Reg. No. : □□□□□□□□□□

Question Paper Code: 44204

B.E. / B.Tech. DEGREE EXAMINATION, NOV 2018

Fourth Semester

Computer Science and Engineering

14UCS404 - DESIGN AND ANALYSIS OF ALGORITHMS

(Regulation 2014)

Duration: Three hours   Maximum: 100 Marks

Answer ALL Questions

PART A - (10 x 1 = 10 Marks)

1. Two main measures for the efficiency of an algorithm are

   (a) Processor and memory   (b) Complexity and capacity
   (c) Time and space   (d) Data and space

2. For defining the best time complexity, let *f (n)* = *log n* and *g (n)* = √*n*, _____

   (a) *f (n) ϵ Ω(g(n)),* but *g(n) Ɇ Ω (f(n))*   (b) *f (n) Ɇ Ω(g(n)),* but *g(n) ϵ Ω (f(n))*
   (c) *f (n) Ɇ Ω(g(n)),* and *g(n) Ɇ Ω (f(n))*   (d) *f (n) ϵ Ω(g(n)),* and *g(n) ϵ Ω (f(n))*

3. The running time of quick sort depends heavily on the selection of

   (a) No of inputs   (b) Size of elements
   (c) Arrangement of elements in array   (d) Pivot element

4. For the improvement of efficiency of quick sort the pivot can be

   (a) the first element   (b) the mean element
   (c) the last element   (d) None of these

5. The OBST algorithm in worst case takes _______ time if all c(i, j)'s and r(i, j)'s are calculated.

   (a) $O(\log n)$ (b) $O(n^4)$ (c) $O(n^3)$ (d) $O(n \log n)$

6. Prim's algorithm is based on _____________ method

(a) Divide and conquer method (b) Greedy method
(c) Dynamic programming (d) Branch and bound

7. A linear programming problem which does not have an optimal solution is called

(a) Unbounded (b) Infeasible
(c) Feasible (d) Non-optimal

8. Fold-Fulkerson can find a maximum matching in a bipartite graph in __________ time

(a) *O(mn)* (b) *O(m)* (c) *O(n)* (d) *O(m+n)*

9. A____________ is a round trip path along n edges of G that visits every vertex once and returns to its starting position.

(a) MST (b) TSP
(c) Multistage Graph (d) Hamiltonian cycle

10. The Knapsack problem where the objective function is to minimize the profit is _______

(a) Greedy (b) Dynamic 0 / 1
(c) Branch and Bound 0/1 (d) Backtracking

PART - B (5 x 2 = 10 Marks)

11. Define big omega notations.

12. Analyze the computing time for Binary search.

13. Differentiate dynamic programming and greedy technique.

14. Show the Mathematical formulation to solve a max flow problem.

15. Define: State Space Tree.

PART - C (5 x 16 = 80 Marks)

16. (a) Discuss the fundamentals of analysis framework and notations used in algorithm design. (16)

Or

(b) Write the linea search algorithm and analyse for its best,worst and average case time complexity. (16)

17. (a) (i) Using brute force method, write a pseudo code to find the two closest points in a set of n points and also identify the basic operation . How many number of times the basic operation can be executed. (8)

(ii) Let A and B be the two 2 x 2 matrices. Using Strassen's matrix multiplication find the product of C = A x B. Find how many number of multiplication and addition operations are performed? (8)

Or

(b) Write a pseudo code for a divide and conquer algorithm for sorting the two unsorted array into a single sorted array with your own suitable data set. Setup and solve the recurrence relation for the number of key comparisons made by the algorithm. (16)

18. (a) Design a to find a solution for following instances of the knapsack problem using dynamic programming algorithm and memory function algorithm. Capacity W = 5. (16)

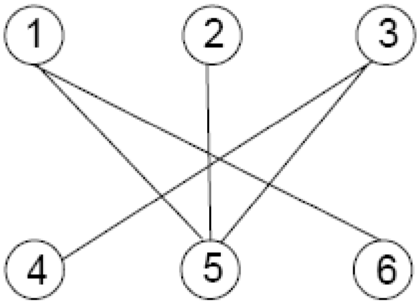
Item	Weight	Value
1	2	$12
2	1	$10
3	3	$20
4	2	$15

Or

(b) Write a pseudo code for Huffman coding algorithm and also construct a Huffman code for the following data. (16)

Symbol	A	B	C	D	-
Frequency	0.35	0.1	0.2	0.2	0.15

19. (a) Apply the maximum matching algorithm to the following bipartite graphs. (16)



Or

(b) Apply stable marriage algorithm, to the following instances and also construct a ranking matrix. (16)

Men's preferences				Women's preferences			
	1st	2nd	3rd		1st	2nd	3rd
Bob:	Lea	Ann	Sue	Ann:	Jim	Tom	Bob
Jim:	Lea	Sue	Ann	Lea:	Tom	Bob	Jim
Tom:	Sue	Lea	Ann	Sue:	Jim	Tom	Bob

20. (a) Explain about backtracking with suitable examples. (16)

Or

(b) Write short notes on NP-Hard and NP-Completeness. (16)