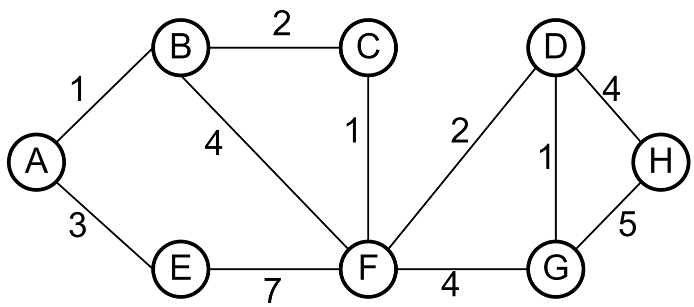
1 Dijkstra's, A*



(a) Run Dijkstra's Algorithm on the graph above starting from vertex A, breaking ties alphabetically. Fill in how the priority values change below. When you remove a node from the fringe, mark it with a check, and leave it blank for the subsequent rows. Stop when you remove G. Also sketch the resulting shortest paths tree in the end.

Node	A	B	C	D	E	F	G	H
Start	0	∞	∞	∞	∞	∞	∞	∞
Iter 1	✓							
Iter 2								
Iter 3								
Iter 4								
Iter 5								
Iter 6								
Iter 7							✓	

(b) The heuristic distance from all nodes to G is defined below. Run A*, starting from A and with G as a goal. For each entry in the table below, fill in the distance, followed by the priority value of each node, separated by a comma. Is the heuristic admissible?

u	A	B	C	D	E	F	G	H
$h(u,G)$	9	7	4	1	10	3	0	5

Node	A	B	C	D	E	F	G	H
Start	0, 9	∞	∞	∞	∞	∞	∞	∞
Iter 1	✓							
Iter 2								
Iter 3								
Iter 4								
Iter 5								
Iter 6							✓	

2 Conceptual Shortest Paths

Answer the following questions regarding shortest path algorithms for a **weighted, undirected graph**. If the statement is true, provide an explanation. If the statement is false, provide a counterexample.

(a) (T/F) If all edge weights are equal and positive, the breadth-first search starting from node A will return the shortest path from a node A to a target node B.

(b) (T/F) If all edges have distinct weights, the shortest path between any two vertices is unique.

(c) (T/F) **Adding** a constant positive integer k to all edge weights will not affect any shortest path between two vertices.

(d) (T/F) **Multiplying** a constant positive integer k to all edge weights will not affect any shortest path between two vertices.

3 Shortest Paths Algorithm Design

Two countries, Mondstadt and Fontaine, are located in the fictional world of Teyvat. The railroad system of Teyvat can be modeled as a **weighted directed graph**, with V vertices, E edges, and weights being the length of the railway. Circle the traveler wishes to take railway from Mondstadt to Fontaine, and needs to determine the shortest railway distance between them. Define the set M to be all cities in Mondstadt, and F to be all cities in Fontaine. The shortest distance between the two countries is the shortest distance between any city c_M in Mondstadt and c_F in Fontaine.

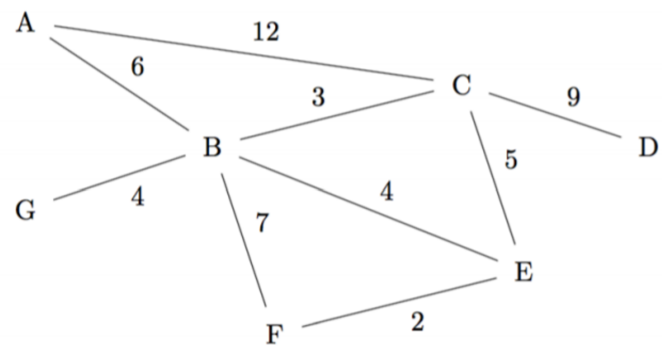
For each of the subparts below, describe an algorithm that compute the minimum railway distance from Mondstadt to Fontaine, in $O((V+E)\log V)$ time. You are able to call all graph algorithms you learned in class as a black box. *Hint: For some parts, consider modifying the graph so that running a graph algorithm yields an equivalent answer to solving the original problem.*

(a) Mondstadt only contains 1 city ($|M| = 1$), but Fontaine contains many cities ($|F| > 1$).

(b) Mondstadt contains many cities ($|M| > 1$), but Fontaine only contains one city ($|F| = 1$).

(c) Both countries contain many cities ($|M|, |F| > 1$).

4 Introduction to MSTs



(a) For the graph above, list the edges in the order they're added to the MST by Kruskal's and Prim's algorithm. Assume Prim's algorithm starts at vertex A. Assume ties are broken in alphabetical order. Denote each edge as a pair of vertices (e.g. AB is the edge from A to B).

Prim's algorithm order:

Kruskal's algorithm order:

(b) True/False: Adding 1 to the smallest edge of a graph G with unique edge weights must change the total weight of its MST.

(c) True/False: If all the weights in an MST are unique, there is only one possible MST.

(d) True/False: The shortest path from vertex u to vertex v in a graph G is the same as the shortest path from u to v using only edges in T, where T is the MST of G.