Partitioning Numbers in an Array (or Vector/ArrayList)

Dr. Byun
03/14/2022

This document presents two different approaches to **put all negative numbers in front of all positive numbers in an array**. Note that we are not trying to sort the numbers. Instead, we try to put all negative numbers first. The sequence among the negative numbers is not important. Similarly, the sequence among the positive numbers is not important as well.

To explain the approaches, we use an array with 8 numbers like below.

5, -3, 1, -9, -8, 2, -4, 7

1. First Approach

In this approach, you use two indexes i and j. The index i points to the first number (= 5) and the index j points to the last number (= 7).

```
i                          j
5, -3, 1, -9, -8, 2, -4, 7
```

Since the index i points to 5 which is a positive number, the index i should stop. Then, check the value of the index j. Since the index j points to a positive number (= 7), we move the index to the left.

```
i                       j
5, -3, 1, -9, -8, 2, -4, 7
```

The index j now points to a negative value (= -4). Thus, we stop the index j. After that, we swap the values of indexes i and j.

```
 i                      j
-4, -3, 1, -9, -8, 2, 5, 7
```

Now, we should resume the operations for the indexes i and j. In other words, we check the value of the index i which points to a negative value (= -4). Since this is a good situation because a negative number is in front, we move the index i to the right.

```
    i                   j
-4, -3, 1, -9, -8, 2, 5, 7
```

Because the index i points to -3 (= a negative number), this is also fine. Thus, we move the index i to the right one more time.

```
          i          j
-4, -3, 1, -9, -8, 2, 5, 7
```

Since the index i points to 1 (= a positive number), the index i should stop. Then, we should check the value of the index j. Since the index j points to 5, we move the index j to the left.

```
          i       j
-4, -3, 1, -9, -8, 2, 5, 7
```

Because the index j points to 2 (= a positive number), we move the index j to the left one more time.

```
          i    j
-4, -3, 1, -9, -8, 2, 5, 7
```

The index j now points to -8 (= a negative value) and we stop the operation of the index j. Then, we swap the values of indexes i and j.

```
          i    j
-4, -3, -8, -9, 1, 2, 5, 7
```

Now, we resume the operations for the indexes i and j again. Since the value of the index i is a negative value (= -8), we move the index i to the right.

```
               i  j
-4, -3, -8, -9, 1, 2, 5, 7
```

Since the idex i points to -9 (= a negative number), we move it to the right one more time.

```
                  i j
-4, -3, -8, -9, 1, 2, 5, 7
```

Since the index i points to 1 (= a positive number), the index i must stop. For the index j, it should move to the left side because it points to 1 (= a positive number).

```
               j  i
-4, -3, -8, -9, 1, 2, 5, 7
```

Since the index j now points to -9, we must stop the operation of the index j. Note that **the index i is greater than the index j** (= crossover). Therefore, we should not swap

the values of indexes i and j because the whole operation is over. All negative numbers (= -4, -3, -8, -9) are in front of all positive numbers (= 1, 2, 5, 7).

2. Second Approach

In this approach, two indexes i and j start from the first number in an array. The index j scans all numbers of an array one by one from left to right. Meanwhile, the index i holds the position of the first positive number of the array by moving from left to right. If the index j points to a negative number while scanning the array, the values of the index i and j are swapped.

This is the initial configuration of the indexes i and j.

```
i
5, -3, 1, -9, -8, 2, -4, 7
j
```

Since the index j points to 5 (= a positive number), it moves to the right. Note that the index i doesn't move because it points to a positive number.

```
i
5, -3, 1, -9, -8, 2, -4, 7
    j
```

Since the index j now points to -3 (= a negative value), we should swap the values of indexes i and j. Remember that we want to put all negative numbers in front. Thus, we should swap the values when the index j meets a negative number.

```
i
-3, 5, 1, -9, -8, 2, -4, 7
     j
```

After the swapping, we move the indexes i and j to the right. We move the index i to the right because the index i must point to the first positive number of the current array. We move the index j to the right because we have to continue scanning the array.

```
      i
-3, 5, 1, -9, -8, 2, -4, 7
         j
```

Now, because the index j points to 1 (= a positive number), we move the index j to the right one more time.

```
      i
-3, 5, 1, -9, -8, 2, -4, 7
         j
```

Since the index j points to a negative number (= -9), we swap the values of indexes i and j.

```
      i
-3, -9, 1, 5, -8, 2, -4, 7
         j
```

After that, we move the indexes i and j to the right. Again, we move the index i to the right because the index i must point to the first positive number of the current array. We move the index j to the right because we have to continue scanning the array.

```
         i
-3, -9, 1, 5, -8, 2, -4, 7
            j
```

Now, since the index j points to a negative number (= -8), we should swap the values of indexes i and j.

```
         i
-3, -9, -8, 5, 1, 2, -4, 7
            j
```

After the swapping, we move the indexes i and j to the right.

```
            i
-3, -9, -8, 5, 1, 2, -4, 7
               j
```

Because the index j now points to a positive number (= 2), we move it to the right.

```
            i
-3, -9, -8, 5, 1, 2, -4, 7
                  j
```

Since the index j points to a negative number (= -4), we swap the values of indexes i and j.

```
          i
-3, -9, -8, -4, 1, 2, 5, 7
              j
```

After that, we move the indexes i and j to the right.

```
             i
-3, -9, -8, -4, 1, 2, 5, 7
                 j
```

Finally, the index j points to the last number which is a positive number (= 7). Because the index j reaches to the end of the array, the whole operation is over. All negative numbers (= -3, -9, -8, -4) are in front of all positive numbers (= 1, 2, 5, 7).

Exercise

Perform the partitioning operation for the following numbers using the two approaches.

-4, 3, 9, -6, 2, -5, 8, 7

Do not look at the answer immediately. Try to solve it by yourself.

When you're done, check here for the solution.