Problem of the Week – First Missing Positive Integer

Company: Stripe
Difficulty: Hard
Topic: Arrays, Hashing, In-place Rearrangement

📌 Scenario

Stripe’s payment system deals with sequential identifiers, where missing numbers may cause serious issues. For example, suppose a sequence is supposed to have all positive integers in order, but one is missing. Stripe engineers must quickly identify the **first missing positive number** in a given list of integers.

Your task is to **find the smallest positive integer that is missing from the array**, using **linear time (O(n))** and **constant extra space (O(1))**.

📌 Problem Statement

Given an unsorted array of integers `arr[]`, return the **first missing positive integer**.

- The array may contain **duplicates** and **negative numbers**.
- You may modify the input array **in-place**.

🔹 Input Format

- First line: Integer `N` (size of the array).
- Second line: N space-separated integers (the array elements).

🔹 Output Format

- Print the smallest positive integer that is missing.

🔹 Constraints

- 1 ≤ N ≤ 10^5

- -10^9 ≤ arr[i] ≤ 10^9

🔹 Sample Input 0

```
4
3 4 -1 1
```

🔹 Sample Output 0

```
2
```

🔹 Sample Input 1

```
3
1 2 0
```

🔹 Sample Output 1

```
3
```

🔑 Approaches

1. **Naïve Approach (Sorting / Hashing)**
   - Sort the array and check missing positive.
   - Or use a HashSet.
   - Time: **O(n log n)**, Space: **O(n)** ❌ (fails requirement).
2. **Optimal Approach (Index Placement Trick – In-Place Hashing)** ✅
   - Place each number `x` in index `x-1` (only if `1 ≤ x ≤ N`).
   - After rearrangement, traverse array:
     - If `arr[i] != i+1`, return `i+1`.
   - If all positions are correct, return `N+1`.
   - Time: **O(n)**, Space: **O(1)**.

🔗 Practice Links

- LeetCode – First Missing Positive
- GeeksforGeeks – Smallest Positive Missing Number