National University of Computer & Emerging Sciences, Karachi
School of Computing
Mid-1 Examination, Fall 2023
26th September, 2023, 10:00 am – 11:00 am

Course Code: CS-2009	**Course Name: Design and Analysis of Algorithm**
Instructors: Dr. Muhammad Atif Tahir, Dr. Fahad Sherwani, Dr. Farrukh Salim, Dr. Nasir Uddin, Mr. Faisal Ahmed, Mr. Sandesh Kumar, Ms. Anaum Hamid, Mr. Syed Faisal Ali, and Mr. Minhal Raza	
Student Roll No:	**Section:** BCS-5A, B, C, D, E, F, G, H, J, K, BSE-5A, B, BAI-5A, B, BCY-5A, B

Instructions:

- Return the question paper
- Read each question completely before answering it. There are **5 questions** on **2 pages**
- In case of any ambiguity, you may make assumptions. But your assumption should not contradict any statement in the question paper

Time: 60 Minutes **Max Marks:** 12.5

Question#1 **[2 Marks, CLO-1]**

Imagine a scenario where we have two different computers, Computer A and Computer B, both tasked with sorting a large dataset of one million records. Computer A has a processing speed of 500 million instructions per second (500 MIPS), while Computer B operates at an impressive 2 billion instructions per second (2 GIPS).

Now, let's add an element of friendly competition to the mix. Computer A and Computer B are participating in a sorting challenge. Computer A decides to use Bubble Sort, an algorithm known to require "n^2" instructions for sorting n records. On the other hand, Computer B chooses Heap Sort but uses an implementation with a high-level language and an inefficient compiler, resulting in code that demands "30 x n x log n" instructions for sorting. With this competition set up, let's explore which computer and sorting algorithm combination outperforms the other when sorting this substantial dataset of one million records.

Question#2 **[4 Marks, CLO-1]**

You're consulting for a small computation-intensive investment company, and they have the following type of problem that they want to solve over and over. A typical instance of the problem is the following. They're doing a simulation in which they look at n consecutive days of a given stock, at some point in the past. Let's number the days i = 1, 2, . . ., n; for each day i, they have a price p(i) per share for the stock on that day. (We'll assume for simplicity that the price was fixed during each day.) Suppose during this time period, they wanted to buy 1,000 shares on someday and sell all these shares on some (later) day. They want to know: When should they have bought and when should they have sold in order to have made as much money as possible? (If there was no way to make money during the n days, you should report this
instead.) For example, suppose n = 4, p (1) = 9, p (2) = 1, p (3) = 5, p (4) = 3. Then you should return "buy on 2, sell on 3" (buying on day 2 and selling on day 3 means they would have made $4 per share, the maximum possible for that period). Design

(a) Algorithm to solve a problem using O(n^2) [1 Point]
(b) Algorithm to solve the above problem using O($nlogn$) [1.5 Points]

You can explain designing in simple sentences but must be clear and all scenarios should be covered [1.5 Points]. Use n = 8, p (1) = 2, p (2) =9, p (3) =3, p (4) =8, p (5) = 11, p (6) = 1, p (7) = 6, p (8) = 6 to check whether result is correct from your design.

Question#3 **[1.5 Marks, CLO-2]**

Compute the running time for the following code fragment. Express it in the term of big O notation.

```
int arr[n];
for (int i=0;i<n;i++){
arr[i]=i;
}
for (int i=0;i<n;i++){
int j=n*n;
while(j>0){
arr[i]=arr[i]+j;
j=(int)(j/2);
}
}
```

Question#4

Solve the following recurrence equations.

PART 4A) Solve the following using the master method **[1 Mark, CLO-2]**

$A)\ T(n) = 3T\left(\frac{n}{3}\right) + c(\frac{n}{2})$ **[0.5 Mark]**

$B)\ T(n) = 4T\left(\frac{n}{2}\right) + n^3(\log_2 n)$ **[0.5 Mark]**

PART 4B) Solve the following using Iterative Substitution or Tree Method. **[3 Marks, CLO-2]**

$A)\ T(n) = \begin{cases} 1 & n = 0 \\ 2T(n-1)+1 & n > 0 \end{cases}$ **[1.5 Marks]**

PART 4C) Solve the following using Iterative Substitution Method.

$B)\ T(n) = \begin{cases} 1 & n = 1 \\ \sqrt{2}T\left(\frac{n}{2}\right) + \sqrt{n} & n > 1 \end{cases}$ **[1.5 Marks]**

Proof for O(logn) and for O(1)

Question#5 **[1 Mark, CLO-2]**

For each of the following questions, indicate whether it is T (True) or F (False).

Also, provide justification for your answer using some example e.g. assuming a function (where possible)

1. $5n\ \log_2 n\ +\ 9n\ =\ O(n\ \log_2 n)\ for\ c = 8, and\ n_0 = 2$ **[0.5 Mark]**
2. $3n^2 + 5n + 7 = O(n^2 \log_2 n)$ **[0.5 Mark]**