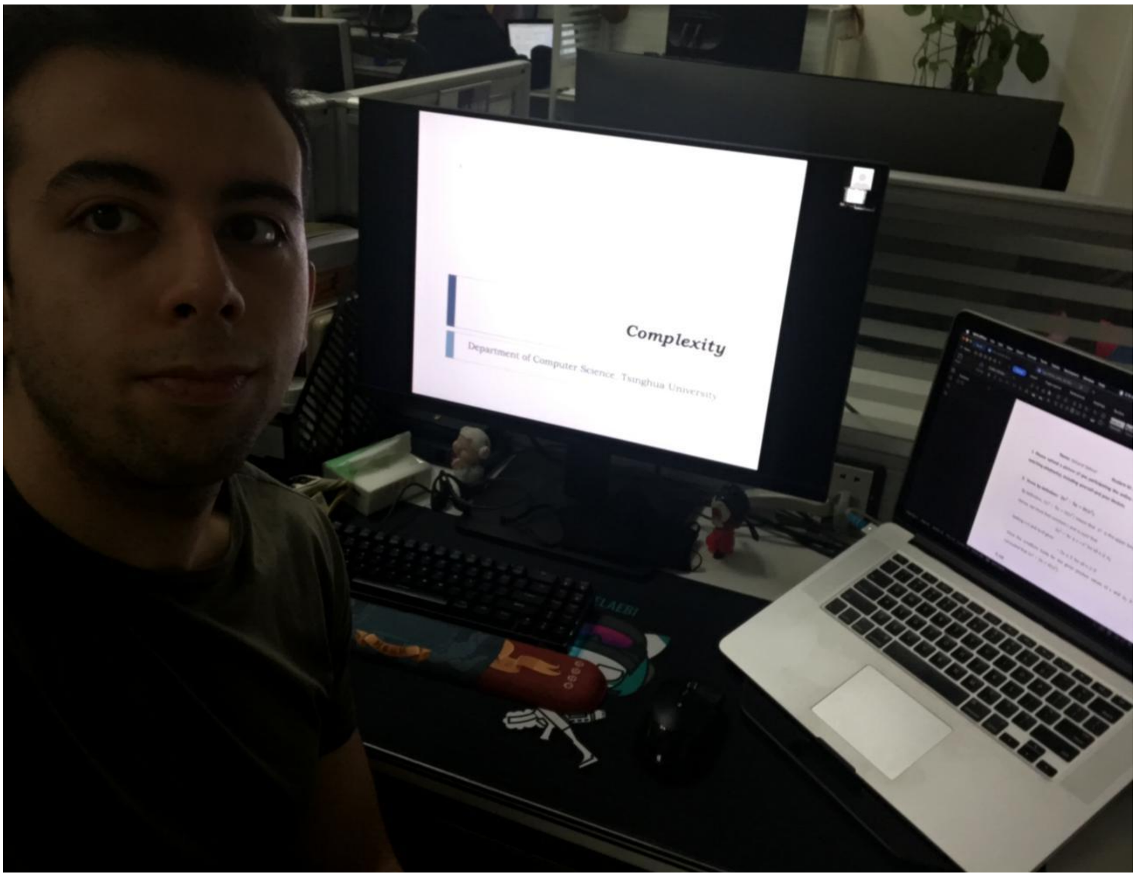
Homework - Week 9

Name: Sahand Sabour **Student ID:** 2020280401

1. Please upload a picture of you participating the online meeting of the course (or watching playbacks), including yourself and your devices.



2. Prove by definition: $2n^2 - 3n = O(n^2)$.

By definition, $2n^2 - 3n = O(n^2)$ means that n^2 is the upper bound for $2n^2 - 3n$.

Hence, we must find constant c and n_0 such that

$$2n^2 - 3n \leq c * n^2 \text{ for all } n \geq n_0$$

Setting c=2 and n_0=0 gives

$$-3n \leq 0 \text{ for all } n \geq 0$$

Since the condition holds for the given positive values of c and n_0, it can be concluded that $2n^2 - 3n = O(n^2)$.

3. Let

$$\mathbf{P(n) = \sum_{i=0}^{d} a_i n^i}$$

Where $a_d > 0$, be a degree- polynomial in , and let be a constant. Use the definitions of the asymptotic notations to prove:

$$\textbf{If } k \geq d, \textbf{ then } p(n) = O(n^k)$$

By definition, p(n)= $O(n^k)$ means that n^k is the upper bound for p(n). Hence, we must find constant c and n_0 such that

$$P(n) \leq c * n^k \text{ where } k > d \text{ and } a_d > 0 \text{ for all } n \geq n_0$$

Accordingly, P(n) can be rewritten as

$$P(n) = \sum_{i=0}^{d} a_i n^i = a_d n^d + \sum_{i=0}^{d-1} a_i n^i = a_d n^d + n^d\left(\sum_{i=0}^{d-1} a_i n^{i-d}\right) = n^d\left(a_d + \sum_{i=0}^{d-1} a_i n^{i-d}\right)$$

Therefore, it can be observed that the largest power in the sum $\sum_{i=0}^{d-1} a_i n^{i-d}$ would be -1 (when i=d-1) and the rest of the powers for n would be less than that. Hence, the values in this sum tend to move closer to zero as the value of n and/or its power increases, with $n \geq 2$. Setting the upper bound for this sum to 1, given that n^k increases as fast or faster than n^d since $k \geq d$, gives

$$P(n) = n^d\left(a_d + \sum_{i=0}^{d-1} a_i n^{i-d}\right) \leq n^d(a_d + 1) \leq n^k(a_d + 1) \text{ where } a_d \geq 0 \text{ and } n \geq 2$$

Setting $c = a_d + 1$ and n_0= 2 satisfies the above equation and therefore, these positive values could be used to prove the given statement.

4. Show that the majority element problem can be reduced to the sorting problem, following the three steps of reduction.

The input to the majority element problem is an array A of n numbers.

Step 1: The same array A could be used as the input to the sorting problem.

Step 2: Using the sorting ∝ convex hall problem sorts the array in ascending order and the output would be A', sorted version of A, with lower bound of Ω(nlogn).

Step 3: Select the median of A' and check if it's the majority element via counting the number of its occurrences and comparing it to the length of array, given that majority element M appears more than half of the array: count(M) > len(A)/2. This would be completed in runtime with upper bound of O(n).