CIS 1210 — Data Structures and Algorithms
Homework Assignment 6

Assigned: October 22, 2024 **Due:** October 28, 2024

Note: The homework is due **electronically on Gradescope** on October 28, 2024 by 11:59 pm ET. For late submissions, please refer to the Late Submission Policy on the course webpage. You may submit this assignment up to 2 days late.

A. **Gradescope**: You must select the appropriate pages on Gradescope. Gradescope makes this easy for you: before you submit, it asks you to associate pages with the homework questions. Forgetting to do so will incur a 5% penalty, which cannot be argued against after the fact.

B. **LaTeX**: You must use the LaTeX template provided on the course website, or a 5% penalty will be incurred. Handwritten solutions or solutions not typeset in LaTeX will not be accepted.

C. **Solutions**: Please write concise and clear solutions; you will get only a partial credit for correct solutions that are either unnecessarily long or not clear. Please refer to the Written Homework Guidelines for all the requirements. Piazza will also contain a complete sample solution in a pinned post.

D. **Algorithms**: Whenever you present an algorithm, your answer must include 3 separate sections. Please see Piazza for an example complete solution.

1. A precise description of your algorithm in English. No pseudocode, no code.
2. Proof of correctness of your algorithm
3. Analysis of the running time complexity of your algorithm

E. **Collaboration**: You are allowed to discuss **ideas** for solving homework problems in groups of up to 3 people but *you must write your solutions independently.* Also, you must write on your homework the names of the people with whom you discussed. For more on the collaboration policy, please see the course webpage.

F. **Outside Resources**: Finally, you are not allowed to use *any* material outside of the class notes and the textbook. Any violation of this policy may seriously affect your grade in the class. If you're unsure if something violates our policy, please ask.

1. [20 pts] Lori's Halloween Tricks

It's Halloween, and Lori has played a trick on Arvind and Zora! She's secretly replaced all of the candies they collected with wasabi-filled treats. These $n > 1$ candies are spread out around Meyerson B1, connected by m bidirectional trails between candies. Arvind steps out of the room for something, when Zora realizes what's happened. Determined to stop Arvind from the aura loss of eating a wasabi-filled candy, she decides to travel along the trails to switch the candies back to normal. However, she realizes that no matter which candy she starts from, she can't reach all of the other candies via trails. Zora wants to add trails (without changing any pre-existing ones) to save Arvind from all of the fake candies. Help her design an $O(n + m)$ algorithm that finishes before Arvind gets back to choose a minimal set P of trails to add, specifying where the trails must be arranged relative to the candies.

2. [20 pts] Trickery at the Quad

Every year, Arvind and the other TAs love to go trick-or-treating together in the Quad, where students are prepared with candy. The Quad has $n \geq 2$ dorm rooms, which are connected by m undirected hallways. A hallway connects two dorm rooms, and you can reach any dorm room in the Quad by following some hallway path.

Julia finds out about this and decides to carry out some trickery. She goes to exactly k dorm rooms ahead of time and asks them to place pictures of the students whose names Arvind doesn't remember on their doors and on the hallways that are incident on their rooms (a hallway h is considered incident on a room r if either end of h connects to r). In fear, Arvind will not use these hallways or trick-or-treat at these rooms. However, Julia still wants all remaining rooms to be reachable from the others so Arvind and the other TAs won't be stranded. Design an $O(n + m)$ algorithm to help Julia find the k rooms that she can scare Arvind and the TAs away from. Note that k cannot be treated as a constant.

3. [20 pts] The Maze of Misinformation

On Halloween night, Advit decides to play a spooky trick on Arvind by secretly leading him into a haunted maze with n different eerie doors. Advit has rigged the maze so each door will appear and disappear exactly once and, initially, none of the doors are visible. As Arvind explores the maze, Advit keeps track of the order in which each door appears and disappears in his prank logbook. The next day, when Arvind accuses Advit of manipulating the maze to trap him in terrifying situations, Advit hands over the logbook. However, to keep the prank alive, Advit only gives Arvind limited information about the doors in the maze. For any pair of doors x, y, Advit says one of the following:

(a) "At some point in time, doors x and y were visible at the same time"
(b) "Door x disappeared before door y appeared" (or vice versa)
(c) "I have no information about doors x and y"

However, Arvind suspects that the information that Advit provided was inaccurate. Suppose Advit provided information for each of the $\binom{n}{2}$ pairs of doors. Design an $O(n^2)$ time algorithm to determine whether Advit told the truth. In other words, determine if Advit described a valid door appearance and disappearance sequence. **No proof of correctness is required, but please do justify the runtime of your algorithm.**

For example, the following set of information would be contradictory because you cannot assign relative times to doors a, b, c and d such that all four statements are true:

- Entry 1: Door a disappeared before door b appeared
- Entry 2: Door c disappeared d before door d appeared
- Entry 3: Doors b and c were both visible at one point in time
- Entry 4: Doors a and d were both visible at one point in time

Hint: Consider constructing a graph with $2n$ *vertices.*

4. [20 pts] Arvind's Pumpkin Parade Planning

Arvind loves pumpkins, so the CIS Department has organized a special "Pumpkin Parade" for him this Halloween. The TAs have carved pumpkins and placed each of them at a different professor's office in the CIS Department. Let P be the set of pumpkins, and H be the set of bidirectional hallways where each hallway goes between two offices. The goal is for Arvind to collect every pumpkin without accidentally revisiting any office in a cycle, as that would ruin the parade!

To ensure parade routes remain cycle-free, the TAs have already assigned directions to a subset $H_1 \subseteq H$ of the hallways without any cycles, but the spirit of last Halloween's lost pumpkin stole the map of the Levine hallways, so they couldn't finish. Consider the subset of remaining hallways to be $H_2 = H \setminus H_1$. Design an $O(|P| + |H|)$ time algorithm to help the TAs assign directions to the remaining hallways without creating cycles. You may assume that you have access to $G(P, H_1)$ and $G(P, H_2)$ as separate adjacency lists.

5. [20 pts] Jake's Jack-O'-Lantern Jamboree

Jake is helping organize a halloween party where Arvind will be the guest of honor. With the help of his loyal TA's, he has arduously carved out n Jack-O'-Lanterns in preparation for party. He plans to lay them out across College Green, connecting them using m wires. Unfortunately, Jake could only afford the cheapest type of wires, which transfer power in only one direction. For example, if Jack-O'-Lanterns p and q are connected using a wire from p to q, lighting up p will also light up q, but **not** the other way around. Note that two Jack-O'-Lanterns p and q may be connected using two distinct wires, one directing power from p to q, and one directing power from q to p.

We consider a Jack-O'-Lantern to be a "beacon" if lighting up that one Jack-O'-Lantern will cause all of the n Jack-O'-Lantern to light up as well. In order to make sure that the party runs smoothly and efficiently, Jake wants to find all the "beacon" Jack-O'-Lanterns so he can set up College Green in time for the party and so Arvind will not be disappointed :(. Design an $O(n + m)$ algorithm that takes in the n Jack-O'-Lanterns with the m wires connecting them, and outputs the set of all "beacon" Jack-O'-Lanterns, or states that there are no "beacon" Jack-O'-Lanterns. **No proof of correctness is required, but please do justify the runtime of your algorithm.**