Homework 9

Due by 4:30pm on November 11, 2020

Viewings and Readings

[Links to the slides, recordings, and quiz are on the course web site.]
Review the week 9 lecture slides.
View the week 9 lecture recordings and the discussion session recording.
Complete the week 9 quiz.
Read sections 6.5-6, chapter 7, and sections 8.1-6 of the textbook (except the more challenging "heart" parts; section 8.7 is optional).

Problems

Please write precise and concise answers. Except where explicitly indicated, your algorithm descriptions should use either clear, concise, and precise plain English or clear, concise, and precise pseudo-code that uses a style similar to the pseudo-code in your textbook. Submit your solutions to problems **1(a), 2(a), and 3(a)** via D2L as a Word or PDF file or as scans/photos of legible handwritten notes. Submit your solutions to problems **1(b), 2(b)** and **3(b)** via Kattis.

1. Week 9 problem *builddeps* on Kattis.

(a) Explain how DFS can be used to solve this problem. Make sure you describe what needs to be done to print the files in the right order.

```
If we organize the graoh so every dependency has a directed edge to what depends on it, we then just need to find the topological order of the graph
This will ensure every vertex is printed before any vertex it has a directed edge to.
We can get topological by tracking the postorder of a dfs from a vertex and reversing that postorder
```

(b) Use your insight from **(a)** to implement a solution using your preferred language and submit your implementation via Kattis.

2. Week 9 problem *reversingroads* on Kattis.

(a) Explain how Kosaraju-Shamir's strongly connected components algorithm can be used to solve this problem. Make sure you also describe how to find the edge to reverse, if it is necessary and if it exists, that makes the case valid.

```
We can use Kosaraju-Shamir's algorithm to count the number of strongly connected components.  If theres only one that means the problem is solved.
If there's more than one, there's only one way to make the graph valid in one edge reversal.
There has to be exactly one sink and one source.  If there's more than one of either then it's impossible to reverse one edge to get rid of multiple
If there is one sink and one source, we then need there to be an edge between the sink and source.  If we want to get rid of the sink and source, the
If the above checks are passed then the problem can be solved by reversing the road from the source to the sink.

Theres an edge case where we also need to check the the sink has multiple roads connected to it.
If there are only two vertexes and one road, that's one sink, one source, and a road connecting them but the reverse will do nothing but switch the s
```

(b) [One of 2(b) and 3(b) is optional] Use your insights from **(a)** to implement a solution using your preferred language and submit your implementation via Kattis.

3. Week 9 problem *kitchen* on Kattis.

(a) Explain how Dijkstra's algorithm can be used to solve this problem. Since no graph is described explicitly in the problem, describe the graph that Dijkstra's algorithm is applied to and how its nodes and edges are generated.

```
We can define the graph as vertexes being the current state of the cups and the edges being the amount of liquid poured to get to the next state.
Since one attribute of Dijkstra's algorithm is that the current node off the priority queue is the shorest distance, we don't need to build the whole
So we dynamically build the graph starting the the first cup being full then adding vertexes for each state we can achieve by pouring the liquid to t
We keep going through Dijkstra's algorithm, marking the min distance to each state until we reach the state which contains our target for the first c
```

(b) [One of 2(b) and 3(b) is optional] Use your insights from **(a)** to implement a solution using your preferred language and submit your implementation via Kattis.