Enrolment No./Seat No_____

GUJARAT TECHNOLOGICAL UNIVERSITY

BE - SEMESTER–V EXAMINATION – SUMMER 2025

Subject Code:3150703 Date:28-05-2025

Subject Name:Analysis and Design of Algorithms

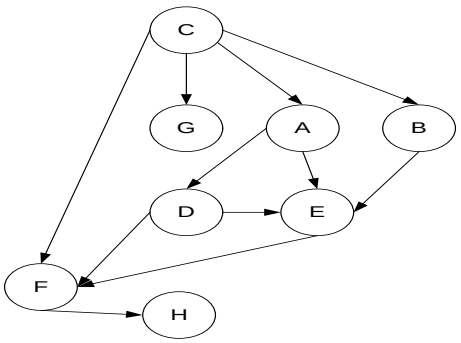
Time:02:30 PM TO 05:00 PM Total Marks:70

Instructions:

1. **Attempt all questions.**
2. **Make suitable assumptions wherever necessary.**
3. **Figures to the right indicate full marks.**
4. **Simple and non-programmable scientific calculators are allowed.**

			MARKS
Q.1	**(a)**	Find Big-oh (O) notation of following functions $f(n)=3n^6 + 2\,n * \lg n + 6n$, $h(n)= 3n^2+ 6n*\lg n+ 2n^{2.5}$	**03**
	(b)	Define Omega and Theta notations with graph.	**04**
	(c)	Write binary search algorithm and analyze it for worst case time complexity. Represent its time complexity using Big-oh (O) notation.	**07**
Q.2	**(a)**	Prove that $(n + 6)^3 = \Theta(n^3)$.	**03**
	(b)	If $P(n) = a_0+ a_1 n + a_2 n^2 + \ldots\ldots + a_m n^m$ then prove that $P(n) = \Omega(n^m)$. Here $a_0, a_1, a_2 \ldots..a_m$ are constants and $a_m>0$.	**04**
	(c)	Solve following recurrence relation using suitable method and express your answer using Big-oh (O) notation. $T(n) = T(n/6) + T(5n/6) + \Theta(n)$	**07**
		OR	
	(c)	Solve following recurrence relations using suitable method and express your answer using Big-oh (O) notation. 1. $T(n) = 2\,T(n/2) + n \log n$ 2. $T(n) = 4\,T(n/4) + n$	**07**
Q.3	**(a)**	Arrange the following growth rates from the lowest to highest asymptotic order: $O(n \log n)$, $O(n^{2n})$, $\Omega(\log n)$, $O(n^{0.5})$, $O(n!)$, $\Omega(2^n)$, $O(n^{0.5} \log n)$	**03**
	(b)	Write the recurrence relation for the quick sort on input instance: 15, 19, 20, 25, 32, 37, 50, 62, 70. Comment on the nature of input i.e. best case, average case or worst case.	**04**
	(c)	Write greedy algorithm for activity selection problem. Give its time complexity. For following intervals, select the activities according to your algorithm. I_1 (1-3), I_2 (0-2), I_3 (3-6), I_4 (2-5), I_5 (5-8), I_6 (3-10), I_7 (7-9).	**07**
		OR	
Q.3	**(a)**	Prove that $\text{Log}(\sqrt{n}) = O(\log n)$.	**03**
	(b)	Write any algorithm (need not to have meaningful) which has a time complexity of O(1) and also includes a loop statement. Also, confirm its time complexity of O(1) using tabular method.	**04**

(c) Write DFS traversal and draw DFS tree corresponds to following graph. Also, write time complexity of DFS algorithm. **07**



Q.4 **(a)** What is Principle of Optimality in dynamic programming? Explain it with example. **03**

(b) What is knapsack problem? Using greedy algorithm find an optimal solution for fractional knapsack instance n=7, M =17, profits p[ ] = {10,5,15,7,6,16,4} and weights w[ ] = {2,3,5,7,1,4,1}. **04**

(c) Find the optimal way of multiplying following matrices using dynamic programming. Also indicate optimal number of multiplications required. A:2 x 3, B: 3 x 5, C:5 x 6, D: 6 x 2, E: 2 x 3 **07**

OR

Q.4 **(a)** What is the Travelling Salesman Problem (TSP)? Why TSP is considered an NP-hard problem? **03**

(b) Illustrate the working of Kruskal's algorithm using greedy technique. Take the suitable graph for illustration. **04**

(c) Find the longest common subsequence for the following two sequences using dynamic programming. Show the complete process and indicate the time complexity. **07**

X = 10101010011

Y = 101001

Q.5 **(a)** Define P and NP problems. Also give example of each type of problem. **03**

(b) Explain the backtracking strategy to solve optimization problem. Explain how it can be applied to find the solution for 4-queen problem. **04**

(c) Given a text and a pattern, write the naive string matching algorithm to find all occurrences of the pattern in the text. Use the text "ABCABCABC" and the pattern "ABC". Also, give time complexity of naive string matching algorithm. **07**

OR

Q.5 **(a)** What is a polynomial reduction? Explain its role in proving NP-completeness. **03**

(b) Give example of any NP complete problem. Explain how to prove that particular problem is NP complete. **04**

(c) Show that Hamiltonian cycle is a NP problem. **07**

****