Problem 1)
I have modified the algorithm so that if an edge is negative we take the prior removed neighbors of u and put them back on the priority queue to re-evaluate their distances with the negative edge. If there is a negative edge then each vertex will be checked more than once. So, it will take more than the original O(|E|+|V|log|V|) run time of Dijkstra's algorithm. But the algorithm will work well in practice and will give us the proper distances including the negative edges. Otherwise, the efficiency for my modified version of Dijkstra will not decrease for a positive weighted graph.

```
DIJKSTRA (G, w, s):
1   INITIALIZE-SINGLE-SOURCE (G, s)
2   S = 0
3   Q = G.V
4   while Q != 0
5         u = EXTRACT-MIN(Q)
6         S = S ∪ {u}
7         for each vertex v in G.Adj[u]:
8               if w(u, v)<0:                  // if negative edge
9                     RELAX(u, v, w)          //evaluate its vertex distance
10                    for each vertex s in S.Adj[u]: // go through removed neighbors
11                          Q = Q ∪ {s}       //add removed neighbors back to Q
12                          RELAX(u, s, w)    //re-evaluate their distances
13              else: RELAX(u, v, w)
```

Citations: 1) http://cs.stackexchange.com/questions/2482/using-dijkstras-algorithm-with-negative-edges, Author: Kaveh and nhahtdh, Title: Using Dijkstra's algorithm with negative edges?, Date: Jun. 52, 2012
2) http://stackoverflow.com/questions/6799172/negative-weights-using-dijkstras-algorithm?rq=1, Author: amit, Title: Negative weights using Dijkstra's Algorithm, Date: Jul. 23, 2011
3) http://stackoverflow.com/questions/10799938/can-we-change-dijkstras-algorithm-to-work-with-negative-weights, Author: Leif Ericson and Cherish, Title: Can we change Dijkstra's Algorithm to work with negative weights?, Date: Nov. 7, 2013
4) Intro to Algorithms 3edition p.649 and p.658-661, Author: CLRS, Date: 2009

Problem 2)
If there is a negative cycle in graph G then there will be at least one vertex $v_i$ that belongs to that cycle such that $D_{ii} < 0$. So, it is possible to detect the presence of a negative cycle by checking if there is a negative entry in the diagonal of the matrix. The Floyd-Warshall Algorithm takes $\Theta(n^3)$ running time because of its triply nested for loop. Below I have modified the algorithm by checking the condition $D_{ii} < 0$ in the third loop and setting the associated vertex to $-\infty$. Thus, by doing this we do not increase the run time of the algorithm at all.

```
FLOYD-WARSHALL(W):
1   n = W.rows
2   D^(0) = W
3   for k = 1 to n:
4         let D^(k) = (d_ij^(k)) be a new nxn matrix
5         for i = 1 to n:
6               for j = 1 to n:
7                     d_ij^(k) = min(d_ij^(k-1), d_ik^(k-1) + d_kj^(k-1))
8                     if d_ii^(k) < 0:       // if a diagonal entry is negative
9                           d_ij^(k) = – ∞ // then set corresponding vertex in path to – ∞
10  return D^(n)
```

Citations: 1) http://www.or.uni-bonn.de/~hougardy/paper/Floyd-Warshall.pdf, Author: Stephan Hougardy, Title: The Floyd-Warshall Algorithm on Graphs with Negative Cycles, Date: 2010
2) http://cseweb.ucsd.edu/classes/sp02/cse101/homework/101_hw4_sol.pdf, Author: Unknown, Title: Homework 4 Solutions, Date: 6/10/2002
3) http://stackoverflow.com/questions/15709277/floyd-warshall-with-negative-cycles-how-do-i-find-all-undefined-paths, Author: mseln, Title: Floyd-Warshall with negative cycles. How do I find all undefined paths?, Date: Mar. 30, 2013.
4) Intro to Algorithms 3edition p.693-695, Author: CLRS, Date: 2009