2. (32 marks) Consider the following edge-weighted directed graph:

![Graph Image]

a) (8 marks) Run Dijkstra’s algorithm (p, 595) on the directed graph above using vertex $x$ as the source vertex. In the style of Figure 24.6 in the textbook, show the $d$ and $\pi$ values and the vertices in set $S$ after each iteration of the while loop.

b) (8 marks) Run the Bellman-Ford algorithm (p, 588) on the directed graph above using vertex $x$ as the source vertex. Relax edges in lexicographic order in each pass, and in the style of Figure 24.4 on the textbook, show the $d$ and $\pi$ values after each pass. Finally, give the boolean value returned by the algorithm.

c) (8 marks) Re-do part (a) with the weight of arc $(y, v)$ reset to 7.

d) (8 marks) Re-do part (b) with the weight of arc $(y, v)$ reset to 7.

3. (24 marks) Consider the following decision problems:

**Dumbbell Subgraph (DS)**

*Input:* An undirected graph $G = (V, E)$ and two positive integers $k, l \geq 1$.

*Question:* Are there two cliques $C_1$ and $C_2$ and a simple path $P$ in $G$ such that $C_1$ and $C_2$ have $\geq k$ vertices apiece, $P$ has $\geq l$ edges, $P$ connects $C_1$ and $C_2$, the cliques and path do not have any edges in common, and the only vertices that $P$ shares with $C_1$ ($C_2$) is its connection-vertex?

**Bounded-Weight Subset Cover (BWSSC)**

*Input:* A set $I = \{i_1, \ldots, i_n\}$ of items, a set $R = \{r_1, \ldots, r_b\}$ of subsets of $I$, an integer-valued subset-weight function $w()$ such that for each $r_x \in R$, $w(r_x) > 0$, a subset $N \subseteq I$, and integers $0 < k_1 \leq k_2$.

*Question:* Is there a subset $R' \subseteq R$ such that $\cup_{r \in R'} r = N$ and $k_1 \leq \sum_{r \in R'} w(r) \leq k_2$?
a) (12 marks) Prove that problem DS is $NP$-complete by (1) showing that this problem is in $NP$ and (2) giving a polynomial-time many-one reduction (algorithm + proof of correctness) to this problem from an $NP$-hard problem.

b) (12 marks) Prove that problem BWSSC is $NP$-complete by (1) showing that this problem is in $NP$ and (2) giving a polynomial-time many-one reduction (algorithm + proof of correctness) to this problem from an $NP$-hard problem.