Course Code	**:**	**MCS-211**
Course Title	**:**	**Design and Analysis of Algorithms**
Assignment Number	**:**	**MCA_NEW(I)/211/Assign/2026**
Maximum Marks	**:**	**100**
Weightage	**:**	**30%**
Last Dates for Submission	**:**	**30th April 2026 (for January Session)**
		31st October 2026 (for July Session)

This assignment has four questions (80 Marks). Answer all questions. The remaining 20 marks are for your viva voce examination. You may use illustrations and diagrams to enhance the explanations. Please go through the guidelines regarding assignments given in the Programme Guide for the presentation format.

Q1:

a) Design and develop an efficient algorithm to find all the prime numbers in the range 2 to 10000.Access the complexity of this algorithm? **(2 Marks)**

b) Differentiate between Quadratic-time and Exponential-time algorithms. Give example of one problem each for these two running times. **(2 Marks)**

c) A sorted array of size n contains several duplicate values. Design an algorithm to search a value in the array. The algorithm should output all the indexes where the searched value exists. Explain the worst time complexity of this algorithm. **(2 Marks)**

d) Explain the Big Theta (Θ) and Big Omega (Ω) notations with the help of a diagram. What is the need of asymptotic bounds? Also, list the shortcomings of the asymptotic bounds. Find the Big O, Θ and Ω bounds for the following function: **(4 Marks)**

$$f(n)= 999n^3+999999n+100$$

e) Write the Right to Left binary exponentiation algorithm. Demonstrate the use of this algorithm to compute the value of 5^{23}. Show all steps of the computation. Explain the worst-case complexity of this algorithm. **(4 Marks)**

f) Write and explain the Insertion sort algorithm. Discuss its best and worst-case time complexity. **(3 Marks)**

g) What is a recurrence relations? When are they used? Solve the following recurrence relations using the Master's method **(3 Marks)**

a. $T(n) = 16T\left(\frac{n}{4}\right) + n^2$

b. $T(n) = 8T\left(\frac{n}{2}\right) + n^2$

Q2:

a) How can an optimisation problem be represented mathematically. Explain with the help of an example. Explain the steps of designing a Greedy solution for an optimisation problem. Also explain the concept of local and global optimal solutions. Solve the following fractional Knapsack problem using greedy approach. Show all the steps. **(4 Marks)**

Suppose there is a knapsack of capacity 10 Kg and the following 5 items are to packed in it. The weight and profit of the items are as under:

$(p_1, p_2,\ldots, p_5) = (10, 20, 18, 15, 5)$
$(w_1, w_2,\ldots, w_5) = (2, 5, 3, 5, 1)$

b) Assuming that data to be transmitted consists of only characters'a' to 'g', design the Huffman codes for the following frequencies of character data. Show all the steps of building a Huffman tree. Also, show how a coded sequence using Huffman code can be decoded. **(4 Marks)**

a:25, b:3, c:2,d:6, e:40, f:14, g:10

c) Expalin the PARTITION algorithm of the Quick Sort taking the last element as pivot. Demonstrate the use of Quick sort algorithm for sorting the following data of size 8: [1, 5, 8, 11, 16, 19, 29, 38]. Compute the complexity of the worst case of Quick Sort algorithm. **(4 Marks)**

d) Explain the divide and conquer approach of multiplying two large matrices of size 16 × 16. Compute the time complexity of this approach. **(4 Marks)**

e) Write and explain the algorithm of Topological sorting with the help of an example. Also, define the concept of strongly connected components in a directed Graph. **(4 Marks)**

Q3: Consider the following Graph:

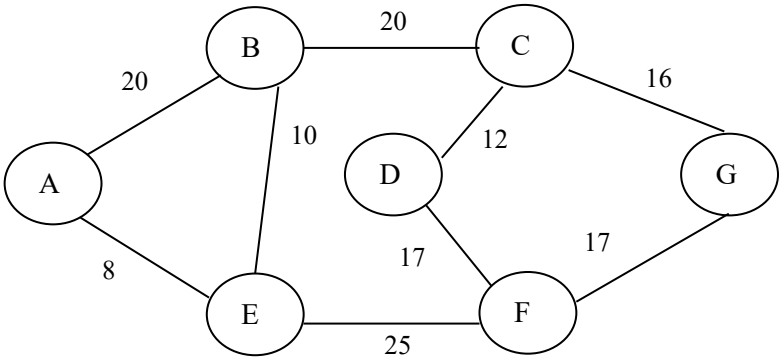


Figure 1: A sample weighted Graph

a) Write the Kruskal's algorithm to find the minimum cost spanning tree of a graph. Also, find the time complexity of Kruskal's algorithm. Demonstrate the use of Kruskal's algorithm and Prim's algorithm to find the minimum cost spanning tree for the Graph given in Figure 1. Show all the steps. **(4 Marks)**

b) Write the Bellman-Ford shortest path algorithm. What is the time complexity of this shortest path algorithm? Find the shortest paths from the vertex 'A' using Bellman-Ford's shortest path algorithm for the graph given in Figure 1. Show all the steps of computation. **(4 Marks)**

c) Write the algorithm to find the Optimal Binary Search Tree. Demonstrate this algorithm to find the Optimal Binary Search Tree for the following probabilities of search (where p_i represents the probability that the search will be for the key node k_i, whereas q_i represents that the search is for dummy node d_i. Make suitable assumptions, if any) **(6 Marks)**

i	0	1	2	3	4
p_i		0.15	0.05	0.15	0.25
q_i	0.10	0.05	0.05	0.10	0.10

d) Given the following sequence of chain multiplication of the matrices. Find the optimal way of multiplying these matrices: **(2 Marks)**

Matrix	Dimension
A1	25 × 10
A2	10 × 35
A3	35 × 10
A4	10 × 30

e) Explain the Knuth Morris Pratt (KMP) algorithm for finding a string in a text/paragraph with the help of an example. Find the time complexity of this algorithm. **(4 Marks)**

Q4: a) Explain the term intractable problems with the help of an example. Define the following problems with the help of an example. **(4 Marks)**

(i) Travelling Salesman Problem
(ii) Clique Problem

b) What are NP and NP-Complete classes of problems? Explain with the help of at least two examples each. **(4 Marks)**

c) Define the relationship between P, NP, NP-Complete and NP-Hard classes of problems. How are they different from each other. **(4 Marks)**

d) Describe the following Problems: **(8 Marks)**

(i) 3-CNF SAT Problem
(ii) 0/1 Knapsack Problem
(iii) Vertex Cover Problem
(iv) Graph colouring problem

Q1: a) Design and develop an efficient algorithm to find all the prime numbers in the range 2 to 10000.Access the complexity of this algorithm? (2 Marks)

Answer

An efficient method is the Sieve of Eratosthenes. Instead of testing every number separately for divisibility, it marks the multiples of each prime. For the required range, create a Boolean array isPrime[0…10000], initially true, and mark 0 and 1 as non-prime.

```
SIEVE(10000)
1. isPrime[0…10000] <- true
2. isPrime[0] <- false; isPrime[1] <- false
3. for p <- 2 to floor(sqrt(10000))
4.     if isPrime[p] = true
5.         for m <- p*p to 10000 step p
6.             isPrime[m] <- false
7. output every i for which isPrime[i] = true
```

Starting at p^2 is sufficient because smaller multiples of p have already been marked by smaller prime factors. Thus the algorithm produces all primes from 2 through 10000 without repeated trial division. For a general upper bound n, the time complexity is O(n log log n) and the space complexity is O(n). For n = 10000, the array needs only 10001 Boolean positions.

b) Differentiate between Quadratic-time and Exponential-time algorithms. Give example of one problem each for these two running times. (2 Marks)

Answer

Feature	Quadratic-time	Exponential-time
Growth	Running time is proportional to n^2.	Running time grows like c^n, usually 2^n or another constant base greater than 1.
Effect of doubling n	Work becomes about 4 times larger.	Work may become enormously larger; for 2^n it becomes 2^n times the old scale when n doubles.
Typical source	Two nested passes over the input.	Enumeration of many subsets or combinations.
Example	Insertion sort in the worst case: $\Theta(n^2)$.	Brute-force subset search: $\Theta(2^n)$.