Common Mistake: Using an undirected graph
The graph must be directed because some pours cannot be reversed.
For example, the last step in the sequence above

Note: The induced graph is not a DAG.
Question 2

**Intuition:**
You needed to understand that the graph is a DAG.

**Common Mistake #1:**
Using BFS or DFS to find the shortest paths, then finding the maximum value at a node.
The maximum length shortest path may not be the longest path!
This includes trying to run BFS/DFS from every source node (which is not linear)
It also includes adding a single node as a parent to all the source nodes, then running BFS

**Common Mistake #2:**
Modifying BFS to get the longest path instead of the shortest path, by inheriting the 1+maximum distance of its parents rather than the minimum. This will produce the correct result, eventually, but has an $O(n^2)$ runtime.

**Other Common Mistakes:**
- Returning the length of the longest path, instead of length of the longest path +1
- Removing nodes from the graph one at a time instead of in batches (per semester)
- Basing the order of the nodes strictly on the number of incoming edges
Question 3

Common Mistake:
Determining whether the removal of an edge can disconnect part of the graph from the source relies on a node having only one incoming edge, not only one path from the source.
Question 4

**Common Mistake:** Assuming the graph is a DAG.
This was not specified in the problem. The graph may not have any source or sink nodes. The fact that the graph may have cycles means you should not have used Topological Sort.
Question 5

**Intuition:**
The problem asked for an $O( (|V| + |E|) \log |V| )$ runtime. This is a hint.

**Common Mistake:**
Using BFS/DFS to simply find the maximum length edge on any path from s to t.
Notes & Advice for the Test

The test is open-book and open-note.

Some people have asked about whether you can use your homework during the test - **you can.**

Professor Anshelevich selects the homework, recitation, and lab problems so that they cover all of the concepts he wants you to understand. You need to understand all of those problems.

There are 5-6 questions on the test that will be similar to the types of questions on the homework.

You should understand the Black Box for every algorithm (input, output, runtime).
Example: If a question asks for an algorithm that runs in linear time, you should be able to immediately rule out using Dijkstra’s, Prim’s, Kruskal’s, etc.

If a graph problem involves events happening in sequence: topological sort should come to mind

You don’t have to understand how Kosaraju’s algorithm works, but you do need to be able to use it as a black box (input, output, and runtime).

A tree is a connected, undirected, acyclic graph with a single source node where every child node has exactly one parent.

If an algorithm requires modifying the graph (adding/removing edges), you should pay close attention to whether these modifications occur before, during, or after you traverse/search the graph.
Examples: In Homework 1, Problem 4 required that you run BFS first and then build edges, while Problem 6 required that you remove an edge before you run BFS. In Homework 2, Problem 2 is most easily solved by removing source nodes iteratively during the traversal of the graph.

The answer to a problem is pretty much never going to be “run BFS from every node” or “run ____ from every node in the graph.” Even if something like this can return the correct result, there is virtually always a more efficient way to do it for any problem you will see in this course. For this first exam, quadratic runtimes are evil.

Be sure to read the questions on homework and the test carefully!
I would say about 5-10% of lost points on homework are from not reading the question carefully. You don’t get much credit for finding the right answer to the wrong problem.
Topics Covered on Test

- Basic graphs, components
- BFS and DFS
- Directed graphs
- DAGs
- Topological Sort
- Strongly Connected Components
- Dijkstra’s Algorithm
- Trees
- Minimum Spanning Trees

The focus will be mostly on the topics covered in Homework #1-2

Topics NOT Covered

- Greedy algorithms
- Divide & Conquer
- MergeSort

You will NOT be asked to prove anything