CMPS 101
Winter 2009
Homework Assignment 5

1. (1 Point) p. 538: 22.2-6
   There are two types of professional wrestlers: "good guys" and "bad guys." Between any pair of professional wrestlers, there may or may not be a rivalry. Suppose we have *n* professional wrestlers and we have a list of *r* pairs of wrestlers for which there are rivalries. Give an $O(n+r)$-time algorithm that determines whether it is possible to designate some of the wrestlers as good guys and the remainder as bad guys such that each rivalry is between a good guy and a bad guy. If it is possible to perform such a designation, your algorithm should produce it. (Hint: figure out how to use BFS to solve this problem.)

2. (1 Point) p.547: 22.3-1
   Make a 3-by-3 chart with row and column labels WHITE, GRAY, and BLACK. In each cell (i, j), indicate whether, at any point during a depth-first search of a directed graph, there can be an edge from a vertex of color *i* to a vertex of color *j*. For each possible edge, indicate what types it can be.

3. (1 Point) p.547: 22.3-2
   Show how depth-first search works on the graph of Figure 22.6 (p.548). Assume that the **for** loop of lines 5-7 of the DFS procedure considers the vertices in alphabetical order, and assume that each adjacency list is ordered alphabetically. Show the discover and finishing times for each vertex, and show the classification of each edge.

4. (1 Point) p.548: 22.3-9
   Modify the pseudocode for depth-first search so that it prints out every edge in the directed graph together with its type. (Hint: use the result stated in the last paragraph of page 546.)

5. (1 Point) p.549: 22.3-11
   Show that a depth-first search of an undirected graph *G* can be used to identify the connected components of *G*, and that the depth-first forest contains as many trees as *G* has connected components. More precisely, show how to modify depth-first search so that each vertex *v* is assigned an integer label $cc[v]$ between 1 and *k*, where *k* is the number of connected components of *G*, such that $cc[u] = cc[v]$ if and only if *u* and *v* are in the same connected component.