CS 340.001: Algorithms and Data Structures  
**Homework 7**  
Due on Dropbox by 1:00 PM on Wednesday, March 30, 2011

1. (1 point each) The directed graph above represents a set of scheduling conditions for this course, where an edge from node Y to node Z signifies that the date associated with event Y must precede the date associated with event Z. For each of the event orderings below, indicate whether there is any topological sort of this graph for which the ordering could be valid.
   (a) $Q_1$ precedes $X_1$.  (b) $P_3$ precedes $D_2$.  (c) $Q_3$ precedes $H_6$.  (d) $X_2$ precedes $H_6$.  (e) $X_2$ precedes $D_1$.

2. (5 points each) Recall Dijkstra’s Algorithm for determining the shortest path from a distinguished vertex in a weighted graph to every other vertex in the graph. Pseudocode for this algorithm is listed below:
   ```
   void Dijkstra(vertex v, graph G)
   {
     // NBR_VERTICES is the number of vertices in G.
     bool finalized[NBR_VERTICES];  // finalized marks the finished vertices.
     float dist[NBR_VERTICES];      // dist[i] = min cost between v & vertex #i.
     int predecessor[NBR_VERTICES]; // predecessor[i] = vert before #i in shortest path.
     for (int i = 1; i <= NBR_VERTICES; i++)
     {
       finalized[i] = false;
       dist[i] = cost[index(v), i];  // cost is the matrix of G's edge costs.
       predecessor[i] = index(v);     // index is ordinal number of G's vertices.
     }
     finalized[index(v)] = true;
     dist[index(v)] = 0;
     for (int j = 2; j < NBR_VERTICES; j++)
     {
       choose vertex u such that (finalized[index(u)] == false) and dist[index(u)] is minimal.
       finalized[index(u)] = true;
       for (every vertex w with finalized[index(w)] == false)
       {
         if (dist[index(w)] > dist[index(u)] + cost[index(u), index(w)])
         {
           dist[index(w)] = dist[index(u)] + cost[index(u), index(w)];
           predecessor[index(w)] = index(u);
         }
       }
     }
   }
   ```

   (a) Alter the algorithm’s original pseudocode so that it also produces a count of how many different minimum paths there are between the distinguished vertex and every other vertex in the graph.
   (b) Alter the algorithm’s original pseudocode so that if there is more than one minimum path between the distinguished vertex and another vertex, then a path with the fewest number of edges is chosen.
3. EXTRA CREDIT (5 points) Use Dijkstra's Algorithm to determine the shortest paths from node A to every other node in the weighted directed graph below. (For your convenience, three copies of this graph have been placed below: the original graph, a weightless version, and an edgeless version.)

4. EXTRA CREDIT (5 points) Prove that if T is a spanning tree for an undirected graph $G = (V, E)$, then adding any edge in $E$ that is not in $T$ creates one and only one cycle. For instance, in the graph below, the darkened edges form a spanning tree. Adding edge $DG$ to that tree produces the unique cycle $ADGFCA$.

You must provide your own solutions to these problems in a clearly presented Word document. Obtaining solutions from any outside source is considered academic misconduct. The only person with whom you may discuss these problems is the instructor.