UNITED INTERNATIONAL UNIVERSITY (UIU)

Department of Computer Science and Engineering (CSE)

MID-TERM EXAMINATION — **SUMMER, 2025**

DURATION: 1 HOUR 30 MINUTES — **FULL MARKS: 30**

CSE 2215: Data Structures and Algorithms I

Answer **all 5 (five)** questions. Figures in the right margin indicate full marks of questions.

1. a) Suppose you want to sort each of the following three arrays in **ascending order** using **Quicksort**. You are considering the **last element** as pivot. Identify the array for which Quick Sort will face the worst case scenario. Show the simulation for **first partition** of **that array** and justify your answer. 3

   i. `[12, 34, 11, 98, 43, 63]`

   ii. `[95, 87, 57, 11, 5, 38]`

   iii. `[83, 75, 49, 36, 31, 25]`

   b) "Merge Sort requires more space than Quick Sort" — is this statement **True** or **False**? Explain briefly. 2

   c) Is it a good idea to use **Counting Sort** to sort an array of n integers in the range $[1, n^3]$? Explain your answer. 2

2. a) You have a 0-indexed array `grid[50][100]`. Every element of this array is a 4 byte integer. The element `grid[17][25]` is located at address 16900 and the element `grid[32][1]` is located at address 22804. 4 (CO2) (PO3)

   Does this array follow **row-major order memory addressing** or **column-major order memory addressing**? Determine with proper justification.

   Note: It is guaranteed that the addresses are correct and the memory addressing order can be determined conclusively.

   b) For each of the following arrays, determine the more efficient algorithm between *Linear Search* and *Binary Search* to find a given target value. 2 (CO1) (PO3)

   i. `[2, 5, 8, 12, 16, 23, 38, 45, 56, 72]`

   ii. `["apple", "banana", "blueberry", "cherry", "grape", "guava", "mango", "pear"]`

   iii. `[42, 7, 19, 85, 13, 27, 66, 34]`

   iv. `[99, 87, 74, 63, 51, 40, 29, 15]`

3. a) Find the **exact time equation** for the following algorithm and represent it using **asymptotic tight bound notation**. 3

```
for(int i = 1; i <= n; i = i * 2)
{
    for(int j = 1; j * j <= n; j++)
    {
        s = s + i * j;
    }
}
```

   b) Assume two algorithms to complete a specific task take $O(n^2)$ and $O(nlogn)$ times, respectively. However, the first algorithm takes $O(nlogn)$ space while the second algorithm takes $O(n^2)$ space. Which algorithm should you use and why? Justify your answer. 2

4. a) Explain a situation where an **array** is **more suitable** than a **linked list**, and justify your answer with one clear reason. 2

   b) Consider a task scheduling system that maintains a list of processes to be executed sequentially. New processes may arrive at any time and must be added to the end of the list, while execution always begins with the first process, and completed processes are removed from the system. 2

   Explain the advantage of maintaining both `head` and `tail` pointers in a **linked list** for efficiently managing such a system.

   c) You are given the `head` of a **sorted** singly linked list and a value `key` that needs to be inserted into the list. Write a pseudocode for a function that inserts the key into its correct position so that the **list remains sorted** after insertion. 3

5. a) Suppose you have implemented a queue using a circular array of size 3, and it already has two elements enqueued, so the array and the queue pointers looks like this: 3

   `array = [12, 15, ], front = 0, rear = 1`

   Execute the following operations on this queue, and after every operation, show the status of the `array`, and the values of `front` and `rear`.

   i. `dequeue()`
   ii. `enqueue(8)`
   iii. `enqueue(17)`
   iv. `dequeue()`
   v. `dequeue()`
   vi. `dequeue()`

   b) Suppose you have implemented a stack using a singly linked list, and you are maintaining both the `head` and `last` pointers of the list. 2
   What should be the point of `push` and `pop` in this implementation for optimal time complexity — `head` or `last`? Explain briefly.