General Notes:

- Indicate the level they could reach while sharing the profile.
- Share the solution coded by the candidate while sharing the profile.
- Candidate should attempt Min 2 coding problems.
- Candidate should be able to compile, execute and test the code.

Sample Questions:

1. Reverse a String without using any inbuilt programming language features.
   (Min Level for Testing: 2
   Min Level for Development Roles: 3)

   Level 1: Able to reverse in place (without using any additional memory)

   - Input: "hello" Output: "olleh"

   Level 2: Reverse a String with Numbers (Write a function that takes a string as input and reverses the string while keeping the numbers in their original positions)

   - Input: "hello123" Output: olleh123

   - Input: "hello123abc" Output: "cbaol123leh"

   Level 3: Reverse Individual Words in a Sentence with Special Characters (Write a function that takes a sentence as input and reverses each word in the sentence while keeping the special characters in their original positions.)

   - Input: "Hello, World!" Output: "olleH, dlroW!"

2. **Remove Duplicates Without Using Inbuilt Data Structures**

   (Min Level for Testing: 2
   Min Level for Development Roles: 3)
   Candidate must write and test code without using sets, maps, or language-provided deduplication.

   Level 1: Remove duplicates in a string

   - Input: "aabbcc" Output: "abc"

   Level 2: Remove duplicates but keep order of first occurrence

   - Input: "cbacdcbc" Output: "cbad"

   Level 3: Case-insensitive deduplication, maintain first casing

   - Input: "aAbBcC" Output: "abc"

3. **Implement Basic String Compression**

   (Min Level for Testing: 2
   Min Level for Development Roles: 3)

   Level 1: Compress consecutive characters

   - Input: "aaabb" Output: "a3b2"

   Level 2: Only compress if result is shorter than original

   - Input: "abc" Output: "abc"

   Level 3: Handle case-insensitive compression and retain original casing for output

   - Input: "AaAaBB" Output: "A2a2B2"

4. **Replace Characters Based on Frequency**
   (Min Level for Testing: 2
   Min Level for Development Roles: 3)

   Level 1: Replace each character with its frequency
   - Input: "aabbc" Output: "22221"

   Level 2: Return only characters that occur once
   - Input: "aabbcd" Output: "cd"

   Level 3: Replace each character with number of times it appears after it in the string
   - Input: "abab" Output: [1,1,0,0]

5. **Remove Duplicates from Array Without Using Set**

   (Min Level for Testing: 1
   Min Level for Development Roles: 2)

   Level 1: Remove duplicates from a sorted array in-place (without using additional memory)

   ○ Input: [1, 1, 2, 2, 3] → Output: [1, 2, 3]

   Level 2: Remove duplicates from an unsorted array (maintain order)

   ○ Input: [3, 1, 2, 3, 2, 4] → Output: [3, 1, 2, 4]

   Level 3: Return count of unique elements only, no extra array

    - Input: [4, 4, 5, 1, 2, 2] → Output: 3 (unique elements: 5, 1, 2)

6. **Find the Missing Number**
   (Min Level for Testing: 2
   Min Level for Development Roles: 3)

   Level 1: Find one missing number in a range 1...n
   - Input: [1, 2, 4, 5] → Output: 3

   Level 2: Find multiple missing numbers (unsorted input)
   - Input: [1, 3, 5, 6], Range 1–6 → Output: [2, 4]

   Level 3: Handle negative numbers and return missing elements from the smallest to largest
   - Input: [-2, 0, 1, 4] → Output: [-1, 2, 3]