ALGORITHM DESIGNING AND PROMLEM SOLVING

Section A

1. Define a procedure and give an example of one in pseudocode.
2. Explain decomposition and list two real-life examples where it’ s applicable.
3. What is pattern recognition in computational thinking? Provide a practical example.
4. Write pseudocode to calculate the sum of the first ten positive integers.
5. Describe abstraction and give an example of how it is used in a mobile application.
6. Create a flowchart for checking if a number is prime.
7. Write an algorithm in structured English for a program that outputs the square of numbers from 1 to 5.
8. What are the differences between structured English and pseudocode?
9. Illustrate how stepwise refinement simplifies a complex problem with a brief example.
10. Write a pseudocode to check if a number is even or odd.
11. Create an identifier table for a program that calculates the area of a rectangle.
12. Explain the importance of input validation in an algorithm.
13. Using decomposition, divide the task of creating a shopping list application into smaller tasks.
14. Draw a flowchart for a program that counts down from 10 to 1.
15. Write pseudocode to find the maximum of three numbers.
16. Explain selection statements and give an example using pseudocode.
17. In a pseudocode example, explain the difference between a loop and an IF statement.
18. Create a FOR loop to print numbers 1 to 10.
19. Define iteration and list the types of loops used in pseudocode.
20. Write pseudocode for converting a temperature from Celsius to Fahrenheit.

21. Illustrate a WHILE loop that prints "Hello" three times.
22. Explain the purpose of comments in pseudocode.
23. Describe boolean logic and create an example in pseudocode using AND, OR, NOT.
24. Design an algorithm in structured English that outputs the first 5 even numbers.
25. Explain the significance of meaningful variable names with examples.
26. What is structured programming, and why is it important?
27. Describe the CASE statement with an example in pseudocode.
28. Write pseudocode to find the factorial of a number.
29. Differentiate between definite and indefinite iteration with examples.
30. Explain error handling in pseudocode with a simple example.
31. Define modular programming and its benefits.
32. Write pseudocode to reverse a string.
33. Explain function vs procedure in terms of return values.
34. Describe a nested IF statement and give an example.
35. Explain the use of subroutines and give an example.
36. Write pseudocode to count the number of vowels in a string.
37. Define scope of variables and give an example.
38. Describe global vs local variables in pseudocode.
39. Illustrate a DO UNTIL loop with an example.
40. Write pseudocode to calculate the average of a list of numbers.
41. Define recursion and write a simple example.
42. Differentiate between pseudocode and real code.
43. Write pseudocode to check if a year is a leap year.
44. Describe program flow and its importance in pseudocode.
45. Explain the difference between syntax and semantic errors.
46. Write a simple pseudocode for a login system.
47. Define string manipulation and provide an example.
48. Describe the importance of testing an algorithm.
49. Write pseudocode for a program that calculates the square root of a number.
50. Explain dry-run testing with a pseudocode example.

Section A

1. Write pseudocode for a binary search algorithm.
2. Explain time complexity and analyze the time complexity of a FOR loop.
3. Design a pseudocode algorithm to sort an array using insertion sort.
4. Write a recursive algorithm for calculating the factorial of a number.
5. Explain trace tables and create one for a given pseudocode.
6. Illustrate error handling for invalid inputs in a temperature conversion algorithm.
7. Write pseudocode for a program that finds the longest word in a list.
8. Create a flowchart for a program that counts the number of words in a paragraph.
9. Explain the importance of modularity in large programs.
10. Define algorithm efficiency and discuss factors that affect it.
11. Design an algorithm in pseudocode for bubble sort.
12. Create pseudocode for a function that returns the nth Fibonacci number.
13. Explain divide and conquer strategies in algorithms with examples.
14. Write pseudocode for checking if a string is a palindrome.
15. Explain binary search and why it's more efficient than linear search.
16. Create a pseudocode algorithm for calculating GCD of two numbers.
17. Illustrate how error messages improve debugging in pseudocode.
18. Define data validation and give an example with pseudocode.
19. Write pseudocode to simulate a simple calculator.
20. Explain the importance of identifier tables in tracking variables.
21. Design a menu-driven program with at least three options in pseudocode.
22. Describe pass-by-value vs pass-by-reference with examples.
23. Write pseudocode for depth-first search (DFS) in a graph.
24. Differentiate between procedures and functions with pseudocode examples.
25. Explain big O notation and provide examples.
26. Describe encapsulation in terms of pseudocode.
27. Write pseudocode to merge two sorted arrays.
28. Explain parameterized functions in pseudocode.

29. Create pseudocode for a binary tree traversal.
30. Describe backtracking algorithms and give an example.
31. Write pseudocode for finding prime factors of a number.
32. Explain algorithm optimization with an example.
33. Write a dynamic programming example in pseudocode.
34. Explain the difference between iterative and recursive algorithms.
35. Create pseudocode for a hashing function.
36. Describe state machines and give an example.
37. Write pseudocode to implement a queue data structure.
38. Explain the stack data structure and provide a use case.
39. Write pseudocode for a simple spell-checking algorithm.
40. Describe the concept of memoization with an example.
41. Explain greedy algorithms and write a simple example.
42. Write pseudocode for a breadth-first search (BFS).
43. Explain control structures and their importance in pseudocode.
44. Describe the difference between linear and binary search.
45. Write pseudocode to parse and evaluate a mathematical expression.
46. Explain modulus operation with an example.
47. Write a program in pseudocode for linear regression.
48. Explain hash tables and write a simple pseudocode for one.
49. Describe dynamic memory allocation.
50. Write pseudocode to generate all permutations of a string.

Section C

1. Explain space complexity and analyze it for a specific algorithm.
2. Write pseudocode for Dijkstra's algorithm.
3. Create a flowchart for QuickSort and explain each step.
4. Write a recursive backtracking algorithm to solve a maze.
5. Explain polynomial time complexity with examples.
6. Write pseudocode for Knuth-Morris-Pratt (KMP) string matching algorithm.
7. Describe *A search algorithm** with pseudocode.
8. Explain Red-Black Tree operations with pseudocode examples.
9. Write pseudocode for topological sorting in a directed graph.
10. Create pseudocode for dynamic programming solution to the knapsack problem.
11. Write pseudocode for prim's algorithm for finding minimum spanning trees.
12. Explain divide and conquer strategy in QuickSort.
13. Write a Monte Carlo simulation in pseudocode.
14. Describe polynomial vs exponential algorithms.
15. Write pseudocode for Kruskal's algorithm.
16. Explain Boyer-Moore string search algorithm.
17. Write pseudocode to solve the n-queens problem.
18. Describe multi-threading with pseudocode.
19. Explain dynamic programming with an example.
20. Write pseudocode for Longest Common Subsequence (LCS) problem.
21. Describe memoization vs tabulation.
22. Explain the Branch and Bound method with pseudocode.
23. Create a flowchart for a genetic algorithm.
24. Explain the Ford-Fulkerson algorithm.
25. Write pseudocode for Floyd-Warshall algorithm.
26. Describe the Viterbi algorithm in pseudocode.
27. Explain graph traversal in directed acyclic graphs.
28. Describe Heapsort with pseudocode.
29. Explain greedy algorithm for the fractional knapsack problem.
30. Write pseudocode for solving Sudoku using backtracking.

31. Describe Bellman-Ford algorithm.
32. Explain Huffman coding with pseudocode.
33. Write pseudocode for cycle detection in a directed graph.
34. Explain convex hull problem with pseudocode.
35. Write pseudocode for Union-Find algorithm.
36. Describe Kadane's algorithm with pseudocode.
37. Explain dynamic programming in matrix chain multiplication.
38. Write pseudocode for Turing machine simulation.
39. Describe the traveling salesman problem.
40. Write pseudocode for Levenshtein distance algorithm.
41. Explain NP-completeness with examples.
42. Describe approximation algorithms with pseudocode.
43. Write pseudocode for Breadth-First Search on a weighted graph.
44. Explain linear programming with pseudocode.
45. Describe the Ford-Fulkerson max flow algorithm.
46. Explain Eulerian paths and circuits.
47. Write pseudocode for stable matching algorithm.
48. Describe Strassen's matrix multiplication.
49. Write pseudocode for Boyle's cycle detection algorithm.
50. Explain Markov Decision Processes with pseudocode.