Department of Computer Science and Engineering (CSM & CSD)

AS22-66PC03- Design and Analysis of Algorithm

Question Bank

St. Peter's Engineering College (Autonomous) Dullapally (P), Medchal, Hyderabad – 500100. QUESTION BANK						Dept.	:	CSM,CSD
						Academic Year 2025-26		
Subject Code	**:**	**AS22-66PC03**	**Subject**	**:**	**Design and Analysis of Algorithm**			
Class/Section	**:**	**B. Tech.**	**Year**	**:**	**III**	**Semester**	**:**	**I**

BLOOMS LEVEL					
Remember	**L1**	**Understand**	**L2**	**Apply**	**L3**
Analyze	**L4**	**Evaluate**	**L5**	**Create**	**L6**

*****

UNIT-I

Q. No	Question (s)	Marks	BL	CO
PART-A				
1	What is an algorithm?	**1M**	BL1	C311.1, C311.2
2	Define time complexity.	**1M**	BL2	C311.1, C311.2
3	Define space complexity.	**1M**	BL1	C311.1, C311.2
4	What is Big-O notation used for?	**1M**	BL2	C311.1, C311.2
5	Give the worst-case time complexity of Merge Sort.	**1M**	BL3	C311.1, C311.2
6	Name the divide and conquer strategy used in Quick Sort.	**1M**	BL3	C311.1, C311.2
7	What is Theta (θ) notation?	**1M**	BL3	C311.1, C311.2
8	State the best-case time complexity of Binary Search.	**1M**	BL2	C311.1, C311.2
9	Mention any one application of Strassen's matrix multiplication.	**1M**	BL1	C311.1, C311.2
10	Define little-oh (o) notation.	**1M**	BL2	C311.1, C311.2
PART- B				
1	Sort the records with the following index values in the ascending order using quick sort algorithm 30, 20, 10,50, 60, 40.	**5M**	BL2	C311.1, C311.2

2	Write the procedure for Strassen's Matrix multiplication?	**5M**	BL4	C311.1, C311.2
3	Using Merge sort algorithm sort the given list:7,5,2,4,1,6,3,0.	**5M**	BL3	C311.1, C311.2
4	Describe about Asymptotic Notations.	**5M**	BL3	C311.1, C311.2
5	Explain about Divide and Conquer technique in algorithm design and write its applications?	**5M**	BL2	C311.1, C311.2
6	**Explain the differences between Big-O, Big-Ω, and Big-Θ notations with suitable examples.**	**5M**	BL3	C311.1, C311.2
7	**Illustrate how Binary Search follows the divide and conquer paradigm.**	**5M**	BL5	C311.1, C311.2
8	**Write and explain the recurrence relation for Merge Sort. Derive its time complexity.**	**5M**	BL4	C311.1, C311.2
9	**Analyze the best, average, and worst-case time complexities of Quick Sort.**	**5M**	BL2	C311.1, C311.2
10	**Compare and contrast Quick Sort and Merge Sort in terms of algorithm design, time complexity, and space usage.**	**5M**	BL4	C311.1, C311.2
11	Sort the records with the following index values in the ascending order using Quick Sort algorithm 2, 3, 8, 5, 4, 7, 6, 9, 1.	**10M**	BL5	C311.1, C311.2
12	Write an algorithm for Merge Sort and the complexity of the algorithm with an example?	**10M**	BL5	C311.1, C311.2
13	Explain Strassen's Matrix multiplication with one example? What is the time complexity of this algorithm?	**10M**	BL4	C311.1, C311.2
14	**Design and implement Merge Sort using the divide and conquer approach. Explain its time and space complexity with the help of a recurrence relation and recursion tree.**	**10M**	BL3	C311.1, C311.2
15	**Write the algorithm for Quick Sort. Analyze its performance in the best, average, and worst cases. Include pivot choice impact in your discussion.**	**10M**	BL3	C311.1, C311.2
16	**Critically evaluate the differences between all four asymptotic notations: Big-O, Big-Ω, Big-Θ, and little-o, using graphs and examples.**	**10M**	BL4	C311.1, C311.2
17	**Explain Strassen's matrix multiplication algorithm in detail. Compare its computational complexity with standard matrix multiplication and analyze the space-time tradeoffs.**	**10M**	BL3	C311.1, C311.2
18	Apply Merge Sort on the array [38, 27, 43, 3, 9, 82, 10] and show all intermediate steps including the merging process.	**10M**	BL4	C311.1, C311.2
19	Explain the divide and conquer approach used in Binary Search.	**10M**	BL3	C311.1, C311.2
20	Write pseudocode for Binary Search (both recursive and iterative versions).	**10M**	BL2	C311.1, C311.2

UNIT-II

Q. No	Question (s)	Marks	BL	CO
PART-A				
1	What is a disjoint set?	**1M**	BL2	C311.2, C311.3
2	Name two operations performed on disjoint sets.	**1M**	BL2	C311.2, C311.3
3	Define the 'find' operation in a disjoint set.	**1M**	BL2	C311.2, C311.3
4	State one application of union-find.	**1M**	BL1	C311.2, C311.3
5	What is backtracking?	**1M**	BL2	C311.2, C311.3
6	Which data structure is commonly used in the union-find algorithm?	**1M**	BL2	C311.2, C311.3
7	What is the base condition for the N-Queens problem?	**1M**	BL2	C311.2, C311.3
8	List any one application of backtracking.	**1M**	BL3	C311.2, C311.3
9	Which graph problem can be solved using backtracking?	**1M**	BL2	C311.2, C311.3
10	What is the output of find (5) in a set where 5 → 3 → 1?	**1M**	BL3	C311.2, C311.3
PART- B				
1	Explain the union and find operations with an example.	**5M**	BL5	C311.2, C311.3
2	Write pseudocode for the 'find' operation using path compression.	**5M**	BL2	C311.2, C311.3
3	Explain the working of 4-Queen problem with a simple example.	**5M**	BL4	C311.2, C311.3
4	Describe the steps of solving the N-Queens problem using backtracking.	**5M**	BL3	C311.2, C311.3
5	Write the backtracking algorithm for the Subset Sum problem.	**5M**	BL4	C311.2, C311.3
6	Explain the working of 8-Queen problem with a simple example.	**5M**	BL3	C311.2, C311.3
7	Explain how disjoint sets can be used to detect cycles in an undirected graph.	**5M**	BL4	C311.2, C311.3
8	Analyze the time complexity of union-find with path compression.	**5M**	BL5	C311.2, C311.3
9	Discuss the recursive nature of the backtracking method.	**5M**	BL4	C311.2, C311.3
10	Trace the solution tree of N = 4 in the N-Queens problem.	**5M**	BL5	C311.2, C311.3

11	Design and implement a disjoint set data structure with union-by-rank and path compression.	**10M**	BL6	C311.2, C311.3
12	Solve the Graph Coloring problem using backtracking for a given graph and explain your steps.	**10M**	BL5	C311.2, C311.3
13	Implement the Sum of Subsets problem using backtracking and analyze the time complexity.	**10M**	Bl5	C311.2, C311.3
14	Evaluate the advantages and limitations of backtracking in solving constraint satisfaction problems.	**10M**	BL4	C311.2, C311.3
15	Implement 4 Queen and 8 Queen problem using backtracking.	**10M**	BL3	C311.2, C311.3
16	Given a constraint-based puzzle (like Sudoku), frame it as a backtracking problem and outline your solution.	**10M**	BL3	C311.2, C311.3
17	Compare and contrast Disjoint Set Union-Find with other dynamic connectivity algorithms.	**10M**	BL4	C311.2, C311.3
18	Apply the backtracking algorithm to solve the following instance of the sum of subsets problem S={1,3,4,5} and m=8.	**10M**	BL3	C311.2, C311.3
19	Create a program to solve the N-Queens problem for N=8 using both recursion and backtracking. Show the full output.	**10M**	BL3	C311.2, C311.3
20	Critically analyze the difference between brute-force and backtracking strategies with suitable examples.	**10M**	BL4	C311.2, C311.3

UNIT-III

Q. No	Question (s)	Marks	BL	CO
PART-A				
1	Define dynamic programming.	**1M**	BL1	C311.1, C311.3
2	What is the principle of optimality?	**1M**	BL2	C311.1, C311.3
3	What is the time complexity of the 0/1 knapsack problem using dynamic programming?	**1M**	BL1	C311.1, C311.3
4	Name any two applications of dynamic programming.	**1M**	BL3	C311.1, C311.3
5	State one advantage of dynamic programming over divide and conquer.	**1M**	BL1	C311.1, C311.3
6	In Optimal Binary Search Tree, what does w(i,j) represent?	**1M**	BL3	C311.1, C311.3
7	What is the goal of the Travelling Salesperson Problem (TSP)?	**1M**	BL2	C311.1, C311.3

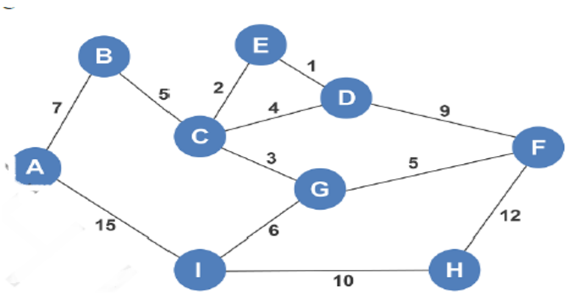
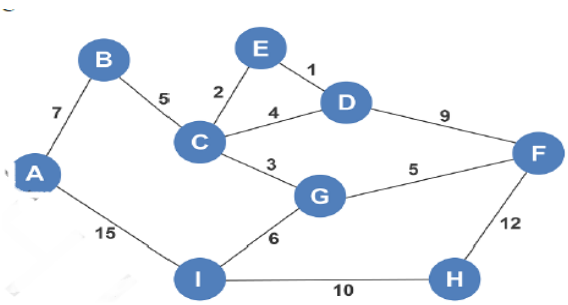
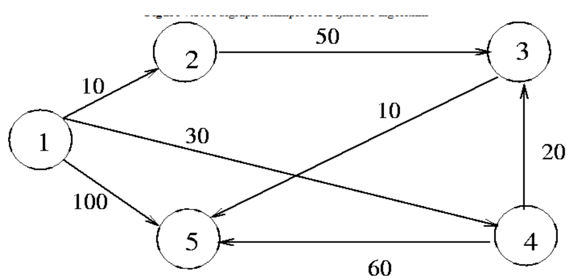
8	What type of graph is used in the All-Pairs Shortest Path problem?	**1M**	BL2	C311.1, C311.3
9	In the 0/1 knapsack problem, can items be divided?	**1M**	BL2	C311.1, C311.3
10	What is meant by reliability in system design?	**1M**	BL3	C311.1, C311.3
PART- B				
1	Explain the general method of dynamic programming with an example.	**5M**	BL2	C311.1, C311.3
2	Given n = 4, weights = {2, 3, 4, 5}, values = {3, 4, 5, 6} and capacity = 5, solve the 0/1 Knapsack problem using dynamic programming.	**5M**	BL3	C311.1, C311.3
3	Explain the construction of Optimal Binary Search Trees with a small example.	**5M**	BL3	C311.1, C311.3
4	Describe the Floyd-Warshall algorithm and its working for the All-Pairs Shortest Path problem.	**5M**	BL3	C311.1, C311.3
5	Explain how dynamic programming is used to solve the Travelling Salesperson Problem.	**5M**	Bl4	C311.1, C311.3
6	Compare the greedy method and dynamic programming with a relevant example.	**5M**	BL4	C311.1, C311.3
7	Analyze the time and space complexity of the 0/1 Knapsack dynamic programming solution.	**5M**	BL4	C311.1, C311.3
8	Identify overlapping subproblems in the computation of Fibonacci numbers.	**5M**	BL3	C311.1, C311.3
9	Construct the cost matrix and explain the DP approach to solve the All-Pairs Shortest Path problem.	**5M**	BL3	C311.1, C311.3
10	Analyze how system reliability is computed using dynamic programming in reliability design.	**5M**	BL2	C311.1, C311.3
11	Design a dynamic programming algorithm to solve the 0/1 Knapsack problem and explain the steps with time complexity analysis.	**10M**	BL3	C311.1, C311.3
12	Discuss the dynamic programming approach to construct Optimal Binary Search Trees. Provide a full worked-out example.	**10M**	BL2	C311.1, C311.3
13	Write and explain the dynamic programming solution for the All-Pairs Shortest Path problem using the Floyd-Warshall algorithm.	**10M**	BL3	C311.1, C311.3
14	Develop a dynamic programming solution to the Travelling Salesperson Problem (TSP). Analyze its time complexity.	**10M**	BL3	C311.1, C311.3
15	Describe a dynamic programming approach for system reliability design. How does it differ from other approaches?	**10M**	BL3	C311.1, C311.3

16	Using OBST compute w(i,j),r(i,j),c(i,j),$0<i<=j<=4$ for the identifier set (a1,a2,a3,a4) = (end, goto, print, stop) with p(1) =1/20, p(2)=1/5, p(3)=1/10, p(4)=1/20, q(0)=1/5, q(1)=1/10, q(2)=1/5, q(3)=1/20, q(4)=1/20. Using r(i,j) construct the optimal binary search tree	**10M**	BL2	C311.1, C311.3
17	Create a comparative analysis of dynamic programming applications in knapsack, TSP, and OBST.	**10M**	BL3	C311.1, C311.3
18	Explain about 0/1 Knapsack problem in Dynamic programming and hence solve the following 0/1 knapsack problem for n=4, (W1, W2, W3,W4) = (2, 4, 6,9), (P1, P2, P3,P4) =(10,10,12,18) and m=15.	**10M**	BL3	C311.1, C311.3
19	Explain about 0/1 Knapsack problem in Dynamic programming and hence solve the following 0/1 knapsack problem for n=3, (W1, W2, W3) = (2, 3,4), (P1, P2, P3) =(1,2,5) and m=6.	**10M**	BL3	C311.1, C311.3
20	Using OBST compute w(i,j),r(i,j),c(i,j),$0<i<=j<=4$ for the identifier set (a1,a2,a3,a4)=(end, goto, print, stop) with p(1)=3, p(2)=3, p(3)=1, p(4)=1, q(0)=2, q(1)=3, q(2)=1, q(3)=1, q(4)=1. Using r(i,j) construct the optimal binary search tree	**10M**	BL4	C311.1, C311.3

UNIT-IV

Q. No	Question (s)	Marks	BL	CO
PART-A				
1	Define the greedy method.	1M	BL2	C311.4, C311.5
2	What is the main characteristic of a greedy algorithm?	1M	BL2	C311.4, C311.5
3	Does the greedy approach guarantee an optimal solution for all problems?	1M	BL2	C311.4, C311.5
4	What is the objective of the job sequencing with deadlines problem?	1M	BL2	C311.4, C311.5
5	In which scenario does greedy method fail for the knapsack problem?	1M	BL2	C311.4, C311.5
6	What is the difference between 0/1 knapsack and fractional knapsack?	1M	BL2	C311.4, C311.5
7	Name any one greedy algorithm used for finding Minimum Cost Spanning Trees.	**1M**	BL3	C311.4, C311.5
8	What is the time complexity of Kruskal’s algorithm?	**1M**	BL2	C311.4, C311.5
9	Which data structure is typically used in Prim’s algorithm?	**1M**	BL2	C311.4, C311.5
10	What is the output of Dijkstra’s algorithm?	**1M**	BL1	C311.4, C311.5

PART- B				
1	Explain the general method of the greedy approach with a real-world example.	**5M**	BL3	C311.4, C311.5
2	Solve the fractional knapsack problem using greedy strategy for the following: n = 3, weights = {10, 20, 30}, values = {60, 100, 120}, capacity = 50	**5M**	BL3	C311.4, C311.5
3	Explain Kruskal's algorithms with example.	**5M**	BL4	C311.4, C311.5
4	Describe the job sequencing with deadlines problem and its greedy solution.	**5M**	BL3	C311.4, C311.5
5	Compare and contrast Prim's and Kruskal's algorithms.	**5M**	BL5	C311.4, C311.5
6	Apply Dijkstra's algorithm on a given graph and compute the shortest paths.	**5M**	BL4	C311.4, C311.5
7	What conditions must a problem satisfy to be solved using a greedy approach?	**5M**	BL3	C311.4, C311.5
8	Analyze why greedy algorithms give optimal results for fractional knapsack but not 0/1 knapsack.	**5M**	BL4	C311.4, C311.5
9	Explain how greedy method differs from dynamic programming with examples.	**5M**	BL3	C311.4, C311.5
10	Illustrate how the minimum cost spanning tree is obtained using Kruskal's algorithm.	**5M**	BL3	C311.4, C311.5
11	Design and explain a greedy algorithm to solve the fractional knapsack problem. Include step-by-step execution and complexity analysis.	**10M**	BL3	C311.4, C311.5
12	With an example, explain the greedy strategy for job sequencing with deadlines. Show how the schedule is created and compute the profit. deadline[] = [2, 1, 2, 1, 1], profit[] = [100, 19, 27, 25, 15]	**10M**	BL5	C311.4, C311.5
13	Explain both Prim’s and Kruskal’s algorithms for minimum cost spanning tree. Provide pseudo-code, example graph, and analyze complexity.	**10M**	BL4	C311.4, C311.5
14	Using Dijkstra’s algorithm, compute the shortest paths from a given source vertex. Include a table to show stepwise progress.	**10M**	BL4	C311.4, C311.5
15	Evaluate the limitations of greedy algorithms. Compare them with dynamic programming using the 0/1 knapsack.	**10M**	BL3	C311.4, C311.5
16	Find the optimal solution of the Knapsack problem using Greedy method where n=4, m=15, (p1-p4) = (10,10,12,18) And (w1-w4) = (2,4,6,9)	**10M**	BL4	C311.4, C311.5

17	Solve the below Job sequencing with deadline problem usingGreedy method n=4, Profits (p1, p2, p3,p4)=(100,10,15,27) deadlines(d1,d2,d3,d4)=(2,1,2,1)	**10M**	BL5	C311.4, C311.5
18	Explain Krushkal Algoritham and solve the graph using it .  	**10M**	BL3	C311.4, C311.5
19	Explain Prims Algoritham and solve the graph using it.  	**10M**	BL4	C311.4, C311.5
20	Explain about single source shortest path problem in Greedymethod with a simple example.  	**10M**	BL5	C311.4, C311.5

UNIT-V

Q. No.	Question (s)	Marks	BL	CO
PART-A				
1	What is the main idea of the branch and bound technique?	**1M**	BL2	C311.6
2	Name two strategies used in branch and bound.	**1M**	BL2	C311.6
3	What is the role of bounding in branch and bound?	**1M**	BL1	C311.6
4	What is the difference between LC and FIFO branch and bound?	**1M**	BL2	C311.6

5	Name one problem that can be solved using branch and bound.	**1M**	BL2	C311.6
6	Define NP-Complete.	**1M**	BL3	C311.6
7	Define NP-Hard.	**1M**	BL2	C311.6
8	What does NP stand for?	**1M**	BL3	C311.6
9	What is the significance of Cook's Theorem?	**1M**	BL3	C311.6
10	Give an example of an NP-Complete problem.	**1M**	BL2	C311.6
PART- B				
1	Explain the general method of solving problems using branch and bound.	**5M**	BL3	C311.6
2	Describe the LC (Least Cost) branch and bound solution for the 0/1 Knapsack problem.	**5M**	BL4	C311.6
3	Explain how FIFO branch and bound is used to solve the 0/1 Knapsack problem.	**5M**	BL3	C311.6
4	Describe the application of branch and bound in solving the Travelling Salesperson Problem.	**5M**	BL5	C311.6
5	Compare branch and bound with backtracking. In what way is it more efficient?	**5M**	BL4	C311.6
6	Differentiate between NP, NP-Hard, and NP-Complete with examples.	**5M**	BL4	C311.6
7	Analyze the significance of non-deterministic algorithms in NP problems.	**5M**	BL3	C311.6
8	Explain Cook's Theorem in brief and its role in computational complexity.	**5M**	BL4	C311.6
9	Discuss the characteristics that make a problem NP-Complete.	**5M**	BL4	C311.6
10	Why are NP-Complete problems considered intractable?	**5M**	BL5	C311.6
11	Design and explain a branch and bound algorithm for the 0/1 Knapsack problem using LC strategy. Trace with an example.	**10M**	BL3	C311.6
12	Describe the working of FIFO branch and bound for the 0/1 Knapsack problem. Include tree structure and bound calculations.	**10M**	BL4	C311.6
13	Apply branch and bound to solve a given Travelling Salesperson Problem instance. Show matrix reduction and bounding steps.	**10M**	BL3	C311.6
14	Evaluate the efficiency of branch and bound in solving combinatorial optimization problems compared to greedy and dynamic programming methods.	**10M**	BL5	C311.6
15	Create a state-space tree for a given knapsack instance and explain the decisions made at each node in the LC branch and bound approach.	**10M**	BL5	C311.6
16	Define and explain the classes P, NP, NP-Complete, and NP-Hard with suitable examples.	**10M**	BL3	C311.6
17	Describe in detail the significance of Cook's Theorem. How did it influence computational theory?	**10M**	BL4	C311.6

18	Evaluate different methods for showing that a problem is NP-Complete.Prove that 3-SAT is NP-Complete using polynomial-time reduction concepts.	**10M**	BL3	C311.6
19	Discuss the practical implications of NP-Complete problems in real-world computing. Suggest how approximation or heuristic methods are used as alternatives.	**10M**	BL4	C311.6
20	Design and explain a branch and bound algorithm for the 0/1 Knapsack problem using LCBB strategy for the knapsack instance n=4, (p1,p2,p3,p4)=(10,10,12,18), (W1,W2,W3,W4)=(2,4,6,9), m=15.	**10M**	BL4	C311.6