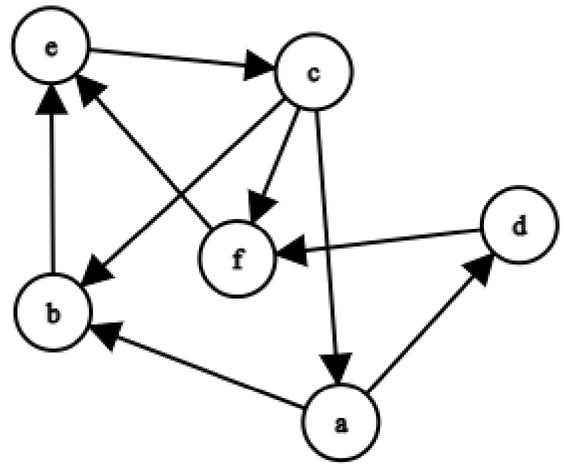
RUNI – Algorithms – Homework 3

NOTE: the general homework guidelines published in homework 1 are valid for all homework assignments in this course.

Due date: Sunday 04/12/2022



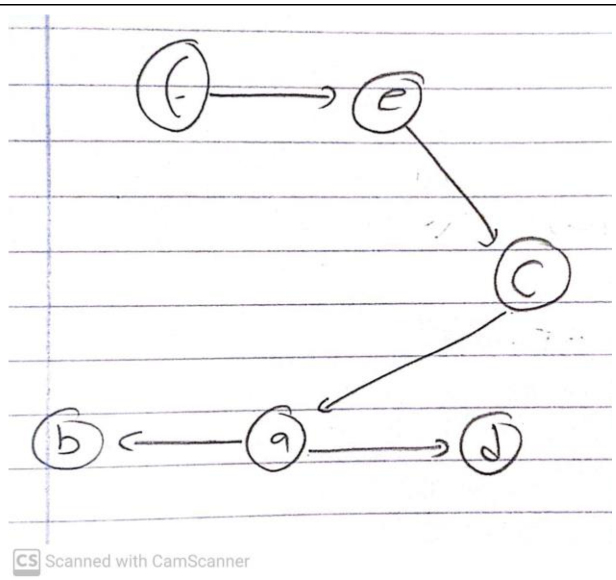
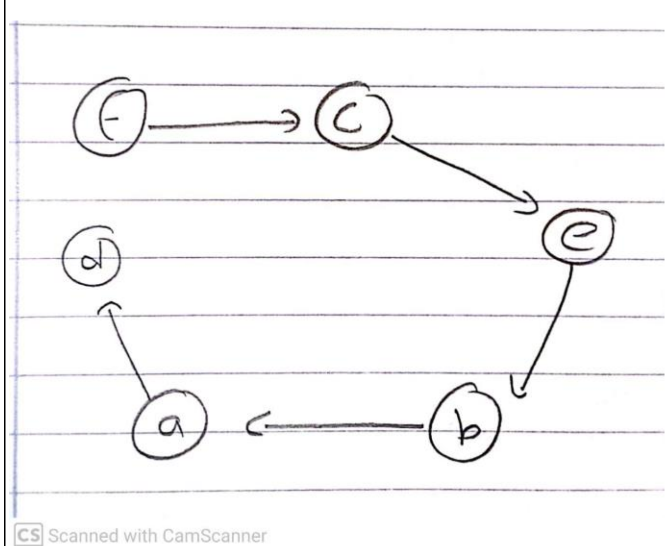
Problem 1 (20 pts)

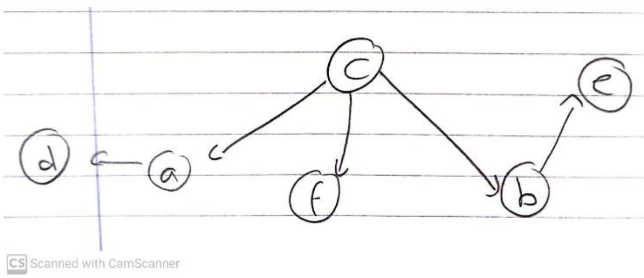
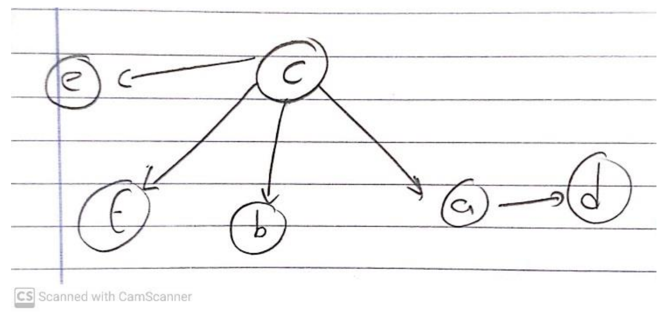
Let G be the graph illustrated in the figure.

Let G' be the underlying undirected graph of G.

In the following questions, assume vertices in each adjacency list appear in alphabetical order, for example, b's list in G' is a→c→e.

Pay attention to whether you're asked to run **D**FS or **B**FS, on G or on G', and from which vertex. Whenever tie-breaking is required (for example, to select a root of the next tree in a forest) give priority to vertices in alphabetical order.

1. Run DFS(f) on G and draw the resulting directed spanning forest of G.	2. Run DFS(f) on G' and draw the resulting directed spanning tree of G'.
	

3. Run BFS(c) on G and draw the resulting directed spanning forest of G.	4. Run BFS(c) on G' and draw the resulting directed spanning tree of G'.
	

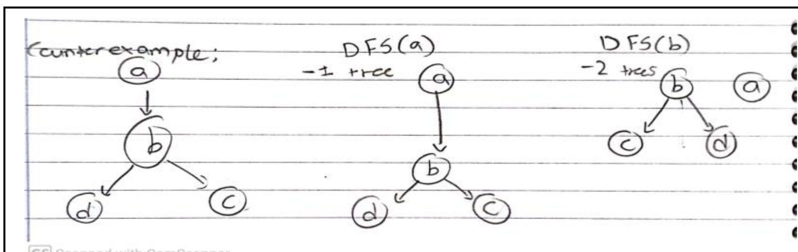
<u>Problem 2 (35 pts):</u>

<u>For each of the following statements, decide whether it is true or false and prove or refute accordingly.</u>

1. Let G=(V,E) be an **undirected** graph, let a,b $\in$V. The number of trees in the forests returned by DFS(a) and DFS(b) is the same.

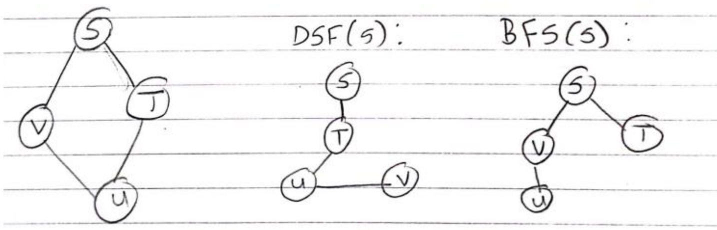
True. Since G is an undirected graph, DFS can traverse the edges without restrictions. In this case, the number of trees in the forest returned by DFS is the number of connected components of G. Regardless of if DFS(a) or DFS(b) is called, the number of connected components in G remains the same, so it will return slightly differing trees, but both DFS calls will result in a forest with n trees (where n is the number of connected components)

2. Let G=(V,E) be a **directed** graph, let a,b $\in$V. The numbers of trees in the forests returned by a DFS(a) and DFS(b) is the same is the same.



3. Let G be an undirected graph. Assume you run BFS in G, starting from s. During the run, the vertex v marks the vertex u. In the run of $DFS(s)$ on G, that uses the same adj. lists as the run of BFS, v will be marked before u.

False, Counterexample:



4. Let G be an undirected graph that has a Hamiltonian cycle. For every vertex v∈V, there exists a run of DFS(v) in which the resulting directed spanning tree has a single leaf.

True. Since G has a Hamiltonian cycle, there exists a cycle that goes through every single vertex once. Calling DFS(v) for any vertex will causes DFS to traverse through the Hamiltonian cycle from v to the last vertex in the Hamiltonian cycle. This results in the tree returned to be a single path, which always has one leaf.

5. Let G=(V,E) be a simple undirected graph with |V|=4. Some run of DFS produces a spanning tree in which the root has degree 2. Claim: |E|≤4.

True. The spanning tree returned is a root with 2 children – this subtree already has 2 edges. For the spanning tree to be complete it needs to have all the 4 vertices. Therefore there is 2 options for the 4th vertex to connect to the tree – it can connect to either one of the root's children. This will result in only 1 more edge being added (connecting the 4th vertex to one of the children) and no more. If there was 2 edges connecting the 4th vertex to the tree, this would results in a cycle with all the 4 nodes – which is a contradiction to the graph being a spanning tree.

Problem 3 (20 pts)

Prove or disprove each of the following claims. Consider each claim independently (that is, assumptions from one claim don't apply to the other):

1. Let G=(V,E) be a connected undirected graph such that $|V|>2$. Let T be a directed spanning tree of G that has been computed using DFS. If u and v are leaves in T (meaning they have no children), then $(u,v) \notin E$.

True. ATC that $(u,v) \in E$. Then during any run of DFS, when u is active, it will mark v, meaning that in the resulting tree there will be an edge from u to v. This implies that u is not a leaf. Therefore contradiction. Same idea for if v is active (it would mark u).

2. Given a connected undirected graph, an edge e is called a *bridge* if removing e increases the number of connected components in the graph. Let G=(V,E) be a graph with $|V| \geq 3$, that has no bridges. For every vertex $u \in V$, if BFS(u) is performed, then in the resulting undirected spanning tree, $deg(u)>1$.

True. ATC that deg(u) = 1 (it cannot equal 0 as G is a connected graph). This implies that there is 1 edge connecting u to the graph G, as if there was more then after the run of BFS, deg(u) would've been calculated to be more than 1. This 1 edge can therefore be considered a bridge, as removing it would make u isolated and not connected to G, which would make u its own connected component. This would then increase the number of connected components in G. Therefore contradiction to G having no bridges.

Problem 4 (25 pts):

Let G=(V,E) be an unweighted simple directed graph. Some of the edges are colored red. Let E'⊆E denote the set of red edges. Given a vertex s∈V, suggest an efficient algorithm for finding the length of a shortest path from s to every other vertex in the graph, fulfilling the following condition: **the path includes at most two red edges**. In other words, every v∈V should be labeled with the length of a shortest path from s to v in which there are at most two edges from E' and any number of edges from E\E'.

Describe the algorithm, prove its correctness, and determine and justify its running time complexity.

I don't know