National University of Computer & Emerging Sciences, Karachi

Fall-2020 Department of Computer Science

Mid Term-1

20th October 2020, 10:30 AM – 11:30 AM

Course Code: CS302	**Course Name:** Design and Analysis of Algorithm
Instructor Name / Names: Dr. Muhammad Atif Tahir, Dr. Fahad Sherwani, Zeshan Khan, Waqas Sheikh, Sohail Afzal	
Student Roll No:	**Section:**

Instructions:

- Return the question paper.
- Read each question completely before answering it. There are **6 questions** on **3 pages.**
- 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: 12.5**

Question # 1 **[0.5*4 = 2 marks]**

Are these following statements True or False? Prove your answer by computing the values of n_0, c_1, c_2 or by contradiction. [Θ is Theta and Ω is Omega]

$a)\ 4^3n + 4^2n + 4^1n = \Theta(n^2)$

$b)\ 2^n + n^2 = \Omega(2^n)$

$c)\ 4^{\log_2 n} + n = \Theta(2^n)$

$d)\ \omega(f(n)) + o(f(n)) = \Omega(f(n))$

Question # 2 **[0.5*4 = 2 marks]**

Solve the following recurrences using **Master's Method.** Give argument, if the recurrence cannot be solved using Master's Method. [See appendix for Master's method 4th case if required]

$a)\ T(n) = 2T\left(\frac{n}{2}\right) + nlogn$

$b)\ T(n) = 4T\left(\frac{n}{2}\right) + 4$

$c)\ T(n) = 5T\left(\frac{n}{1}\right) + n^2$

$d)\ T(n) = 2T\left(\frac{n}{2}\right) + n * 2^3$

Question # 3 **[1.5 + 1 = 2.5 marks]**

Compute the time complexity of the following recurrence relations by using **Iterative Substitution Method** or **Recurrence-Tree Method**. [See appendix for formulas if required]

$\boldsymbol{a}) \; \boldsymbol{T(n) = 3T\left(\frac{n}{3}\right) + n^3}$ **, Assume T(1) = 1**

$\boldsymbol{b}) \; \boldsymbol{T(n) = 4T\left(\frac{n}{4}\right) + n^2}$ **, Assume T(1) = 1**

Question # 4 **[1 mark]**

Consider the given recurrence relation. You need to apply **Substitution Guess and Test method** by assuming given guesses to find correct one.

$$\boldsymbol{T(n) = 3T\left(\frac{n}{2}\right) + n^2}$$

Guess 1 : $\boldsymbol{T(n) = O(n)}$ **, Guess 2 :** $\boldsymbol{T(n) = O(n^2)}$

Question # 5 **[1.5 marks]**

Consider below algorithm which solves the following :

"Find the middle of the linked list while only going the linked once"

```
GetMiddle (List l){
  pSlow = pFast = l;
  while ((pFast->next)&&(pFast->next->next)){
    pFast = pFast->next->next
    pSlow = pSlow->next
  }
  return pSlow
}
```

Invariant Property: At the start of the i-th iteration of the while loop, "pSlow" points to the i-th element in the list and "pFast" points to the 2i-th element.

Prove given invariant property for above pseudocode.

Question # 6 **[1 + 0.5 + 0.5 + 1.5 = 3.5 marks]**

Let $A[1 \ldots\ldots n]$ be an array of n distinct numbers. If $i < j$ and $A[i] > A[j]$ then the pair (i, j) is called an **inversion** of A.

a) List the five inversions of the array $\langle 2, 3, 8, 6, 1\rangle$. *(Recall that inversions are specified by indices rather than array values)*

b) What array with elements from the set $\{1, 2, \ldots, n\}$ has the most inversions? How many does it have?

c) What is the relationship between the running time of insertion sort and the number of inversions in the input array? Justify your answer.

d) Give an algorithm *(you may write algorithm in plain text)* that determines the number of inversions in any permutation on n elements in $\Theta(nlogn)$ worst-case time. (Hint : Modify merge sort)

Appendix

Masters Theorem 4th Case

$$\textit{If } f(n) \in \Theta(n^{\log_b a} \log^k n) \textit{ for some } k \geq 0 \textit{ then}$$

$$T(n) \in \Theta(n^{\log_b a} \log^{k+1} n)$$

Mathematical Properties :

$$\sum_{n=1}^{\infty} a_1(r)^{n-1} = \frac{a_1}{1-r}, \quad \sum_{i=0}^{k-1} \mathbf{a}^{\mathbf{i}} = \frac{\mathbf{1} - \mathbf{a}^{\mathbf{i}}}{\mathbf{1} - \mathbf{a}}$$

BEST OF LUCK