CS211: Problem Set 1

Due Friday, February 6

1. **(2.1-8, CLR) We can extend the O notation to the case of two parameters n and m that can go to infinity independently at different rates. For a given function $g(n,m)$, we denote $O(g(n,m))$ the set of functions**

   $O(g(n,m)) = \{f(n,m)$: there exist positive constants c, n_0, m_0 such that $0 \le f(n,m) \le cg(n,m)$ for all $n \ge n_0, m \ge m_0$ $\}$

   Give corresponding definitions for $\Omega(g(n,m))$ and $\Theta(g(n,m))$.

2. **(2.3) Take the following list of functions and arrange them in ascending order of growth rate. That is, if function $g(n)$ immediately follows function $f(n)$ in your list, then it should be the case that $f(n)$ is $O(g(n))$.**

$$\begin{array}{ll} f_1(n) = n^{2.5} & f_2(n) = \sqrt{2n} \\ f_3(n) = n + 10 & f_4(n) = 10^n \\ f_5(n) = 100^n & f_6(n) = n^2 logn \end{array}$$

3. **(2.4) Take the following list of functions and arrange them in ascending order of growth rate. That is, if function $g(n)$ immediately follows function $f(n)$ in your list, then it should be the case that $f(n)$ is $O(g(n))$.**

$$\begin{array}{ll} f_1(n) = 2^{\sqrt{logn}} & f_2(n) = 2^n \\ f_3(n) = n^{4/3} & f_4(n) = n(logn)^3 \\ f_5(n) = n^{logn} & f_6(n) = 2^{2^n} \\ f_7(n) = 2^{n^2} & \end{array}$$

4. **Suppose that each row of an $n \times n$ array A consists of 1's and 0's such that, in any row i of A, all the 1's come before any 0's in that row. Suppose further that the number of 1's in row i is at least the number in row $i+1$, for $i = 0, 1, ..., n-2$. Assuming A is already in memory, describe a method running in $O(n)$ time for counting the number of 1's in the array.**

5. **(3.2) Give an algorithm to detect whether a given undirected graph contains a cycle. If the graph contains a cycle, then your algorithm should output one. (It should not output all cycles in the graph, just one of them.) The running time of your algorithm should be $O(m+n)$ for a graph with n nodes and m edges.**