**Recursion**

**✅ 1. Theory Questions (Basic to Advanced)**

**🔹 Basic**

1. What is recursion?
2. What are the two parts of a recursive function?
3. What is a base case and why is it necessary?
4. Difference between recursion and iteration.
5. What is stack overflow error in recursion?

**🔹 Intermediate**

1. How is recursion implemented internally in Java?
2. How does the call stack work in recursion?
3. What is tail recursion? Does Java support it?
4. How to optimize recursive functions to avoid stack overflow?

**🔹 Advanced**

1. What are the advantages and disadvantages of recursion?
2. Explain the concept of backtracking with recursion.
3. Compare recursive and dynamic programming approaches.
4. How can memoization help in recursive solutions?
5. What is mutual recursion? Give an example.

**💻 2. Coding Questions (Basic to Advanced)**

**🔹 Basic**

1. Print numbers from 1 to N using recursion
2. Find factorial of a number using recursion
3. Calculate nth Fibonacci number
4. Sum of digits of a number
5. Reverse a string using recursion

**🔹 Intermediate**

1. Check if a string is a palindrome
2. Find GCD (HCF) of two numbers using recursion
3. Print an array in reverse order
4. Find power of a number (a^b)
5. Convert number to words (e.g., 1947 → one nine four seven)

**🔹 Backtracking (Advanced)**

1. Rat in a Maze
2. N-Queens Problem
3. Sudoku Solver
4. Generate all subsets of a set (Power set)
5. Permutations of a string or array

**🔹 More Advanced Recursive Problems**

1. Remove duplicates from a string
2. Generate valid parentheses combinations
3. Recursive binary search
4. Count number of paths in a maze (m x n grid)
5. Fast Exponentiation (Binary exponentiation using recursion)

**🔄 Bonus: Important Concepts to Master**

* Recursion tree diagram (to visualize calls)
* Space and time complexity in recursive solutions
* Recursion vs Dynamic Programming
* Converting recursion to iteration and vice versa