EASY LEVEL

1) Two Sum

Given an array of integers `nums` and an integer `target`, return the indices of the two numbers such that they add up to the target.
You may assume each input has exactly one solution, and you may not use the same element twice.

Example
Input: nums = [2,7,11,15], target = 9
Output: [0,1]

---

2) Valid Parentheses

Given a string containing just the characters `'('`, `')'`, `'{'`, `'}'`, `'['` and `']'`, determine if the input string is valid.

A string is valid if:

- Open brackets are closed by the same type.
- Open brackets are closed in the correct order.

Example
Input: "()[]{}" → true
Input: "(]" → false

---

3) Contains Duplicate

Given an integer array `nums`, return true if any value appears at least twice.

Example
Input: [1,2,3,1] → true
Input: [1,2,3,4] → false

---

4) First Unique Character in a String

Given a string `s`, return the first non-repeating character.
If none exists, return _.

Example
Input: "leetcode" → "l"
Input: "aabb" → "_"

---

5) Move Zeroes

Move all zeros in the array to the end while maintaining the order of non-zero elements.

Example
Input: [0,1,0,3,12]
Output: [1,3,12,0,0]

---

6) Valid Anagram

Given two strings `s` and `t`, return true if `t` is an anagram of `s`.

Example
Input: s="anagram", t="nagaram" → true

---

7) Intersection of Two Arrays

Return the intersection of two arrays as unique elements.

Example
Input: [1,2,2,1], [2,2] → [2]

---

8) Single Number

Every element appears twice except one. Find that single one.

Example
Input: [4,1,2,1,2] → 4

---

9) Longest Consecutive Sequence

Given an unsorted array, find the length of the longest consecutive elements sequence.

Example
Input: [100,4,200,1,3,2] → 4

---

10) Multiply Strings

Multiply two non-negative numbers represented as strings without converting to integers.

Example
Input: "123", "456" → "56088"

---

11) Valid Parenthesis String (* allowed)

Given a string with `'('`, `')'`, and `'*'`, return true if it can be valid.
`*` can be `'('`, `')'`, or empty.

Example
Input: "(*))" → true

---

12) Word Pattern

Check if a string follows a given pattern.

Example
pattern = "abba"
s = "dog cat cat dog" → true

---

13) Isomorphic Strings

Two strings are isomorphic if characters map one-to-one.

Example
Input: "egg", "add" → true
Input: "foo", "bar" → false

---

14) Longest Substring Without Repeating Characters

Return the length of the longest substring without repeating characters.

Example
Input: "abcabcbb" → 3

---

15) Group Anagrams

Group strings that are anagrams.

Example
Input: ["eat","tea","tan","ate","nat","bat"]
Output: [["eat","tea","ate"],["tan","nat"],["bat"]]

---

16) Product of Array Except Self

Return an array where each element is product of all elements except itself.
Do not use division.

Example
Input: [1,2,3,4] → [24,12,8,6]

---

17) Encode and Decode Strings

Design an algorithm to encode a list of strings to a single string and decode it back.

---

18) Top K Frequent Elements

Return the k most frequent elements.

Example
Input: [1,1,1,2,2,3], k=2 → [1,2]

---

19) Longest Palindromic Substring

Return the longest palindromic substring.

Example
Input: "babad" → "bab" or "aba"

---

20) Subarray Sum Equals K

Return the total number of subarrays whose sum equals `k`.

Example
Input: nums=[1,1,1], k=2 → 2

Round 2 practice

1) Poison Bottle Problem

You are given **1000 bottles**, and exactly **one bottle is poisoned**.
You also have **10 test strips**. A strip shows a positive result if poison is present, but it takes **24 hours** to get the result.

You can use the strips only once.
Design a strategy to identify the poisoned bottle within **24 hours**.

---

2) Clock Angle

Given a time in hours and minutes, calculate the **smaller angle** between the hour hand and minute hand of an analog clock.

Example
Input: hour = 3, minutes = 15
Output: 7.5 degrees

---

3) Climbing Stairs

You are climbing a staircase with `n` steps.
Each time you can climb either **1 step or 2 steps**.
Return the number of distinct ways to reach the top.

---

4)Missing Number

Given an array containing `n` distinct numbers in the range `[0, n]`, return the only number missing from the array.

Example
Input: [3,0,1] → Output: 2

---

5) Majority Element

Given an array `nums`, return the element that appears **more than ⌊n/2⌋ times**.
You may assume that such an element always exists.

□ String Manipulation (Similar to Pascal → kebab)

6) CamelCase to snake_case

You are given a string written in **CamelCase**.
Convert it into **snake_case**.

Rules:

- All letters must be lowercase.
- Words are separated using `_`.

Example
Input: `"HelloWorldTest"`
Output: `"hello_world_test"`

---

7) Reverse Words in a Sentence

Given a string `s`, reverse the order of words.

Example
Input: `"I love coding"`
Output: `"coding love I"`

---

8) Count Words in PascalCase

Given a PascalCase string, count the number of words in it.

Example
Input: `"SaveChangesInTheEditor"`
Output: `5`

---

9) String Compression

Compress a string by replacing consecutive repeating characters with character + count.

Example
Input: `"aabcccccaaa"`
Output: `"a2b1c5a3"`

10) String Rotation Check

Given two strings `s1` and `s2`, check if `s2` is a rotation of `s1`.

Example
Input: `"waterbottle"`, `"erbottlewat"` → true

□ Arrays Logic

11) Leaders in an Array

An element is a leader if it is **greater than all elements to its right**.
Return all leaders.

Example
Input: [16,17,4,3,5,2]
Output: [17,5,2]

---

12) Equilibrium Index

Find an index such that the sum of elements to the left equals the sum of elements to the right.

Return -1 if no such index exists.

---

13) Second Largest Element

Find the second largest element in an array in **one pass**.

---

14) Rotate Array by k Steps

Rotate the array to the right by `k` steps.

Example
Input: nums=[1,2,3,4,5,6,7], k=3
Output: [5,6,7,1,2,3,4]

---

15) Valid Palindrome (Ignore Symbols)

Check if a string is a palindrome ignoring spaces, punctuation, and case.

Example
Input: `"A man, a plan, a canal: Panama"` → true

● Hashing / Pattern Logic

16) Ransom Note

Given two strings `ransomNote` and `magazine`, determine if the ransom note can be constructed using letters from the magazine.

Each letter can be used only once.

---

17) Happy Number

Starting with a number, replace it with the sum of squares of its digits repeatedly.
Return true if the number eventually becomes 1.

---

18) Valid Palindrome After Removing One Character

Return true if the string can become a palindrome after deleting at most one character.

---

19) Longest Common Prefix

Given an array of strings, find the longest common prefix.

---

20) Excel Column Number

Given a column title as in Excel (A, B, ..., Z, AA, AB, …), return its column number.

Example
A → 1
AB → 28
ZY → 701

3rd Round

Got it 👍 These are **classic medium array questions** (logic + in-place manipulation).
I'll give you **similar level problems with proper interview-style statements.**

⭐ Level of your questions

Difficulty: **Medium (6.5–7/10)**
Topics:

- Array traversal in O(n)
- In-place modification
- Greedy / tracking values
- Cycle placement / index mapping

So here are **10 new problems of the SAME LEVEL** 👇

☐ MEDIUM ARRAY / IN-PLACE LOGIC QUESTIONS

---

1) Third Maximum Number

Given an integer array `nums`, return the **third distinct maximum number** in the array.
If the third maximum does not exist, return the maximum number.

Example

```
Input: [2,2,3,1]
Output: 1
```

---

2) Maximum Difference Between Increasing Elements

Given an array `nums`, find the maximum difference `nums[j] - nums[i]` such that `j > i`.
Return `-1` if no such pair exists.

Example

```
Input: [7,1,5,4]
```

```
Output: 4   (5 - 1)
```

3) Rearrange Array by Sign (In-Place)

Given an array with equal number of positive and negative integers, rearrange it so that positives and negatives appear alternately.

Example

```
Input: [3,1,-2,-5,2,-4]
Output: [3,-2,1,-5,2,-4]
```

4) Find All Duplicates in Array (No Extra Space)

Given an array of integers where `1 ≤ nums[i] ≤ n`, return all duplicates.
You must solve it without using extra space.

Example

```
Input: [4,3,2,7,8,2,3,1]
Output: [2,3]
```

5) Set Mismatch

An array of size `n` contains numbers from `1..n`.
One number is duplicated and one number is missing.
Find both numbers in O(n) time and O(1) space.

Example

```
Input: [1,2,2,4]
Output: [2,3]
```

6) Cyclic Sort / Missing Numbers

Find all numbers missing from array `[1..n]` in O(n) time and O(1) extra space.

Example

```
Input: [4,3,2,7,8,2,3,1]
Output: [5,6]
```

7) Next Greater Element in Circular Array

Given a circular array, return the next greater number for each element.

Example

```
Input: [1,2,1]
Output: [2,-1,2]
```

---

8) Sort Colors (Dutch National Flag)

Given an array containing 0, 1, and 2, sort them **in-place**.

Example

```
Input: [2,0,2,1,1,0]
Output: [0,0,1,1,2,2]
```

---

9) Rotate Matrix 90° In-Place

Rotate an `n x n` matrix by 90 degrees clockwise without extra space.

---

10) Wiggle Sort

Rearrange array such that:

```
nums[0] <= nums[1] >= nums[2] <= nums[3] ...
```

Do it in O(n) time.

Project Related Discussion questions:

Project Discussion — Proper Interview Style Questions

□ Section 1 — Project Overview

1) Project Elevator Pitch

Describe your project clearly in **under 2 minutes**.
Your explanation must include:

- Problem statement
- Target users
- Main features
- Your role in the project

---

2) Problem Identification

What real-world problem does your project aim to solve?
Why is this problem important?

---

3) User Definition

Who are the primary users of your application?
How does your solution help them?

---

4 Motivation Behind Project Choice

Why did you choose this project?
Was it based on a real need, interest, or academic requirement?

---

5 Contribution Clarification

If this was a team project, explain your **exact contribution** in terms of design, coding, and testing.

▯ Section 2 — System Design & Architecture

6) Architecture Explanation

Explain the overall architecture of your project.
Describe how the frontend, backend, database, and APIs interact.

---

7) Technology Selection

Why did you choose your specific tech stack (language, framework, database)?
What alternatives did you consider?

---

8) Data Flow Description

Describe how data flows through your system from user input to final output.

---

9) API Design

Describe the APIs used or created in your project.
Explain request/response structure and their purpose.

---

10) Component Breakdown

Explain the major modules/components of your system and their responsibilities.

▯ Section 3 — Implementation & Coding

11) Data Structures Usage

Which data structures did you use in your project and why?

---

12) Algorithm Usage

Which algorithms were used in your project?
Explain where and why they were necessary.

---

13) Performance Optimization

What steps did you take to optimize performance?

---

14) Complexity Analysis

Pick one core feature and explain its **time and space complexity**.

---

15) Error Handling

How does your system handle invalid inputs or unexpected failures?

🌐 Section 4 — Database & Storage

16) Database Choice

Why did you choose SQL / NoSQL / other database?

---

17) Schema Design

Explain your database schema or collection structure.

---

18) Query Optimization

How did you improve database query performance?

---

19) Data Consistency

How do you ensure data integrity and avoid data loss?

▯ Section 5 — Scalability & Reliability

20) Scalability Planning

How would your system handle **100x more users**?

---

21) Failure Handling

What happens if the server or database crashes?

---

22) Load Handling

How would you improve system performance under heavy traffic?

🔵 Section 6 — Security & Deployment

23) Security Measures

What steps did you take to secure your application?

---

24) Deployment Strategy

How would you deploy this project to production?

---

25) Future Improvements

If you had more time, what features or improvements would you add?

Design patterns questions

PART 1 — Conceptual Questions

1) What are Design Patterns?

Explain what design patterns are and why they are used in software development.

---

2) Types of Design Patterns

Explain the three main categories:

- Creational
- Structural
- Behavioral
  Give examples of each.

---

3) SOLID Principles

Explain how SOLID principles relate to design patterns.

---

4) When should you NOT use design patterns?

Explain drawbacks and over-engineering.

---

5) Difference: Design Pattern vs Framework vs Architecture

Explain differences with examples.

⭑ PART 2 — Singleton Pattern

6) Singleton Implementation

Design a class that allows only one object to be created.

Follow-ups:

- Make it thread-safe
- Prevent cloning and reflection breaking singleton

---

7) Real-world use of Singleton

Give examples where singleton is useful.

⭑ PART 3 — Factory Pattern

8) Factory Pattern Design

Design a **Shape Factory** that creates objects like Circle, Square, Rectangle based on input.

---

9) Factory vs Abstract Factory

Explain difference with examples.

---

10) Real world use of Factory Pattern

Where is Factory used in real systems?

⭑ PART 4 — Builder Pattern

11) Builder Pattern Design

Design a **Pizza Builder** where user can choose toppings step by step.

---

12) When to use Builder vs Constructor?

Explain telescoping constructor problem.

⭑ PART 5 — Observer Pattern

13) Observer Pattern Design

Design a **YouTube channel notification system**:

- Subscribers get notified when creator uploads video.

---

14) Real world uses of Observer

Explain examples (event systems, UI updates).

⭑ PART 6 — Strategy Pattern

15) Strategy Pattern Design

Design a **Payment System** supporting:

- Credit Card
- UPI
- PayPal

User should be able to switch payment method at runtime.

---

16) Strategy vs If-Else

Why is Strategy better than long conditional statements?

⭑ PART 7 — Decorator Pattern

17) Decorator Pattern Design

Design a **Coffee Ordering System**:

- Basic coffee
- Add milk, sugar, chocolate dynamically.

⭑ PART 8 — Adapter Pattern

18) Adapter Pattern Design

Design a **mobile charger adapter** converting 3-pin plug to 2-pin socket.

⭑ PART 9 — LLD Scenario Questions

19) Which design pattern would you use?

Design a **Logger System** with log levels:
INFO → DEBUG → ERROR → FATAL

(Expect: Chain of Responsibility)

---

20) Which pattern fits best?

Design a **cache system** with different eviction strategies:
LRU / FIFO / LFU
(Expect: Strategy Pattern)