**Algorithm design techniques: Greedy methods, Priority queue search, Exhaustive search, Divide and conquer, Dynamic programming, Backtracking and Recursion**

### ****1. Basics of Recursion****

* What is recursion?
* What is a recursive algorithm?
* Define a recursive algorithm. How do you implement recursive algorithms while writing computer programs?
* What are the types of recursions?
* Explain different types of recursions.
* What are tail and non-tail recursions?
* Explain the statement "A junction or an object calls itself" using the idea behind it.
* Explain recursion with its disadvantages.
* Distinguish between a normal function and a recursive function.
* Do you think recursive functions are slow? Compare recursive and non-recursive functions.
* Can every recursive problem be solved iteratively?
* Write the merits and demerits of recursion in programming.
* Write the merits and demerits of a recursive function over a non-recursive function.

### ****2. Data Structures in Recursion****

* What is the data structure used to perform recursion, and why?
* How do recursive algorithms use a stack to store intermediate results? Illustrate with an example.
* Explain how recursion uses stack data structures, use the factorial calculation to illustrate the concept.
* Explain how a recursive algorithm uses a stack with a suitable illustrative stack diagram.

### ****3. Fibonacci Series and Factorial Problems****

* Write a recursive algorithm to find the Fibonacci series up to N terms.
* Write a recursive program to generate Fibonacci numbers up to the nth term.
* Write a recursive algorithm to calculate the nth Fibonacci number. Illustrate the recursion tree for a given value of n.
* Draw the recursive tree diagram for the Fibonacci sequence fib(5).
* Write recursive and iterative algorithms for Fibonacci numbers and compare the efficiency of the two algorithms.
* Explain the implementation of the factorial and Fibonacci sequences with examples.
* Write a recursive program to find the factorial of a number.

### ****4. Tower of Hanoi Problem (TOH)****

* What is the Tower of Hanoi problem?
* Write a recursive algorithm for the Tower of Hanoi (TOH) with 'n' disks.
* Write a C-program to solve the Tower of Hanoi problem for n number of disks.
* Write a recursive module for the Tower of Hanoi problem.
* Construct a recursion tree for the TOH problem with 3 disks.
* Construct a recursion tree for the TOH problem with 4 disks.
* Draw the recursion tree for the Tower of Hanoi problem for 5 disks.
* Explain the recursion algorithm for the Tower of Hanoi problem.
* Show the execution path according to the TOH algorithm.
* Explain the Tower of Hanoi algorithm.
* Illustrate the TOH algorithm for 3 disks.
* Why is recursion required? Explain with the Tower of Hanoi example.
* Write the recursion tree when the number of disks is four.
* What is the Tower of Hanoi problem? Write an algorithm to solve it.

### ****5. Prefix to Postfix Expression Conversion****

* Write a recursive algorithm to convert a prefix expression to a postfix expression.
* Draw the recursion tree and transform the following prefix expressions to postfix:
  + a) +-$ABC\*D\*\*EFG
  + b) ++A-$BCD/+EFGH

### ****6. Recursive Algorithms Efficiency and Comparison****

* Discuss the efficiency of recursion.
* How does a recursive algorithm make a program effective?

### ****7. Recursive Binary Search****

* Write a recursive algorithm to implement binary search.

### ****1. Greedy Algorithm****

* What is a greedy algorithm?
* Write short notes on greedy algorithms.
* Explain the greedy algorithm.
* Define a greedy algorithm and a heuristic algorithm.
* Write and explain an algorithm that uses the greedy approach to solve a problem.

### ****2. Deterministic and Non-Deterministic Algorithms****

* What are deterministic and non-deterministic algorithms?

### ****1. Divide and Conquer Algorithm****

* Define the divide and conquer approach to problem solving.
* What is a divide and conquer algorithm?
* Explain the divide and conquer technique.
* Write an algorithm that uses the divide and conquer approach and explain it with an example.
* Explain the divide and conquer algorithm, taking reference to Merge Sort.
* Where can we use the divide and conquer algorithm? Write any two examples.
* Explain the concept of the Divide and Conquer algorithm.
* Write short notes on:
  + a. Divide and Conquer

### ****2. Dynamic Programming****

* What is dynamic programming?
* Write short notes on:
  + Dynamic Programming
  + Dynamic memory allocation

### ****3. Backtracking Algorithm****

* What is a backtracking algorithm?
* Explain the backtracking algorithm.

### ****4. Randomized Algorithms****

* What are randomized algorithms?
* Explain randomized algorithms.

### ****1. Characteristics and Complexity of Algorithms****

* What is an algorithm? Write down the features of an algorithm.
* What are the characteristics of a good algorithm?
* What are the major characteristics of algorithms?
* What do you mean by the complexity of algorithms? How do you find time complexity?
* How to find the complexity of algorithms? Explain.

### ****2. Huffman Algorithm****

* What do you mean by the Huffman Algorithm? Explain with an example.
* Explain the characteristics of Huffman's algorithm and its application.
* Describe the significance of the Huffman tree.
* Describe the procedure for constructing a Huffman tree. Illustrate it with an example.
* Construct a Huffman code for the given string: BCCAAADDACACBB.
* Construct a Huffman code for the given symbols:
  + Symbols: A, B, C, D, E, F
  + Frequency (in thousands): 35, 18, 10, 10, 20, 20
* Illustrate the importance of the Huffman algorithm in data communication over the network using the following string: BCCAAADDACACBB.
* A file contains 100 symbols with the following characters and their probability of occurrence. Build a Huffman tree according to the Greedy Strategy:
  + a → 48, b → 11, c → 9, d → 14, e → 7, f → 11

### ****1. Sorting Algorithms****

* Explain merge sort in brief with an algorithm and suitable example.
* Explain Quick Sort algorithm.
* Write an algorithm for Quick Sort.
* Explain the basic principles of Quick Sort and write down its partition algorithm.
* Compare Quick Sort and Merge Sort.
* Compare the sorting time efficiencies of Quick Sort and Merge Sort.
* In which case does insertion sort perform better than Quick Sort?
* Sort the following elements using Quick Sort:
  + Example 1: 14, 23, 7, 10, 33, 56, 80, 66, 70
  + Example 2: 62, 71, 69, 26, 31, 85, 93, 58, 47, 99
  + Example 3: 30, 25, 79, 19, 48, 28, 21, 44, 120
* Hand test the Quick Sort algorithm with the following array of numbers: 78, 34, 21, 43, 7, 18, 9, 36, 38, 19.
* What is the time complexity of the Quick Sort algorithm?
* Trace the sorting steps in a radix sort algorithm for the following data:  
  12, 11, 30, 21, 25, 39, 36, 17, 29, 10, 26, 33, 7, 9

### ****2. Algorithm Analysis****

* Explain the use of Big O notation in analyzing algorithms.
* Describe the importance of the growth function in algorithms.

### ****5. Fibonacci Algorithms****

* Write recursive and iterative algorithms for Fibonacci numbers and compare their efficiency.

### ****3. B-Trees****

* Construct the B-tree of order 5 using the following data:
  + 20, 10, 26, 55, 80, 11, 9, 60, 67, 55, 22, 76, 56, 45, 34, 100, 150

### ****4. Binary Trees****

* Describe different types of applications of Binary trees.
* Describe the recursive procedure of the Binary search technique. Discuss the efficiency of Binary search.
* How do you implement the binary search algorithm? What is the time complexity of this algorithm?