

Dutch National Flag Problem (medium)

We'll cover the following ^

- Problem Statement
- Try it yourself
- Solution
 - Code
 - Time complexity
 - Space complexity

Problem Statement

Given an array containing 0s, 1s and 2s, sort the array in-place. You should treat numbers of the array as objects, hence, we can't count 0s, 1s, and 2s to recreate the array.

The flag of the Netherlands consists of three colors: red, white and blue; and since our input array also consists of three different numbers that is why it is called Dutch National Flag problem (https://en.wikipedia.org/wiki/Dutch_national_flag_problem).

Example 1:





Input: [1, 0, 2, 1, 0]
Output: [0 0 1 1 2]

Example 2:





Input: [2, 2, 0, 1, 2, 0]
Output: [0 0 1 2 2 2]

Try it yourself

Try solving this question here:

 Java	 Python3	 JS	 C++
--	---	--	---

```
1 def dutch_flag_sort(arr):
2     # TODO: Write your code here
3     return
4
```









Solution

The brute force solution will be to use an in-place sorting algorithm like Heapsort (<https://en.wikipedia.org/wiki/Heapsort>) which will take $O(N * \log N)$. Can we do better than this? Is it possible to sort the array in one iteration?

We can use a **Two Pointers** approach while iterating through the array. Let's say the two pointers are called `low` and `high` which are pointing to the first and the last element of the array respectively. So while iterating, we will move all 0s before `low` and all 2s after `high` so that in the end, all 1s will be between `low` and `high`.

Code #

Here is what our algorithm will look like:

 Java	 Python3	 C++	 JS
<pre>1 def dutch_flag_sort(arr): 2 # all elements < low are 0, and all elements > high are 2 3 # all elements from >= low < i are 1 4 low, high = 0, len(arr) - 1 5 i = 0 6 while(i <= high): 7 if arr[i] == 0: 8 arr[i], arr[low] = arr[low], arr[i] 9 # increment 'i' and 'low' 10 i += 1 11 low += 1 12 elif arr[i] == 1: 13 i += 1 14 else: # the case for arr[i] == 2 15 arr[i], arr[high] = arr[high], arr[i] 16 # decrement 'high' only, after the swap the number at index 'i' could be 0, 1 or 2 17 high -= 1 18 19 20 def main(): 21 arr = [1, 0, 2, 1, 0] 22 dutch_flag_sort(arr) 23 print(arr) 24 25 arr = [2, 2, 0, 1, 2, 0] 26 dutch_flag_sort(arr) 27 print(arr) 28</pre>			
<div></div>			

Time complexity #


The time complexity of the above algorithm will be $O(N)$ as we are iterating the input array only once.

Space complexity #

The algorithm runs in constant space $O(1)$.

☒ Mark as Completed

 Report an Issue

 Ask a Question

(https://discuss.educative.io/tag/dutch-national-flag-problem-medium__pattern-two-pointers__grokking-the-coding-interview-patterns-for-coding-questions)