Collaboration Policy: You are encouraged to collaborate with up to 4 other students, but all work submitted must be your own *independently* written solution. List the computing ids of all of your collaborators in the `collabs` command at the top of the tex file. Do not share written notes, documents (including Google docs, Overleaf docs, discussion notes, PDFs), or code. Do not seek published or online solutions for any assignments. If you use any published or online resources (which may not include solutions) when completing this assignment, be sure to cite them. Do not submit a solution that you are unable to explain orally to a member of the course staff. Any solutions that share similar text/code will be considered in breach of this policy. Please refer to the syllabus for a complete description of the collaboration policy.

Collaborators: collaborators

Sources: sources

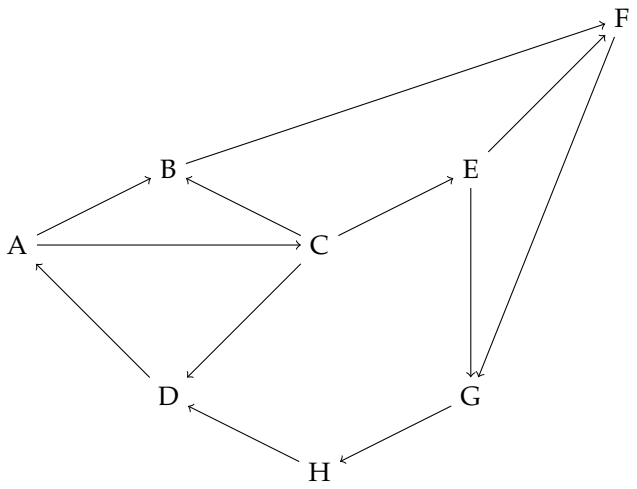


Figure 1: Graph used for Problems 1 and 2

PROBLEM 1 *Using Depth First Search*

Apply depth first search starting at Node *A* on the graph shown in Figure 1. Use Table 1 to record the seen and done timestamps (starting with 0). Note: When multiple nodes could be chosen at a given step, pick the one that comes first alphabetically (i.e. If either node F or node H could follow node C, choose node F because F is alphabetically before H.

Node	Seen (Discovery Time)	Done (Finish Time)
A	?	?
B	?	?
C	?	?
D	?	?
E	?	?
F	?	?
G	?	?
H	?	?

Table 1: DFS Seen and Done Timestamps

PROBLEM 2 *Classifying Edges*

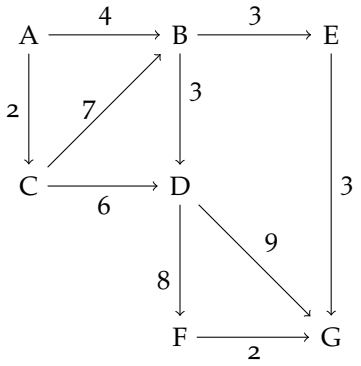
Refer to the graph shown in Figure 1. Use the seen and done times that you calculated above to classify the edges. For each edge choose between: Tree Edge, Back Edge, Forward Edge, Cross Edge. Use Table 2 to record your answers.

Edge	**Edge Type**
(A → B)	? Edge
(C → B)	? Edge
(C → D)	? Edge
(D → A)	? Edge
(A → C)	? Edge
(C → E)	? Edge
(E → F)	? Edge
(F → G)	? Edge
(G → H)	? Edge
(H → D)	? Edge
(B → F)	? Edge
(E → G)	? Edge

Table 2: DFS Edge Types

PROBLEM 3 *Using Dijkstra's Algorith*

Given the following graph:



Apply Dijkstra's Algorithm starting at node A. Each time a decreaseKey call is made, record that in Table 3 below. Record the vertex and the value assigned. For some vertices, decreaseKey will only be called once; others might have more than one call. If decreaseKey is called more than once for a given node, there should be multiple entries in the table. There are 15 rows provided in the table, but you may not need them all, just leave the extra rows at the end of the table as they are if not needed. Note: When multiple nodes could be chosen at a given step, pick the one that comes first alphabetically (i.e. If either node F or node H could be chosen next because they have equally minimum values in the priority queue, choose node F because F is alphabetically before H).

decreaseKey is called	**Vertex**	**Value**
1	A	0
2	?	?
3	?	?
4	?	?
5	?	?
6	?	?
7	?	?
8	?	?
9	?	?
10	?	?
11	?	?
12	?	?
13	?	?
14	?	?
15	?	?

Table 3: DecreaseKey Calls