# CS624: Analysis of Algorithmns

# Midterm exam 1 – practice

Instructor – Nurit Haspel

## General Instructions

1. You may use the class notes and homework assignments. No books or any other printed/copied material is allowed. Electronic devices must be turned off.
2. The work is to be your own and you are expected to adhere to the UMass Boston honor system.
3. Write your answers in the available spaces, using the back of the page if needed. Write clearly and concisely and try to avoid cursive.
4. Please explain your answers if needed **but do it briefly**.
5. You may use any proof technique we showed in class or any other technique, as long as it constitutes a mathematical proof. Remember that a proof by example is generally good only to show that something is NOT true.
6. If you base your answer on a homework question state exactly which question it was.

## Practice Questions

1. (25%) Order of growth:

   Use the substitution/induction method to solve the recurrence formula: $T(n) = 8T(n/2) + n^3$. Assume $T(2) = d$ (some constant). The result should be $T(n) = O(n^3 \log n)$ I know this can be shown easily with the master theorem, but please use induction and do not skip stages. Remember that you really have to show that $T(n) \leq Cn^3 \log n$ for some constant C.

   **Answer:**

   - Base case = T(2) = d.
   - Inductive hypothesis: Assume $T(k) \leq Ck^3 \log k$ for any $k < n$.
   - Use inductive hypothesis to prove for n. $T(n) = 8T(n/2) + n^3 \leq 8C(\frac{n}{2})^3 \log \frac{n}{2} + n^3 = Cn^3(\log n - 1) + n^3 = Cn^3 \log n + n^3(1 - C)$. This expression is $\leq Cn^3 \log n$ for any $C \geq 1$.

2. Sorting (25%) Let $S$ be an unsorted array of $n$ integers. Give an algorithm that finds the pair $x, y \in S$ that maximizes $|x - y|$. Your algorithm must run in $O(n)$ worst-case time.

   **Answer:** This is simply finding the maximum and the minimum...

3. Heaps (25%): In a max-heap of size n, represented as discussed in class, in what index(es) can the smallest element reside? Explain. Assume all the $n$ numbers are different.

   **Answer:** We discussed this in class. The smallest element can be in any of the leaves, that is, in position $\lfloor \frac{n}{2} \rfloor$ to $n$.

4. Sorting bounds (25%): A researcher discovered a comparison-based sorting algorithm that runs in $O\big(n\log(\sqrt{n})\big)$. Given the existence of an $\Omega(n\log n)$ lower bound for sorting, how can this be possible? (**Hint:** It IS possible).

   **Answer:** It looks like a trick question but it really isn't. The square root is just a factor of 1/2. $O(n\log\sqrt{n}) = O(n\log(n^{\frac{1}{2}})) = O(\frac{1}{2}n\log n) = O(n\log n)$, so it's ok.