Push zeros to end

Problem Description: Given a random integer array, push all the zeros that are present to the end of the array. The respective order of other elements should remain the same.

Sample Input:

```
7                          (Size of array)
2 0 4 1 3 0 28             (Elements of array)
```

Sample Output:

```
2 4 1 3 28 0 0
```

How to approach?

Before we discuss the approach for this question let's see what exactly the question requires us to do. It seems that we have to push all the 0s in the array towards the end of the array. It can also be looked at as pushing all the non-zero elements in the array towards the beginning of the array.

So let's use a two pointer approach to solve this problem. We'll maintain two pointers, 'current' and 'nonZeroPos'. 'current' will be used to iterate through the array and 'nonZeroPos' will be used to decide the next position where the next non zero element will go to. Both pointers will be initialised with 0.

Now, we'll iterate through the array. If we encounter a 0, we'll just increase 'current' by 1. However, if we encounter a non-zero value, we put that element to 'nonZeroPos' and bring 'nonZeroPos's element to the current index. Basically we're doing `swap(arr[current], arr[nonZeroPos])`. And after this, we'll increase both 'current' and 'nonZeroPos' by 1. This will ensure that every non-zero element gets pushed towards the front of the array with their order maintained.

The pseudo-code for this approach is shown on the next page.

```
function pushZeroesToEnd(arr):
  current <- 0
  nonZeroPos <- 0
  while(current < arr.size):
    if(arr[current] != 0):
      swap(arr[current], arr[nonZeroPos])
      nonZeroPos <- nonZeroPos + 1
    current <- current + 1

  //print or return the array according to requirement
```