United International University (UIU)

Dept. of Computer Science & Engineering (CSE)

Midterm Exam Total Marks: **30** Spring-2023

Course Code: CSE2217 Course Title: Data Structure and Algorithms II

Time: *1 hour 45 minutes*

Any examinee found adopting unfair means will be expelled from the trimester / program as per UIU disciplinary rules.

There are **Four** questions. **Answer all of them**. Show all the calculations/steps, where applicable. Figures in the right-hand margin indicate full marks.

1

(a) Consider a modified version of the Merge sort algorithm as follows: **[1.5]**

If the array size is less than or equal to 2, then it sorts the array at **constant time**.

Otherwise, it divides the array of size n into 3 subarrays, each with a size of n/3. This division takes **O(n^2)** time. Then the algorithm sorts the subarrays recursively, and then merges their solutions in time **O(nlogn)**. **Write a recurrence relation** for the running-time **T(n)** of this algorithm.

(b) Prove that the divide and conquer method will sort an array in **O(nlogn)** time when n>1. **[2.5]**

(c) Given an array of integers **A = {70, -40, 20, -50, 10, -15, 20}**, find the **Maximum-sum Continuous Subarray** using divide-and-conquer. You must show the **recursion tree** and clearly mention **left, right and crossing sum** for each tree node. **[3]**

2

(a) What is the main difference between **Dynamic Programming** and **Divide-and-Conquer** algorithms? When should we try to solve a problem using **Dynamic Programming**? **[3]**

(b) Suppose you are a motorcycle dealer buying cars wholesale and then selling them at retail price for a profit. You have a budget of **7 lac Tk**. You went to the wholesale market, and you saw the following items on sale: **[3.5]**

Car Name	**Yamaha R15**	**Harley Davidson**	**Apache RTR 160**	**Honda CBR 150R R**
Wholesale price (in lac Tk)	3	6	1	4
Retail price (in lac Tk)	5	14	2	6

You want to get the **maximum profit** from selling all these motorcycles, but your budget restricts you from buying all of them. Find the maximum profit you can obtain by buying some of these motorcycles and then selling them, provided the total wholesale cost of the motorcycles you selected **do not exceed your budget of 7 lac Tk**. *Note that you can only buy one of these motorcycles at a time, so if you purchased a Yamaha R15 from the wholesale market, you cannot purchase it again.* Find the solution to this problem by using **dynamic programming** and creating a **lookup table**.

	(c) Suppose now you have an **infinite supply** of motorcycles in the wholesale market, which means that if you purchased a Yamaha R15 from the wholesale market, you could purchase it again (*if you have the required amount of money*). What **modification should you make** to the algorithm you used in **question (b)** to make it work for an infinite supply of motorcycles? (You do not need to show any lookup tables here)	**[1.5]**

3	**(a)** Calculate the time complexity (Best Case and Worst Case) of the following code snippets
	
	(i)
	`for (int i =1; i<n; i=i*2) {`
	`  p++;`
	`}`
	`for(int j=1; j<p; j=j*2){`
	`  printf("hello");`
	`}`
	
	(ii)
	`for(int i =1; i*i<n; i++){`
	`  printf("hello");`
	`}`
	
	(iii)
	`for(int i =1; i<n; i=i*2){`
	`  for(int j=1; j<i; j++){`
	`  printf("hello");`
	`  }`
	`}`
	
	(b) Given $f(n) = 5n^3 + 6n^2 + 3n + 9$; $g(n) = n^4$; find the values of **c** and $\mathbf{n_0}$ such that when $n > n_0$, **f (n) = O (g (n))**
		[2*3= 6]
		
		[2]

4	**(a)** Find an optimal solution to the **fractional knapsack** instance of n = 4, W = 5, (v1, v2, v3, v4) = (50, 30, 35, 60), and (w1, w2, w3, w4) = (2, 2, 1, 3). **(b)** Suppose we want to encode the symbols {A, B, C, D} in binary. **Is the following a valid Huffman code?** {A: 0; B: 10; C: 110; D: 111} If it is, build the code tree; if not, explain why you can't. **(c)** You are given the arrival and the departure times of eight trains for a railway platform, and each one is in the format: [arrival time, departure time). Only one train can use the platform at a time. Suppose that you have got the following train-use requests for the next day. { [8, 12), [6, 9), [11, 14), [2, 7), [1, 7), [12, 20), [7, 12) , [13, 19) } Find the **maximum number of trains** that can use the platform without any collision by using *earliest departure time*.	**[3]** **[2]** **[2]**