*Note*: Your TA may not get to all the problems. This is totally fine, the discussion worksheets are not designed to be finished in an hour. The discussion worksheet is also a resource you can use to practice, reinforce, and build upon concepts discussed in lecture, readings, and the homework.

## 1 Short Answer

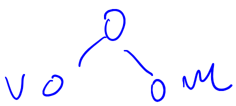
For each of the following, either prove the statement is true or give a counterexample to show it is false.

(a) If $(u, v)$ is an edge in an undirected graph and during DFS, post$(v)$ < post$(u)$, then $u$ is an ancestor of $v$ in the DFS tree.

(a) True

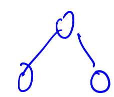
(b) In a directed graph, if there is a path from $u$ to $v$ and pre$(u)$ < pre$(v)$ then $u$ is an ancestor of $v$ in the DFS tree.

(c) In any connected undirected graph $G$ there is a vertex whose removal leaves $G$ connected.




(a) False

(c)



~~False~~

True

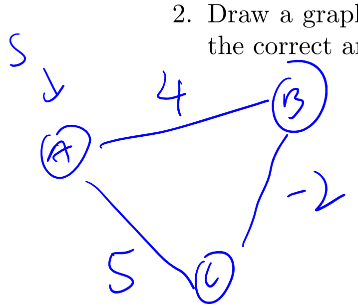
leaf.

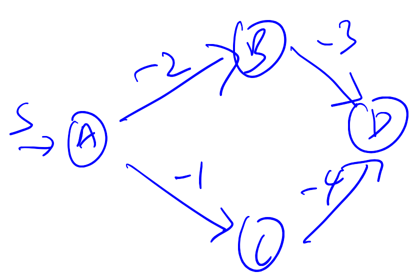
(b) True


## 2 Dijkstra's Algorithm Fails on Negative Edges

Draw a graph with five vertices or fewer, and indicate the source where Dijkstra's algorithm will be started from.

1. Draw a graph with no negative cycles for which Dijkstra's algorithm produces the wrong answer.
2. Draw a graph with at least two negative weight edge for which Dijkstra's algorithm produces the correct answer.

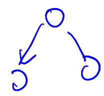


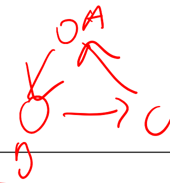




## 3 More Short Answer Questions

(a) Let $G$ be a dag and suppose the vertices $v_1, \ldots, v_n$ are in topologically sorted order. If there is a path from $v_i$ to $v_j$ in $G$, are we guaranteed that $i < j$?

True





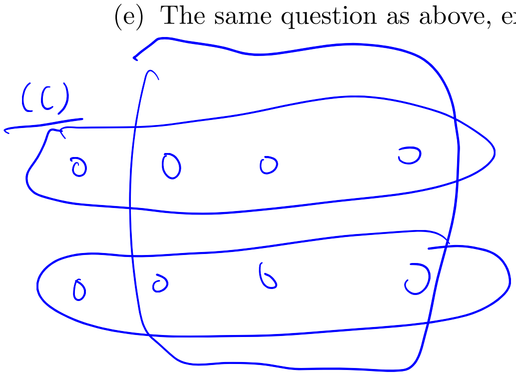

(b) Let $G = (V, E)$ be any strongly connected directed graph. Is it always possible to remove one vertex (as well as edges to and from it) from the graph so that the remaining directed graph is strongly connected? ~~True~~ NO

(c) Give an example where the greedy set cover algorithm does not produce an optimal solution.

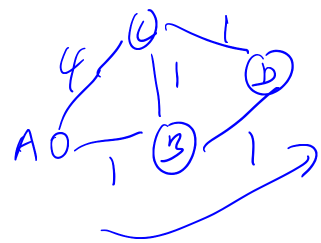
(d) Let $G$ be a connected undirected graph with positive lengths on all the edges. Let $s$ be a fixed vertex. Let $d(s, v)$ denote the distance from vertex $s$ to vertex $v$, i.e., the length of the shortest path from $s$ to $v$. If we choose the vertex $v$ that makes $d(s, v)$ as small as possible, subject to the requirement that $v \neq s$, then does every edge on the path from $s$ to $v$ have to be part of every minimum spanning tree of $G$?

(e) The same question as above, except now no two edges can have the same length.

(c)



(d) the same length



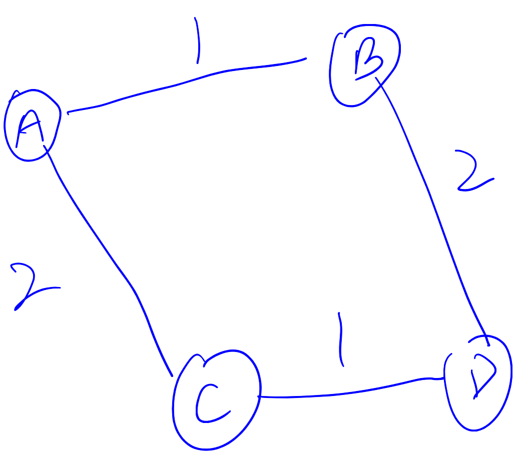

NO

(e) Yes?

## 4 Unique Shortest Path

Shortest paths are not always unique: sometimes there are two or more different paths with the minimum possible length. Show how to solve the following problem in $O((|V| + |E|) \log |V|)$ time.
*Input:* An undirected graph $G = (V, E)$; edge lengths $l_e > 0$; starting vertex $s \in V$.
*Output:* A Boolean array usp[·]: for each node $u$, the entry usp[$u$] should be true if and only if there is a *unique* shortest path from $s$ to $u$. (Note: usp[$s$] = true.)
[Provide 3 part solution.]

→ collected vertex




add a new vertex
u → v is the current shortest.
if usp[u] is True the usp[v] is true
else: for neighbor u of v:
if u in collected vertex,
dist[u] + edge(u,v) = dist[v]
usp[s] = true
break