \geq n$. Provide a running time in terms of m.

24.D. (30 PTS.) (Hard.) We are back to the setting where each rebel can send only a single message to their contact. Describe an algorithm, as fast as possible, that reassigns the minimal number of contacts, so that a message can be sent to *any* single rebel, and it will be propagated to all rebels. Here, a (single) *reassignment* is assigning a rebel a different contact person than their current one. Prove the correctness of your algorithm. (Here, an algorithm without formal proof of correctness is worth no points.)