Final Assessment Test(FAT) - Nov/Dec 2024

Programme	**B.Tech.**	Semester	**Fall Semester 2024-25**
Course Code	**BCSE204L**	Faculty Name	**Prof. Ummity Srinivasa Rao**
Course Title	**Design and Analysis of Algorithms**	Slot	**D1+TD1**
		Class Nbr	**CH2024250101206**
Time	**3 hours**	Max. Marks	**100**

General Instructions

- Write only Register Number in the Question Paper where space is provided (right-side at the top) & do not write any other details.
- If any assumptions are required, assume the same and mention those assumptions in the answer script.
- Use of intelligence is highly appreciated.
- Your answer for all the questions should have both the 'design' component and the 'analysis component'
- The 'Design' component should consist: understanding of the problem, logic to develop the pseudocode, illustration, pseudocode.
- The 'Analysis' component should consist: Proof-of-Correctness, Computation of $T(n)$, Time-complexity.

Course Outcomes

1. Apply the mathematical tools to analyze and derive the running time of the algorithms
2. Demonstrate the major algorithm design paradigms.
3. Explain major graph algorithms, string matching and geometric algorithms along with their analysis.
5. Explain the hardness of real-world problems with respect to algorithmic efficiency and learning to cope with it.

Section - I

Answer all Questions (4 × 10 Marks)

*M - Marks

Q.No	Question	*M	CO	BT
01.	Let $L = < B_1, B_2, \ldots B_n >$ be a list of n boolean expressions and let k be a positive integer with $k < n$. The operators in the boolean expressions are $AND(\wedge)$ and $XOR(\oplus)$ only. Write an algorithm that does not employ sorting but uses a Divide-and-Conquer technique to print all the boolean expressions $B_i \in L$ where the number of $XOR(\oplus)$ operations in B_i are greater than k. Illustrate the algorithm with your example and derive its time complexity. **[Rubrics: Logic- 2 marks, Algorithm-4 marks, Time complexity- 2 marks, and Example-2 marks]**	10	2	
02.	Consider the following algorithm and answer the queries given below. **Count_Algo( )** *K =0* *For* $I_1 = 1$ *to* n	10	1	

For $I_2 = 1$ *to* I_1
For $I_3 = 1$ *to* I_2
.....
.....
For $I_m = 1$ *to* I_{m-1}
$K = K+1$

(a) Write the output of **Count_Algo( )** algorithm, when $n = 5$ and $m = 4$ (3 marks)
(b) Describe the functionality of the **Count_Algo( )** algorithm (5 marks)
(c) Compute the time complexity of the algorithm. (2 marks)

10

03. Let $P = \{p_1, p_2, \cdots p_n\}$ be a given set of n points in the two dimensional plane and let $CH(P)$ denote the convex hull of P. We define the convex layers of P recursively as follows:

1. $P_i = P, i = 0;$
2. $P_i = P_{i-1} - CH(P_{i-1}), i \geq 1,$ and $P_{i-1} \neq \emptyset.$

Here, we start with the initial set of given points $P_0 = P$, and obtain the subsequent convex layers of $P_i, i \geq 1$ (denoted by $CH(P_i)$) from P_{i-1} by removing the points belonging to $CH(P_{i-1})$. This process continues until the set P_i is empty for some i. Then, the convex layers are exactly the points $CH(P_{i-1})$ obtained for each value of i. If for some $k > 0, P_k$ becomes empty then we have $k - 1$ convex layers of P. Given the points P as an input, design an algorithm to compute the convex layers of P.

[Rubrics: Logic- 2 marks, Algorithm-4 marks, Time complexity- 2 marks and, Example-2 marks]

10

04. Let < P_1, P_2,..., P_n> be a list of n points on a two-dimensional plane. Design algorithm to sort a sequence <P_1, P_2,..., P_n> of n points according to their polar angles. Your algorithm should take $O(nlogn)$ time and use cross-product to compare angles.

[Rubrics: Logic- 2 marks, Algorithm-4 marks, Time complexity- 2 marks, and Example-2 marks]

Section - II
Answer all Questions (4 × 15 Marks)

*M -
*M C

Q.No | Question | 15

05. **The 2D_String –Matching Problem** is defined as follows. We assume that a text T is an $n \times n$ 2D array and pattern P is also an $m \times m$ 2D array, where $m \leq n$. We say that pattern P occurs in text T if T_I is a sub-matrixof T such that $T_I = P$. Given, a text T and a pattern P. the task is to find the total number of occurrences of P in T. For example, let

$$T=\begin{bmatrix}1 & 2 & 4\\ 1 & 2 & 6\\ 1 & 2 & 7\end{bmatrix} \text{ and } P=\begin{bmatrix}1 & 2\\ 1 & 2\end{bmatrix}.$$

The pattern P occurs in T, 2 times. Use the Rabin-Karp algorithm to design two algorithms that use different functions to solve the ***2D_String_Matching*** problem. Compare the two functions used to analyze and conclude which works better and why. Compute the time complexity of your algorithm and also Illustrate the algorithm with your example.

[Rubrics: Logic- (2+2) marks, Algorithm- (2+2) marks, Time complexity- (0.5 + 0.5) marks, Function Comparison and Analysis - (1+1) marks and Illustration - (2+2) marks]

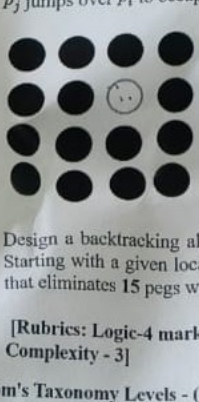
06. Consider the following ***3-PARTITION*** problem. Given integers $<a_1, a_2, \cdots, a_n>$, we want to determine whether it is possible to partition $\{1, 2, \ldots, n\}$ into three disjoint subsets I, J, K such that $\sum_I a_i = \sum_J a_j = \sum_K a_k = \frac{1}{3}\sum_{i=1}^{n} a_i$. As an example, for the given input $< 1, 2, 3, 4, 4, 5, 8 >$, the answer is yes because there is a partition $< 1, 8 >$, $< 4, 5, >$, $< 2, 3, 4 >$. On the other hand, for input $< 2, 2, 3, 5 >$ the answer is

no. Design an algorithm using a dynamic programming approach to solve the *3-PARTITION* problem and compute the time complexity of your algorithm. Illustrate the algorithm with your example.
[Rubrics: Logic- 4 marks, Algorithm-7 marks, Time complexity- 2 marks, and Illustration-2 marks]

07. Let $G = (V, E)$ be a flow network with positive edge capacities. An edge in G is *upper-binding* if increasing its capacity by 1 also increases the value of the maximum flow by 1 in G. Similarly, an edge is *lower-binding* if decreasing its capacity by 1, also decreases the maximum flow value by 1 in G.
(a) Design an algorithm to find all *upper-binding* edges in G, given both G and a maximum flow in G as input.
(b) Design an algorithm to find all *lower-binding* edges in G, given both G and a maximum flow in G as input.
[Rubrics: Logics- (2+2) marks, Algorithms- (3+3) marks, Time complexity- (0.5 + 0.5) marks, and Illustration - (2+2) marks]

08. Consider the following **Puzzle-Peg** problem. The puzzle-peg problem is played on a board with $n \times n$ *board that contain* n^2 *slots.* Initially, all but one of the slots are occupied by pegs. That is, $n^2 - 1$ slots are occupied with pegs and one slot is empty. For example, a board with 16 slots is shown in the figure given below. The occupied slots are represented by dark solid circles and the empty slot is represented as a white hollow circle in the figure. Assuming that the slots are numbered from (1,1) up to (4,4), the empty slot is at (2,3).
A move is legal if,
a. the move is either horizontal or vertical and not diagonal,
b. a peg moves into an empty slot by jumping over exactly one peg that is adjacent to the empty slot. While doing so, the peg that was jumped over is removed from the board.
For e.g., in the figure given below the peg at slot (2,1) can jump over the peg at slot (2,2) and move into the empty slot. This move shifts the empty slot from (2,3) to (2,1) and the peg at slot (2,2) is removed from the board leaving. On the other hand, the peg at slot (4,3) can jump over the peg at slot (3,3) and move into the empty slot shifting the empty slot from (2,3) to (4,3), thereby removing the peg at slot (3,3). Note that in this example no other moves can be made other than the two moves given above. Also, for every move that is made, a new empty slot is created.

For each legal move that is made, a peg p_i is removed from the square if another peg p_j jumps over p_i to occupy an empty slot.



Design a backtracking algorithm for solving the following versions of this puzzle - Starting with a given location of the empty slot, find the shortest sequence of moves that eliminates **15** pegs with no limitations on the final position of the remaining peg.

[Rubrics: Logic-4 marks, Algorithm-5 marks, Example- 3 marks, Time Complexity - 3]

BL-Bloom's Taxonomy Levels - (1.Remembering, 2.Understanding, 3.Applying, 4.Analysing, 5.Eva 6.Creating)