Move Zeros LeetCode

Given a vector of integers, move all the zeros to the end of the vector while maintaining the relative order of non-zero elements.

Example: Input: nums = [0, 0, 4, 4, 0, 9] Output: [4, 4, 9, 0, 0, 0]

Explanation: In the given example, all the zeros are moved to the end of the vector while preserving the order of non-zero elements.

Approach 1: Moves zeros to the end of the vector while maintaining the order of non-zero elements

This approach uses two pointers, **slow** and **fast**, to traverse the vector. The **slow** pointer tracks the position to place the next non-zero element, and the **fast** pointer moves ahead to find the next non-zero element. The algorithm iterates through the vector, checking the values at **slow** and **fast** positions and performing the necessary swaps to move zeros to the end. Finally, all the zeros are moved to the end of the vector while maintaining the relative order of non-zero elements.

The Time complexity is O(n), where n is the size of the input vector.

The approach has a Space complexity of O(1), as they only use a constant amount of extra space to store the pointers and temporary variables.

Approach 2: Optimized version to move zeros to the end of the vector while maintaining the order of non-zero elements

This approach is an optimized version of the first approach. It also uses two pointers, **slow** and **fast**, to traverse the vector. However, instead of performing swaps, it directly assigns the non-zero element to the **slow** position. The algorithm iterates through the vector, and when a non-zero element is found, it assigns it to the **slow** position and increments **slow**. Finally, the remaining positions from **slow** to the end of the vector are filled with zeros.

The Time complexity is O(n), where n is the size of the input vector.

The approach has a Space complexity of O(1), as they only use a constant amount of extra space to store the pointers and temporary variables.