C4-R4: ADVANCED ALGORITHMS

NOTE:

1. **Answer question 1 and any FOUR from questions 2 to 7.**
2. **Parts of the same question should be answered together and in the same sequence.**

Time: 3 Hours **Total Marks: 100**

1.

a) What are the features of a good computer algorithm? When an algorithm is said to be correct? When an algorithm is said to efficient?

b) Is a Recursive algorithm less efficient compared to an Iterative one? Why or Why not?

c) Why we are generally concerned with Time Complexities than Space Complexities? What do you mean by Polynomial time complexity and Logarithmic time complexity? Which one is higher?

d) Prove/Give counter example: A graph with *n* nodes and more than *n-1* edges must contain cycle.

e) Give an example where the choice of data structure has a bearing effect on the performance of the algorithm. Also give the circumstances where:
   i) It is advantageous to use array instead of linked list
   ii) It is advantageous to use linked list instead of array.

f) Analyze the time complexity for Tower of Hanoi problem.

g) Represent the Big-Oh, Omega & Theta notations using graphs.

(7x4)

2.

a) Derive the recurrence relation for quick sort in best case and find the complexity.

b) Explain the methods to compare the algorithms. What do you mean by worst-case, average-case and best-case analysis of an algorithm?

c) Solve the following recurrence equations.
   1. $T(n) = T(n-1) +2T(n-2) – 2T(n-3)$
   2. $T(n) = 2T(n/2) + n \log n$

(6+6+6)

3.

a) Explain Dijkstra's Algorithm, Trace it using an example.

b) Write a recursive algorithm to find Fibonacci numbers. Write recurrence equation for it. Solve the recurrence equation and find out the complexity.

c) Write the Kruskal's algorithm for finding the minimum spanning tree.

(6+6+6)

4.

a) Prove that *Greedy Algorithms* does not always give optimal solution. What are the general characteristics of *Greedy Algorithms*? Also compare *Greedy Algorithms* with *Dynamic Programming* and *Divide and Conquer* methods to find out major difference between them.

b) Justify the general statement that "if a problem can be split using Divide and Conquer strategy in almost equal portions at each stage, then it is a good candidate for recursive implementation, but if it cannot be easily be so divided in equal portions, then it better be implemented iteratively". Explain with an example.

c) Given a finite set of distinct coin types, say 50,20,10,5,2,1, and an integer constant C. Each type is available in unlimited quantity. Write an algorithm to find the exact change with minimum number of coins? Which strategy will you use?

(6+6+6)

5.

a) Write short note on Approximation algorithms.

b) Depth First Search (DFS) algorithm can be used to find the connected components of an undirected graph. Justify the statement with reason.

c) Explain Radix sort with an example.

(6+6+6)

6.

a) Give the algorithm with example to solve 0/1 Knapsack Problem using Dynamic Programming.

b) What are the basic ideas behind string matching? Which are the two popular algorithms for string matching? Explain and analyze any one in brief.

c) Define the class **P** and class **NP**. Is there any NP-Hard problem, which is also NP? If yes, give an example, if no, why?

(6+6+6)

7.

a) State Principle of *optimality*. Principle of *optimality* does not hold for all problems whose solutions can be viewed as a sequence of decisions. Find out such problem and show that it fails.

b) Discuss the role of string matching in sequence alignment and its applicability for two sequence alignments and multiple sequence alignment.

c) What do you mean by *Amortized Analysis*? How does it differ from Asymptotic *Analysis and Average Case Analysis*? Explain their usefulness in brief.

(6+6+6)