1 Conceptual Shortest Paths

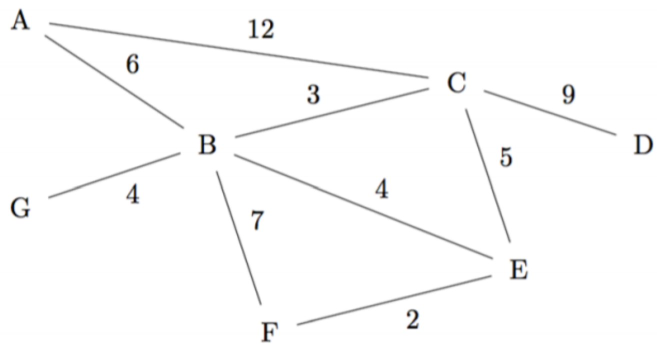
Answer the following questions regarding shortest path algorithms for a **weighted, undirected graph**. If the question is T/F and the statement is true, provide an explanation. If the statement is false, provide a counterexample.

(a) (T/F) If all edge weights are equal and positive, 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 vertices.

2 Warmup with 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) Is there any vertex for which the shortest paths tree from that vertex is the same as your Prim MST?

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

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

(e) True/False: Given any cut, the maximum-weight crossing edge is in the maximum spanning tree.

3 Graph Algorithm Design

For each of the following scenarios, write a brief description for an algorithm for finding the MST in an undirected, connected graph G.

(a) If all edges have edge weight 1. Hint: Runtime is O(V+E)

(b) If all edges have edge weight 1 or 2. Hint: Use your algorithm from part (a)