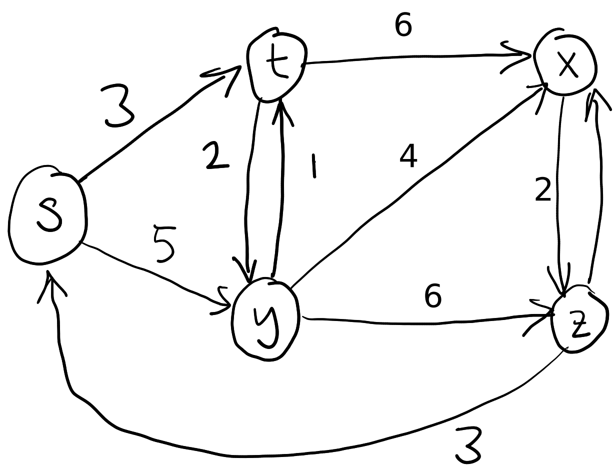
Shortest Path Exercises

1)



Compute the shortest paths and distances to all other nodes for the graph on the left.

Show that the shortest paths are not unique.

Straightforward to compute shortest paths

There are multiple shortest paths. For example:
 s -> t -> y
 s -> y
both have weight 5

2) Run Bellman-Ford algorithm on the graph above.

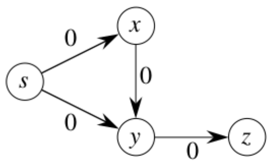
Go ahead and do this.

3) Give an O(V+E) algorithm that checks whether the output of a single-source shortest path is correct.

To check that the edges have fully relaxed, you could run a single round of Bellman-Ford. No distances or predecessors should change with the Relax function.

Furthermore, you should check that s.pi = Nil and s.d = 0 and that v.d = Inf for any vertices outside the connected component containing s. The first part is easy. For the second you might need to run a round of DFS from s to find its out-component; when this restarts you need to verify that all of the distances are infinite.

4) Someone tells you that they think that Dijkstra's algorithm "relaxes edges of every shortest path in the order that they appear in the path". Show that this is incorrect by constructing a directed graph for which Dijkstra could relax the edges out of order.



Dijkstra's algorithm could relax edges in the order $(s, y), (s, x), (y, z), (x, y)$. The graph has two shortest paths from s to z: $\langle s, x, y, z \rangle$ and $\langle s, y, z \rangle$, both with weight 0. The edges on the shortest path $\langle s, x, y, z \rangle$ are relaxed out of order, because (x, y) is relaxed after (y, z).