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.

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++

```
def dutch_flag_sort(arr):
  # TODO: Write your code here
  return

```

Test | Save | Reset

Solution

The brute force solution will be to use an in-place sorting algorithm like Heapsort which will take $O(N * logN)$. 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

```python
def dutch_flag_sort(arr):
  # all elements < low are 0, and all elements > high are 2
  # all elements from >= low < i are 1
  low, high = 0, len(arr) - 1
  i = 0
  while(i <= high):
    if arr[i] == 0:
      arr[i], arr[low] = arr[low], arr[i]
      # increment 'i' and 'low'
      i += 1
      low += 1
    elif arr[i] == 1:
      i += 1
    else:  # the case for arr[i] == 2
      arr[i], arr[high] = arr[high], arr[i]
      # decrement 'high' only, after the swap the number at index 'i' could be 0, 1 or 2
      high -= 1


def main():
  arr = [1, 0, 2, 1, 0]
  dutch_flag_sort(arr)
  print(arr)

  arr = [2, 2, 0, 1, 2, 0]
  dutch_flag_sort(arr)
  print(arr)


main()

```

Run | Save | Reset

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 $O(1)$.

Back
Subarrays with Product Less than a Ta...

Next
Problem Challenge 1

Completed

Report an Issue | Ask a Question