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 y as the source vertex. In the style of Figure 24.6 in the textbook, show the d and π 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 y 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 π 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 (x, u) reset to -5.

d) (8 marks) Re-do part (b) with the weight of arc (x, u) reset to -5.

3. (18 marks) Consider the following decision problems:

**k-Weight Simple Path (kWSP)**

*Input:* An undirected edge-weighted graph \( G = (V, E, w) \), an integer \( l, 0 < l \leq |V| \), such that for each edge \( e \in E \), \( 1 \leq w(e) \leq l \), vertices \( x, y \in V \), and an integer \( k > 0 \).

*Question:* Is there a simple path between \( x \) and \( y \) in \( G \) whose summed edge-weight is \( \geq k \)?

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

*Input:* A set \( I = \{i_1, \ldots, i_n\} \) of items, a set \( R = \{r_1, \ldots, r_k\} \) 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) (9 marks) Prove that problem $k$WSP 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) (9 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.