Array #4

Leetcode #1089

Duplicate Zeros

https://leetcode.com/problems/duplicate-zeros/description/

Given a fixed length integer array arr, duplicate each occurence of zero, shifting the remaining elements to the right

NOTE that elements beyond the length of the original array are not written, Do the above modifications to the input array in place and do not return anything

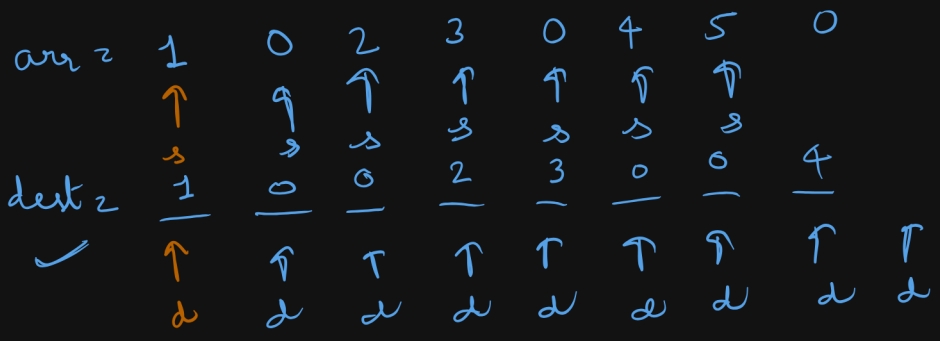
Example 1:
Input: arr = [1, 0, 2, 3, 0, 4, 5, 0]
Output: [1, 0, 0, 2, 3, 0, 0, 4]

Example 2:
Input: arr = [1, 2, 3]
Output: [1, 2, 3]

Constraints:
1 <= arr.length <= 10^4
0 <= arr[i] <= 9

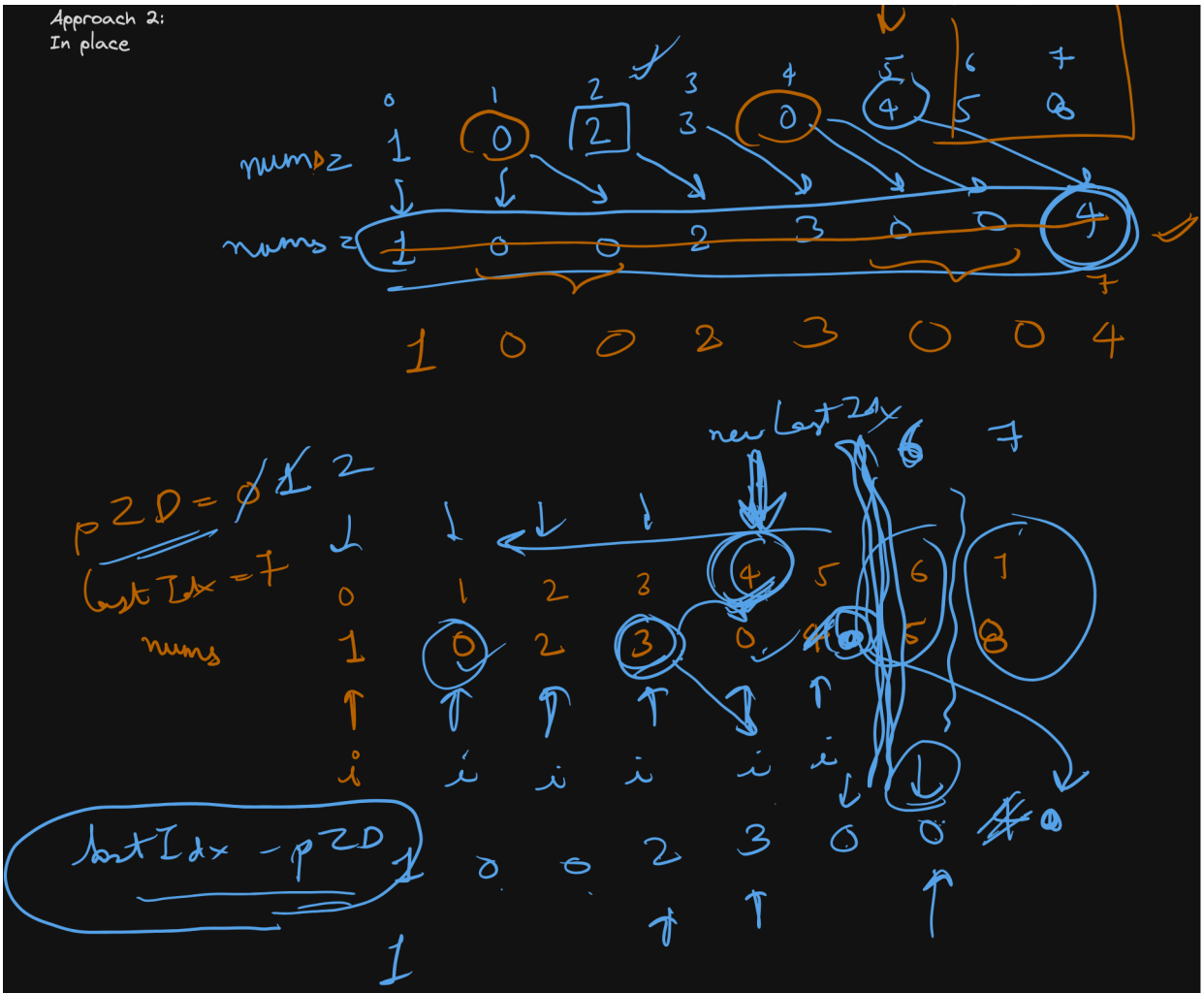
Companies:
Amazon, Google

Approach 1:
Destination array
Source array



Time complexity:
O(N)
Space Complexity:
O(N)

Approach 2:
In place



Time complexity:
O(N)
Space Complexity:
O(1)

```
class Solution {
    public void duplicateZeros(int[] arr) {
        // approach 1
        int[] dest = new int[arr.length];

        int s = 0,
            d = 0;

        while (s < arr.length) {
            if (arr[s] == 0) {
                if (d < arr.length) {
                    dest[d] = 0;
                }
                d += 1;
                if (d < arr.length) {
                    dest[d] = 0;
                }
            } else {
                if (d < arr.length) {
                    dest[d] = arr[s];
                }
            }

            d += 1;
            s += 1;
        }

        for (int i = 0; i < arr.length; i++) {
            arr[i] = dest[i];
        }
    }
}
```

```java
class Solution {
    public void duplicateZeros(int[] arr) {
        int possibleZeroDups = 0;
        int lastIdx = arr.length - 1;

        for (int i = 0; i <= lastIdx - possibleZeroDups; i++) {
            if (arr[i] == 0) {
                // edge case
                if (i == lastIdx - possibleZeroDups) {
                    arr[lastIdx] = 0;
                    lastIdx -= 1;
                    break;
                }
                possibleZeroDups++;
            }
        }

        int newLastIdx = lastIdx - possibleZeroDups;

        for (int i = newLastIdx; i >= 0; i--) {
            if (arr[i] == 0) {
                arr[i + possibleZeroDups] = 0;
                possibleZeroDups--;
                arr[i + possibleZeroDups] = 0;
            } else {
                arr[i + possibleZeroDups] = arr[i];
            }
        }
    }
}
```