Tutorial 1

CS 213: Data Structures and Algorithms

1. What is the time complexity of binary addition and multiplication? How much time does it take to do unary addition?
2. If $f(n) = \mathcal{O}(F(n))$ and $g(n) = \mathcal{O}(G(n))$, show that $h(n) = \dfrac{f(n)}{G(n)} = \mathcal{O}\left(\dfrac{F(n)}{g(n)}\right)$.
3. Let $f(n)$ and $g(n)$ be asymptotically non-negative. Prove that $\max(f(n), g(n)) = \Theta(f(n) + g(n))$.
4. Is $2^{n+1} = \mathcal{O}(2^n)$? Is $2^{2n} = \mathcal{O}(2^n)$?
5. There is a stack of n dosas on a tava, all having distinct radii. We want to serve them in order of increasing radii. Only two operations are allowed:
   (a) serve the top dosa,
   (b) insert a spatula (flat spoon) in the middle, say after the first k, hold up this partial stack, flip it upside-down, and put it back.

   Design a data structure to represent the tava, to input a given tava, and to produce an output in sorted order. What is the time complexity of your algorithm?
6. One problem is to check whether a string of opening and closing brackets of various types is a valid bracket sequence. For example `{(){}[[]]}` and `([[[]]]{})` are valid, while `(){(` and `(()))` are invalid. Write a stack based algorithm to check whether a bracket string is valid.
7. *The mess table queue.* There is a common mess for K hostels. Each hostel has some $N_1, \ldots, N_k$ students. These students line up to pick up their trays in the common mess. However, the queue is implemented as follows: If a student sees a person from his/her hostel, she/he joins the queue behind this person. This is the "enqueue" operation. The "dequeue" operation is as usual, at the front. Think about how you would implement such a queue. What would be the time complexity of enqueue and dequeue? Do you think the average waiting time in this queue would be higher or lower than a normal queue? Would there be any difference in any statistic? If so, what?