CS 240 Midterm Review (Module 1–7)

Asymptotic Analysis

- Problem instance (I) – *input* for the specified problem
- Problem solution – *output* for the specified problem instance
- Problem size – Size(I) = size of instance I
- Algorithm - a step-by-step process for carrying out a series of computations
  - An algorithm A solves a problem P if, for every instance I of P, A computes a valid solution for I in <u>finite</u> time
- RAM model
  - Assume any memory access & primitive operation is constant time
  - Assume infinite amount of memory
  - Sequential operation
  - Running time is determined by the # of memory accesses & primitive operations
- Order notations
  - $\underline{f(n) \in O(g(n))}$ if $\exists\ c > 0$ and $n_0 > 0$ such that $0 \leq f(n) \leq cg(n)\ \forall\ n \geq n_0$
    - f "grows no faster than" g
    - f is "upper-bounded" by g
  - $\underline{f(n) \in \Omega(g(n))}$ if $\exists\ c > 0$ and $n_0 > 0$ such that $0 \leq cg(n) \leq f(n)\ \forall\ n \geq n_0$
    - f "grows no slower than" g
    - f is "lower-bounded" by g
  - $\underline{f(n) \in \Theta(g(n))}$ if $\exists\ c_1, c_2 > 0$ and $n_0 > 0$ such that $0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)\ \forall\ n \geq n_0$
    - f and g grow at the same rate
  - $\underline{f(n) \in o(g(n))}$ if $\forall\ c > 0, \exists\ n_0 > 0$ such that $0 \leq f(n) < cg(n)\ \forall\ n \geq n_0$
    - f *always* grows slower than g, given a n_0
  - $\underline{f(n) \in \omega(g(n))}$ if $\forall\ c > 0, \exists\ n_0 > 0$ such that $0 \leq cg(n) < f(n)\ \forall\ n \geq n_0$
    - f *always* grows faster than g, given a n_0
  - Suppose $L = \lim_{n\to\infty} \dfrac{f(n)}{g(n)}$
    - If $L = 0$ then $f \in o(g)$
    - If $0 < L < \infty$ then $f \in \Theta(g)$
    - If $L = \infty$ then $f \in \omega(g)$

Thing

-