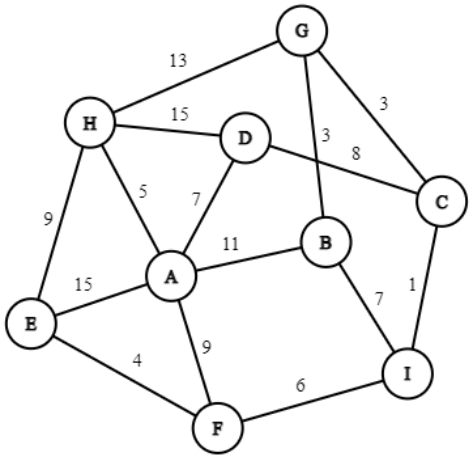
CSE 214 –Fall 2022

Recitation 10 (Graph) Student Version

1. [15 minutes] Graph Representations

Graph G:



A) Write the adjacency matrix representation of the weights of the edges of Graph G, mark -1 for no connecting edge.

Graph G	A	B	C	D	E	F	G	H	I
A									
B									
C									
D									
E									
F									
G									
H									
I									

CSE 214 –Fall 2022

Recitation 10 (Graph) Student Version

B) Construct the edge list for Graph G.

C) Given the following adjacency matrix, draw the corresponding unweighted directed graph.

Graph C	V	W	X	Y	Z
V	0	1	1	0	1
W	0	0	1	0	0
X	1	0	0	0	1
Y	1	0	0	1	0
Z	1	0	1	0	0

CSE 214 –Fall 2022

Recitation 10 (Graph) Student Version

2. [5 minutes] True/False

A) Given a graph with n vertices and m edges, the space complexity of an adjacency matrix is $O(nm)$.

B) For an undirected graph, the adjacency matrix is asymmetrical.

C) It is faster to add or remove an edge on an adjacency matrix compared to an edge list.

D) When representing a graph, an adjacency matrix is always more memory efficient than edge lists.

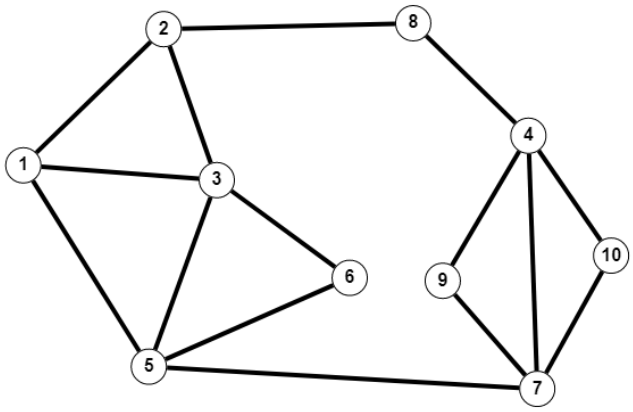
E) Dijkstra's shortest-path algorithm finds the path with the least number of edges from a source to every other vertex of a graph.

CSE 214 –Fall 2022

Recitation 10 (Graph) Student Version

3. [15 minutes] Graph Traversal

Graph T:



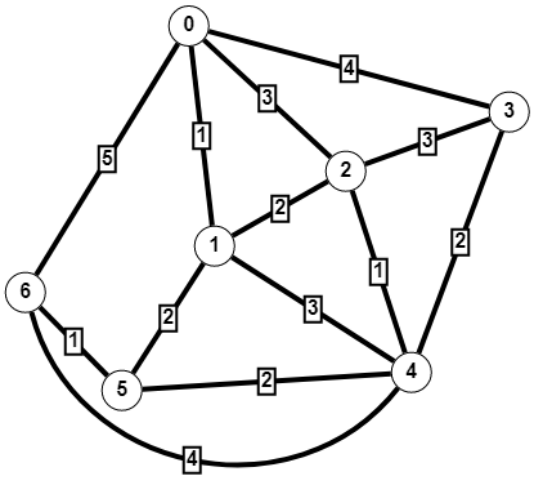
A) Perform breadth-first traversal on Graph T, begin with vertex 3 and vertex 8.

B) Perform depth-first traversal on Graph T, begin with vertex 2 and vertex 7.

CSE 214 –Fall 2022

Recitation 10 (Graph) Student Version

4. [10 minutes] Dijkstra's Shortest Path Algorithm given Graph D:



A) Simulate Dijkstra's shortest path algorithm on Graph D with source vertex 6.

S	V-S	0	1	2	3	4	5	6

B) What is the shortest path from vertex 6 to vertex 3, and what is the total weight of that path?

CSE 214 –Fall 2022

Recitation 10 (Graph) Student Version

5. [10 minutes] Graph Traversal Application

Design an algorithm that returns true if there exists a path between two vertices in the graph.

boolean isReachable(Vertex src, Vertex dst) {