102

Code No: F-13918/N/BL/AICTE

FACULTY OF ENGINEERING

B.E. (I.T) VI –Semester (AICTE) (Main & Backlog) (New) Examinations, August/September 2024

Time: 3 Hours | Subject: Design and Analysis of Algorithm | Max. Marks: 70

Note: (i) First question is compulsory and answer any four questions from the remaining six questions. Each question carries 14 Marks.
(ii) Answer to each question must be written at one place only and in the same order as they occur in the question paper.
(iii) Missing data, if any, may be suitably assumed.

1. a) What is time complexity? How time complexity is measured?
   b) List out problems solved by Greedy method.
   c) Differentiate greedy method with dynamic programming.
   d) Write control abstraction of divide and conquer.
   e) What are constraints in backtracking? Give an example of explicit constraint?
   f) State and explain COOK's Theorem.
   g) Explain what E-node state space tree is.

2. a) How the performance can be analyzed? Explain with the example.
   b) Explain the following set operations (i) weighted union (ii) Collapses find.

3. a) Find an optimal solution to the knapsack instance n=7 objects and the capacity of knapsack m=15. The profits and weights of the objects are (P1,P2,P3, P4, P5, P6, P7)= (10, 5,15,7,6,18,3) (W1,W2,W3,W4,W5,W6,W7)= (2,3,5,7,1,4,1)
   b) What is a Minimum Cost Spanning tree? Explain Kruskal's Minimum cost spanning tree algorithm with suitable example.

4. a) Draw an Optimal Binary Search Tree for n=4 identifiers {a1,a2,a3,a4) = (do,if,read, while) P(1:4)=(3,3,1,1) and Q(0:4)=(2,3,1,1,1).
   b) Draw all possible binary search trees for the identifier set (do, if, stop).

5. a) Write the Control Abstraction of Backtracking method.
   b) What is a Hamiltonian Cycle? Explain how to find Hamiltonian path and cycle using backtracking algorithm.

6. a) Explain Node Covering problem.
   b) Explain NP-HARD generation problems with examples.

7. a) Derive the time complexity of Merge sort algorithm for all cases.
   b) Distinguish between Dynamic Programming and Greedy method.

**