Array Problems (Intermediate Level)

1. **Find Maximum and Minimum in an Array**
   Given an array of n integers, find the maximum and minimum values in **O(n)** time.
2. **Reverse an Array in-place**
   Reverse an array **without using extra space**.
3. **Rotate an Array k Times (Cyclic Rotation)**
   Rotate an array k times to the right efficiently.
4. **Find the Maximum Sum of a Contiguous Subarray (Kadane's Algorithm)**
   Given an array, find the contiguous subarray that has the **maximum sum**.
5. **Reverse Words in a String**
   Example: "I love coding" → "coding love I"
6. **Check if Two Strings are Anagrams**
   Example: "listen" and "silent" → True
7. **Find the Longest Common Prefix Among Strings in an Array**
   Example: ["flower", "flow", "flight"] → "fl"
8. **Remove Consecutive Duplicates from a String**
   Example: "aabbcca" → "abc"
9. **Find the Longest Palindromic Substring (Expand Around Center Algorithm)**
   Example: "babad" → "bab"
10. **Check if a Given String is a Rotation of Another String**
    Example: "ABCD" and "CDAB" → True
11. **Find the Longest Substring Without Repeating Characters (Sliding Window)**
    Example: "abcabcbb" → 3 ("abc")
12. **Find All Permutations of a Given String (Backtracking Approach)**
    Example: "ABC" → ["ABC", "ACB", "BAC", "BCA", "CAB", "CBA"]
13. **Implement the Rabin-Karp Algorithm for Pattern Searching**
    Example: Find all occurrences of "abc" in "abcxabcdabxabcdabcdabcy".
14. **Rotate a 2D Matrix by 90 Degrees (In-place)**
    Rotate an n x n matrix 90° clockwise.
15. **Find the Row with the Maximum Sum in a 2D Array**
    Given an n x m matrix, find the row with the maximum sum.
16. **Find the Saddle Point in a Matrix**
    A saddle point is an element that is the minimum in its row and maximum in its column.
17. **Search for an Element in a Row-wise and Column-wise Sorted Matrix**
    Example:
18. **Spiral Order Traversal of a Matrix**
    Given an n x m matrix, print its elements in spiral order.
19. **Find the Sum of All Elements in a 3D Array**
    Given a p x q x r array, compute the sum of all elements.
20. **Find the Maximum Element in a 3D Array**
    Given a p x q x r array, find the maximum element.

Array Problems (Analytical)

21. Treasure Hunt in the Desert 🏜️

A treasure hunter has a **list of gold coin values** he found while exploring an ancient desert. The coins are in an **array**, and he wants to **maximize his profit by collecting non-adjacent coins** (he cannot pick two consecutive values).

- 🔹 **Data Structure: 1D Array**
- 🔹 **Hint: Dynamic Programming** (Similar to "House Robber" problem)

💡 **Input:**

coins = [3, 2, 7, 10]

💡 **Output:**

13 (Pick 3 and 10)

22. Marathon Leaderboard 🏃

A race organizer is tracking the **top 100 runners' times** in a **sorted array**. A new runner completes the race, and the organizer must **insert the new time** while maintaining the order.

- 🔹 **Data Structure: 1D Sorted Array**
- 🔹 **Hint: Binary Search + Insertion**

💡 **Input:**

times = [120, 125, 130, 135, 140]
new_time = 128

💡 **Output:**

[120, 125, 128, 130, 135, 140]

23. Virus Spread in a Network 🦠

A computer network is under attack by a virus, and each computer is labeled with a **1 (infected) or 0 (safe)**. The **spread follows adjacency rules** (left or right). Find how many time steps it takes to infect the entire system.

- 🔹 **Data Structure: 1D Array (Boolean Representation)**
- 🔹 **Hint: BFS (Breadth-First Search) Simulation**

💡 **Input:**

network = [0, 1, 0, 0, 1, 0]

💡 **Output:**

2 (Steps required to infect all computers)

24. Fireworks at a Festival 🎆

A city is arranging fireworks. Each firework has an **ignition time**, and the mayor wants to know how many fireworks **explode within a given time range**.

- 🔹 **Data Structure: 1D Array + Prefix Sum**

💡 **Input:**

fireworks = [3, 7, 10, 15, 20]
query_range = [5, 15]

💡 **Output:**

3 (Fireworks at 7, 10, 15 explode within the range)

25. Movie Ratings Analytics 🍿

A streaming service tracks **daily movie ratings** in an array and wants to determine **the longest streak of increasing ratings** for marketing.

- 🔹 **Data Structure: 1D Array**
- 🔹 **Hint: Sliding Window / Greedy**

💡 **Input:**

ratings = [1, 2, 2, 3, 4, 1, 5, 6]

💡 **Output:**

4 ([1, 5, 6] is the longest increasing streak)

26. Island Survival 🌴

A survivor is stranded on an island (represented as a n x m **grid**). Some cells contain **food (F)**, others are **water (W)**. Determine the **minimum steps to reach food** from his starting position.

- 🔹 **Data Structure: 2D Array (Matrix Representation)**
- 🔹 **Hint: BFS (Shortest Path in a Grid)**

💡 **Input:**

[['W', 'W', 'F'],
['W', 'S', 'W'],
['F', 'W', 'W']]

💡 **Output:**

1 (One step to the nearest food)

27. Alien Symbol Translator 👽

A scientist finds a **grid of alien symbols** and wants to count how many times a specific **symbol (X) appears** in the grid.

- 🔹 **Data Structure: 2D Array (Matrix Search)**

💡 **Input:**

[['X', 'O', 'O'],
['O', 'X', 'X'],
['X', 'O', 'O']]

💡 **Output:**

4 (Occurrences of 'X')

28. Space Navigation System 🚀

An astronaut in a **3D space station (x, y, z grid)** needs to find the shortest path from one room to another.

- 🔹 **Data Structure: 3D Array**
- 🔹 **Hint: 3D BFS Search**

💡 **Input:**

A 5 x 5 x 5 grid where 0 = open path, 1 = obstacle, S = start, E = exit.

💡 **Output:**

Minimum steps to reach exit

29. DNA Mutation Mapping 🧬

A **DNA strand is modeled as a 4D matrix**, where each layer represents a different time step in an experiment. Find how many cells changed between **time T and T+1**.

- 🔹 **Data Structure: 4D Array**
- 🔹 **Hint: Difference Calculation**

💡 **Input:**
A 4 x 4 x 4 x 4 array of A, T, G, C.
💡 **Output:**
Number of mutations

30. **Quantum Particle Tracking** ⚛️
A quantum physicist is tracking **particle states** in a w x x x y x z matrix. Each state is 0 or 1, and they need to count how many times the state **flips** between 0 and 1.
🔹 **Data Structure: 4D Array**
🔹 **Hint: Bitwise Operations / Difference Checking**
💡 **Input:**
4D matrix representing quantum states over time.
💡 **Output:**
Total state flips detected

31. **Decoding a Secret Message** 🔍
A hacker intercepted a **string-based cipher** and needs to decode it. The cipher follows a **Caesar Shift** (each letter is shifted by k).
🔹 **Data Structure: String Manipulation**
💡 **Input:**
cipher = "frqjudwxodwlrq" (Shift = 3)
💡 **Output:**
"congratulation"

32. **Validate Palindromic DNA Sequences** 🧬
A biologist needs to check if a **DNA sequence** is a palindrome (AGCTTTCGA → ✅).
🔹 **Data Structure: String**
💡 **Input:**
"AGCTTTCGA"
💡 **Output:**
True