CS3000: Algorithms & Data — Summer 2025 — Laney Strange

Homework 5
Due Thursday June 12th@ 9pm via Gradescope

Name:
Collaborators:

- Put your name on the first page. If you are using the LaTeX template we provided, then you can make sure it appears by filling in the `yourname` command.
- This assignment is due Thursday June 12th@ 9pm via Gradescope. You may submit up to 48 hours late for no penalty, but expect a delay in grading.
- Show ALL your work, even if the problem doesn't specify it.
- You'll have an opportunity to resubmit one homework at the end of the semester.
- Solutions must be typeset, preferably in LaTeX. If you need to draw any diagrams, you may draw them by hand as long as they are embedded in the PDF. I recommend using the source file for this assignment to get started.
- Any solutions that include pseudocode are expected to be in the CLRS style.
- I encourage you to work with your classmates on the homework problems. *If you do collaborate, you must write all solutions by yourself, in your own words.* Do not submit anything you cannot explain. Please list all your collaborators in your solution for each problem by filling in the `yourcollaborators` command.
- If you get stuck on a homework problem, come by office hours or post on Piazza! We recommend you spend about 30 minutes trying to figure out a problem, and then ask for help. We'll be happy to clarify material from class and algorithm concepts, but we will not give out solutions or confirm your answers are correct.
- Finding solutions to homework problems on the web, or by asking students not enrolled in the class, is strictly forbidden.

Problem 1. *(Greedy Alts, 5 points)*

Below are some alternative ways to approach the activity-selection problem using a greedy strategy. For each one, state whether it yields an optimal solution and give a counterexample if it doesn't.

(a) Select the activity of least duration (from those that are compatible with already-selected activities).

Solution:

(b) Select the activity with the latest start time that is compatible with all previously selected activities.

Solution:

(c) Select the compatible remaining activity with the earliest start time.

Solution:

Problem 2. *Depth-First Search (10 points)*

(a) Draw an example of a directed graph G including vertices s, u, and v such that:

- There is a path from u to v in G, and
- There is a DFS traversal starting from s such that u is discovered before v but v is not a descendant of u in the DFS tree.
- (You can assume that ties are broken in alphabetical or numeric order.)

Solution:

(b) Show the $v.s$ and $v.f$ time of every vertex v when we run Depth-First Search on your graph G starting at vertex s.

Solution:

(c) Can your DFS results be used to topologically-sort the vertices of your graph? If so, give the topological order. If not, explain why.

Solution:

(d) Classify each edge of your graph as a tree edge, back edge, forward edge, or cross edge.

Solution:

Problem 3. *Graph Properties (10 points)*

For each question below, justify your answer with an argument (if true) or a counterexample (if false).

(a) TRUE or FALSE? If a directed graph G is strongly connected, then G has a simple cycle that contains all the vertices.

Solution:

(b) TRUE or FALSE? A *forest* is an undirected acyclic graph, which is possibly unconnected. The number of edges in a forest with n vertices and k connected components is $n - k$.

Solution:

(c) TRUE or FALSE? In a connected, undirected graph G, suppose D is the highest d value that results from running BFS on G. It must also be true that any two vertices in G are at distance at most $2D$ from one another.

Solution:

Problem 4. *Minimum Spanning Trees (10 points)*

Prove, or use a counterexample to disprove, the following theorem: Let $G = (V, E)$ be a connected, weighted, undirected graph. Let A be a subset of E that is included in some minimum spanning tree for G, let $(S, V - S)$ be any cut of G such that no edge in A crosses the cut. Finally, let (u, v) be a safe edge for A crossing $(S, V - S)$. Then (u, v) is a light edge for the cut.

Solution:

Problem 5. *Topological Sort (10 points)*

(a) To complete your Bachelor's degree, suppose there are 7 classes you must take. There is some flexibility in the order you take them, but some of the classes have to be taken before others because of pre-requisites. Given the following table of class pre-requisites, determine an order you can take the classes while satisfying the requirements.

Course	Preqrequiste
C	A
G	E, D, F
F	C
D	A
E	B

Solution:

(b) You have to take a specific class, say class "F", before you are allowed to do a co-op. There may be several different courses you could start with to get there. Design an algorithm that will determine which start class will have the fewest prerequisites in its path to "F" and print out the vertex to use as the source. (Here, optimal ordering means fewest possible classes to get to "F" while following the prerequisite constraints in this path.) Your algorithm should take in a graph G, and a destination node v representing the target course. You may:

- Call any algorithm procedures we've covered in class
- Assume that you already know the in-degree and out-degree of every vertex

Solution: