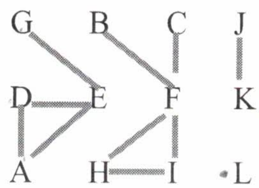
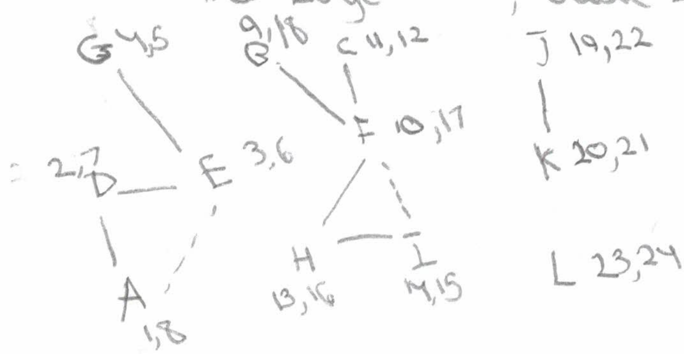
Zachary Neeley

CS372 Assignment #4.

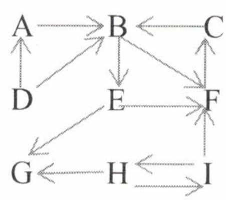
1. Perform a depth-first search on the following graph; whenever there's a choice of vertices, pick the one that is alphabetically first. Classify each edge as a tree edge or back edge, and give the `pre` and `post` number of each vertex. Tree Edge ——, Back Edge - - -

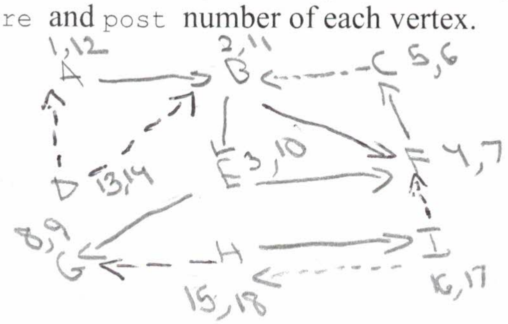




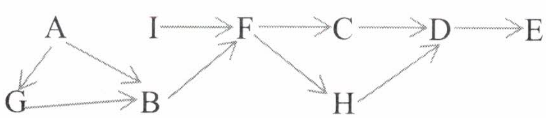
2. Perform depth-first search on the following graph; whenever there's a choice of vertices, pick the one that is alphabetically first. Classify each edge as a tree edge, forward edge, back edge, or cross edge, and give the `pre` and `post` number of each vertex.

Tree Edge, ——>
Forward Edge, ——>
Back Edge, - - - ->
Cross Edge, - - ->





3. Run the DFS-based topological ordering algorithm on the following graph. Whenever you have a choice of vertices to explore, always pick the one that is alphabetically first.
(a) Indicate the `pre` and `post` numbers of the nodes.
(b) What are the sources and sinks of the graph?
(c) What topological ordering is found by the algorithm?
(d) How many topological orderings does this graph have? List all of them.



On Next Page

4. Directed graph G is represented by the adjacency matrix below. Draw the directed graph G and run the strongly connected components algorithm on it (use the algorithm from p.94 of the textbook). When doing DFS on G^R: whenever there is a choice of vertices to explore, always pick the one that is alphabetically first. Answer the following questions.
(a) In what order are the strongly connected components (SCCs) found?
(b) Which are source SCCs and which are sink SCCs?
(c) Draw the "metagraph" (each meta-node is an SCC of G).

5)

a) SCCs: {A, D, G, E}, {F}, {C, B, I}, {H}

b) Sink: {A, D, G, E}
Source: {F}

c)

$$\begin{tikzcd}
\{F\} \arrow[r] \arrow[dr] & \{C, B, I\} \arrow[r] & \{H\} \arrow[dl, bend left] \\
& \{A, D, G, E\} &
\end{tikzcd}$$

d) The minimum numbers of edges we would need to add would be one edge.

(d) What is the minimum number of edges you must add to this graph to make it strongly connected?

	A	B	C	D	E	F	G	H	I
A	0	0	0	0	1	0	1	0	0
B	0	0	1	0	0	0	0	0	0
C	0	1	0	0	0	1	0	0	1
D	1	0	0	0	0	1	1	0	0
E	0	0	0	1	0	0	0	1	0
F	0	0	0	0	0	0	0	0	0
G	0	0	0	1	0	1	0	0	0
H	0	1	0	0	0	0	0	0	0
I	0	1	0	0	0	0	0	0	0

(Recall that 1 in row i and column j means that there is an edge from vertex i to vertex j. For instance, there is an edge from A to E.)

5. Exercise 3.15. Note: "Formulate this problem graph-theoretically" means that you need to explain what the vertices in the graph are and what the edges are (when exactly two vertices are connected by an edge).

On Next Page

6. Exercise 3.23. Note: You need to explain the algorithm and analyze its running time.

On Next Page

What to submit:

- Submit answers to all the questions on Canvas (either type or scan written answers).

5)

a) When looking at a map with roads, intersection you can see that a crossroad would be vertices and the roads will be the edges because they are directional or one-way. This would show that this would be the same as a graph because it would be a strongly connected system. It is important to note that this analogy would only work if the graph only has one SCC.

b) The claim that the mayor claimed says that starting with the town hall you cannot get from one SSC to another in the graph. This is the same as saying that the SCC that the mayor and town hall is the sink for the graph. This can be done by finding the components then running the DFS algorithm starting with the vertex that the mayor and town hall are in.

6) Calling a vertex where all vertices can be reached, a vista vertex. When a graph only has one vista vertex which would show that it would only have one SCC. This is because there are two SCCs that are not reachable from each other, which would show that one of the SCCs would have to be the vista vertex. One thing to note is that every vertex in the source SCC will always be a vista vertex. This algorithm would have to be a simple DFS starting from any node making sure to note the vertex with the highest post value. This vertex would become the source SCC. After this run DFS so that we can see if all the nodes are reachable. Because of this, the algorithm uses decomposition of SCCs and DFS which would show that the running time is linear.