EC 504
Spring, 2021
HW 5

Due Tuesday, March 30, 8PM on Gradescope.

1. (10 points) Assume you have a scheduling problem with 10 customers. Customer i has value V_i, and must be processed before deadline T_i. The processing time of each customer is exactly one unit of time. The values of V_i and T_i are listed below as arrays:

   ```
   V = {3,4,7,5,2,3,5,8,9,6}
   T = {2,3,4,3,1,5,7,3,4,6}
   ```

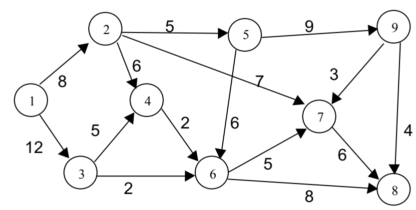
   Find the optimal sequence of jobs that can be scheduled in order to complete as much value as possible.

2. (10 points) Assume you have a scheduling problem with 10 customers. Customer i has processing time T_i. The value of each customer is the same. The objective is to minimize the sum across all customers of the completion time of each customer. The values of T_i are listed below as an array:

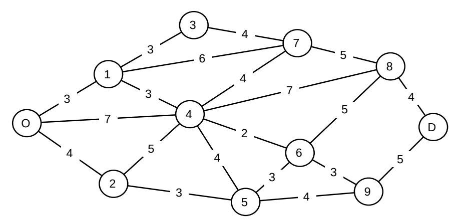
   ```
   T = {2,3,4,9,1,5,7,8,10,6}
   ```

   Find the optimal sequence of jobs to minimize the sum of the completion times.

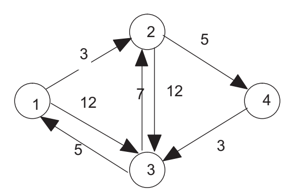
3. (10 points) Consider the directed weighted graph on the right. Show the distance estimates computed by each step of the Bellman-Ford algorithm for finding a shortest path tree in the graph, starting from vertex 3 to all other vertices.



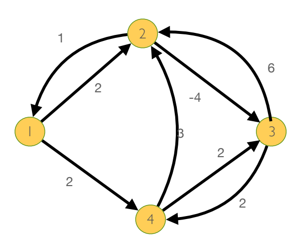
4. (10 points) Consider the weighted, undirected graph shown on the right. Assume that the edges can be traveled in both directions. Illustrate the steps of Dijkstra's algorithm for finding a shortest path from vertex O to vertex D.



5. (10 points) Consider the graph indicated on the right. This is a directed graph. Use the Floyd-Warshall algorithm to find the shortest distance for all pairs of vertices. Show your work as a sequence of 4 by 4 tables.



6. (10 points) Consider the graph indicated on the right. This is a directed graph. Note the negative weight edge from 2 to 3. Use Johnson's algorithm, followed by Dijkstra's algorithm, to find the shortest distance for all pairs of vertices.



7. (10 points) Explain how you modify Dijkstra's shortest path algorithms on a directed graph with non-negative edge weights to count the number of shortest paths from a given origin n to a destination vertex d.

8. (10 points) In city streets, the length of an edge often depends on the time of day. Suppose you have a directed graph of streets connecting vertices that represent intermediate destinations, and you are given the travel time on the edge as a function of the time at which you start to travel that edge. Thus, for edge e, you are given $d_e(t)$, the time it takes to travel edge e if you start at time t. These travel times must satisfy an interesting ordering property: You can't arrive earlier if you started later. That is, if $s < t$, then $s + d_e(s) <= t + d_e(t)$. Suppose that you start at time 0 at the origin vertex 1. Describe an algorithm for computing the minimum time path to all vertices when travel times on edges are time dependent and satisfy the ordering property.