National University of Computer & Emerging Sciences, Karachi
Fall-2017 CS-Department
MidTerm - 1
18th September 2017, 11:00 – 12:00 noon

Course Code: CS302	**Course Name:** Design and Analysis of Algorithm
Instructor Name / Names: Muhammad Atif Tahir, Subhash Sagar and Zeshan Khan	
Student Roll No:	**Section No:**

Instructions:

- Return the question paper.
- Read each question completely before answering it. There are **4 questions.**
- In case of any ambiguity, you may make assumption. But your assumption should not contradict any statement in the question paper.

Time: 60 minutes. **Max Marks**: 10 points

Question 1: **(1.5 points)**

a) Write down the **Worst** and **Best Case** complexities of the following Sorting Algorithms. **[0.7 points]**

Algorithms	Worst Case	Best Case
Insertion Sort		
Merge Sort		
Heap Sort		
Quick Sort		
Counting Sort		
Radix Sort		
Bubble Sort		

b) Prove that if **[0.8 points]**

$$T(n) = 3n^2 + 4n - 3 \quad is \quad O(n^2)$$

Question 2: **(3.5 points)**

a) **3-way-Merge Sort**: Suppose that instead of dividing in half at each step of Merge Sort, you divide into thirds, sort each third, and finally combine all of them using a three-way merge subroutine. What is the overall asymptotic running time of this algorithm? (Hint: Note that the merge step can still be implemented in O (n) time.) Find the **Recurrence (Running Time)** and the solution using **Recurrence Tree** or **Iterative method.** **[2.5 points]**

b) Use the substitution method to show that the following recurrence. **[1 points]**

$$T(n) = \sqrt{n}\, T\left(\sqrt{n}\right) + n$$

has a possible solution $T(n) = O(nloglogn)$

Question 3: Solve the following recurrences using **Master's Method.** **(3 points)**

a) $\boldsymbol{T(n) = 4T(n/4) + 4n^2}$
b) $\boldsymbol{T(n) = T(n/2) + 8}$
c) $\boldsymbol{T(n) = 16T(n/2) + 16n}$
d) $\boldsymbol{T(n) = 2T(n/2) + 2n\,log\,n}$

Question 4: Consider two n-digit numbers *a* and *b* which are represented as arrays of digits e.g. a = {4,5,6} and b = {9,5,6}, Design an O(n) expected time algorithm that can add two n-digit numbers *a* and *b*. Assume *a* and *b* are already defined. **(2 points)**

BEST OF LUCK!