SOLUTIONS TO CSE 331 SAMPLE MID-TERM-I

Please do not read anything into the kind of problems in the sample mid-term. Overall, the mid-term will be harder than this sample mid-term (but still easier than the homeworks). The main purpose of this sample mid-term was to give you an idea of the format of questions. Also you can get an idea of how much detail is expected from your answers in the exam from the solutions below.

1. ($8 \times 5 = 40$ points) Answer True or False to the following questions and briefly JUSTIFY each answer. A correct answer with no or totally incorrect justification will get you 2 out of the total 5 points. (Recall that a statement is true only if it is logically true in all cases while it is is false if it is not true in some case).

   (a) For any instance of the stable marriage problem with n men and n women, there are $n! = n \times (n-1) \times \ldots \times 1$ many possible perfect matchings.

   True. See Solution to Problem 2 in HW0.

   (b) Consider $f(n) = \log\log n$ and $g(n) = 10^{10^{10^{10^{10}}}}$. Then $f(n)$ is $O(g(n))$.

   False. $g(n)$ is a constant (albeit a big one) and does not increase with n. However, $f(n)$ does increase with n (though slowly). Thus, for some large enough n, $g(n) \leq f(n)$ and thus, $g(n)$ is $O(f(n))$ (and not the other way round).

   (c) Consider $f(n) = n^n$ and $g(n) = 2^{400n}$. Then $f(n)$ is $\Omega(g(n))$.

   True. Note that $f(n) = 2^{n\log_2 n}$ and since $n\log_2 n \geq 400n$ for $n \geq 2^{400}$, $f(n) \geq g(n)$ for every $n \geq 2^{400}$, which by definition implies that $f(n)$ is $\Omega(g(n))$.

   (d) There is an algorithm that sorts n numbers $a_1, \ldots, a_n$ where $a_i \in \{0,1\}$ for each $i \in [n]$ in $O(n)$ time.

   True. Consider the following algorithm. Do a scan through $a_1, \ldots, a_n$ and count the number of $i \in [n]$ such that $a_i = 0$. Let this number be n_0. Then output n_0 0's followed by $n - n_0$ 1's.

   (e) BFS is a linear time algorithm. (Recall that an algorithm is a linear time algorithm if it runs in time $O(N)$ on inputs of size N.)

   True. Recall that we have shown that BFS runs in time $O(m+n)$ when the graph is represented in the adjacency list format. Further, in the adjacency list format $N = \Theta(n+m)$, which implies the run time is $O(N)$.

   (f) For any graph, there is a unique BFS tree for it.

   False. Consider the cycle on four vertices: v_1, v_2, v_3, v_4 such that (v_i, v_{i+1}) for $1 \leq i \leq 3$ and (v_4, v_1) are the edges. Consider a BFS run that starts from v_1. Note that v_3 can be discovered from either v_2 or v_4 and each choice leads to a different BFS tree.

   (g) Every directed graph on n vertices with at least $n-1$ edges is connected.

   False. Consider the following DAG on three vertices A, B, C with directed edges $(A,B), (B,C), (A,C)$. In this graph C is not connected to A but it has $n = 3$ edges.

(h) Given a graph on n vertices in its adjacency matrix, there is an $O(n^2)$ time algorithm to convert it into its adjacency list representations.

True. In short here is the algorithm: go through the matrix row by row and for the vertex u corresponding to the current row, add a list of vertices w such that the entry for (u, w) is a 1. Each row takes $O(n)$ time and there are n rows, which makes for a total running time of $O(n^2)$.