Problem 1.

To detect a cycle in an undirected graph $G$ we will slightly DFS as follows. If \textit{DFS\_visit}(u) encounters a neighbor of $u$, call it $v$ that is gray and $v$ is not the parent of $u$, then $v$ is an ancestor of vertex $u$ in the DFS tree. This means there is a path $P_{vu}$ from vertex $v$ to $u$ that does not use the edge $(u,v)$. Thus we could extend this path by edge $(u,v)$ and obtain a cycle. If for any connected component of $G$ we have the situation described above our modified DFS returns “cycle”, otherwise it returns “no cycle.” The running time of the algorithm is $O(|V|)$ and is independent of $E$, because if $|E| > |V|-1$ then there is a cycle in the graph and this will be detected in $O(|V|)$ time.

Problem 2.

Substitute the for loop on line 2 of the BELLMAN-FORD algorithm on page 588 of CLRS with a while loop where the condition to break out of the loop is that there is no change to the any of the $d[u]$ values between two consecutive iterations. By the definition of shortest path and the definition of $m$ in the case when there are no negative-weights (no negative cycles) this change will allow to algorithm to terminate after $m+1$ iterations.

Problem 3.

If there is a negative cycle in $G$ then there will be at least one vertex $v_i$ that belongs to that cycle such that $D_{ii} < 0$. So, to detect a negative cycle using the output $D[]$ of the FLOYD-WARSHALL algorithm all we need to do is to check if any of the diagonal entries is negative.

Problem 4.

The following is an example that shows that the proposed heuristic for vertex cover does not have an approximation ratio of 2. Vertices selected by the heuristic are marked black and vertices in the optimal cover are marked gray. The ratio is $9/4 > 2$. 
**Problem 5.**

Traverse the graph in a DFS manner. When during the traversal we reach a leaf node (that is not in the cover) do the following:

1. Put the parent node of the discovered leaf node in the vertex cover
2. Delete the leaf node and the edge between the parent node and the leaf node from the graph and continue with the traversal.

Since by the definition of a leaf there is exactly one edge touching a leaf and this edge needs to be covered, and since there are only two nodes that can cover this edge (the leaf node or the parent node) the vertex cover induced by always picking the parent node is at least as good as the one induced by picking the leaf node. Hence the vertex cover produced by this algorithm is optimal.