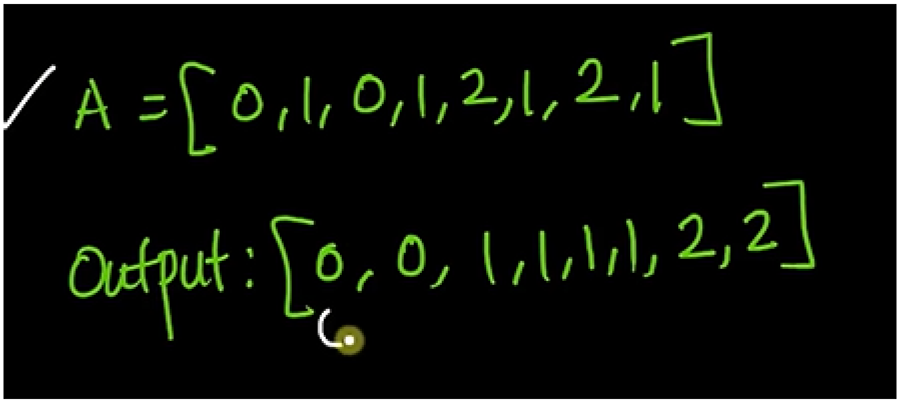
Dutch National Flag Problem

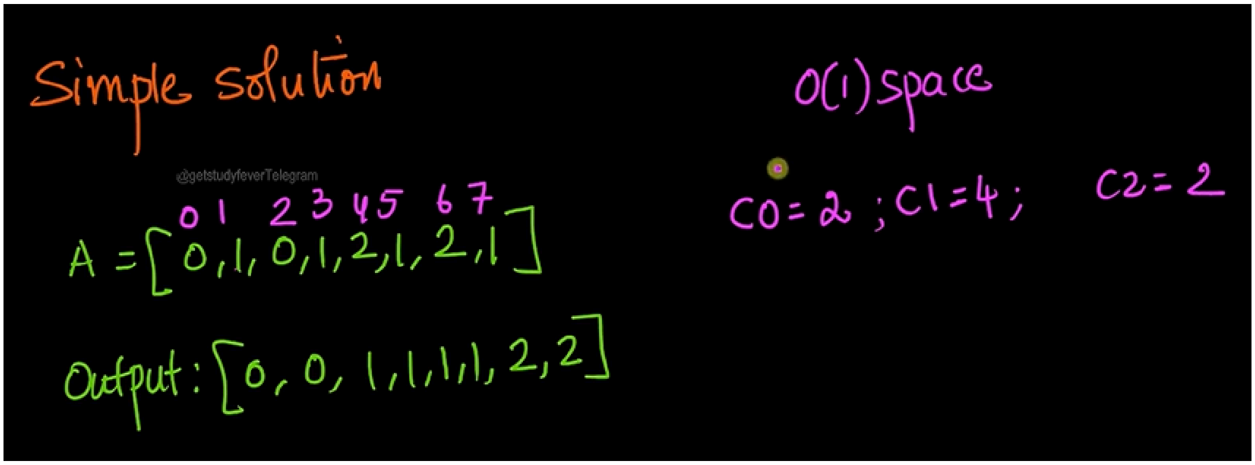
1. Question 

   A = [0,1,0,1,2,1,2,1]

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

2. Given this array as input…the output must be in the order as shown in pic

SOlution:

1. So here if we use any comparison sorting algo(merge,quick,etc)..then it takes areoung O(nlogn) time complexity
2. Here we'll be using bucket sort or counting sort algo
3. We'll be using 3 variables here…as there are only 3 variety of numbers
4. 

   Simple solution

   O(1) space

   0 1 2 3 4 5 6 7

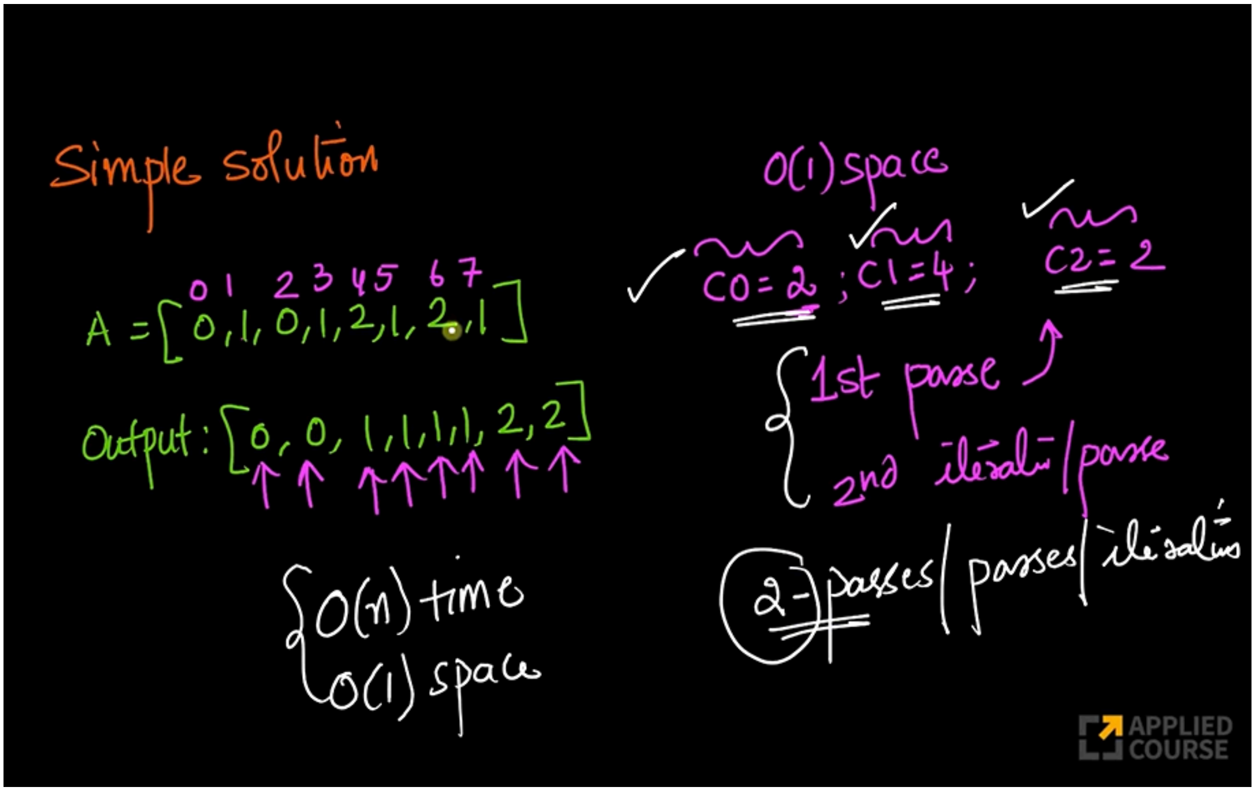
   A = [0,1,0,1,2,1,2,1]

   C0 = 2 ; C1 = 4 ; C2 = 2

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

5. Now in the first iteration..c0 stores the count of 0's….c1 stores the count of 1's and c2 stores the count of 2's
6. It is similar to counting sort algo…and here the time complexity is O(n) and space complexity is O(1)

7. In this solution we are using 2 passes



Simple solution

$A = [\overset{0}{0}, \overset{1}{1}, \overset{2}{0}, \overset{3}{1}, \overset{4}{2}, \overset{5}{1}, \overset{6}{2}, \overset{7}{1}]$

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

$\begin{cases} O(n) \text{ time} \\ O(1) \text{ space} \end{cases}$

O(1) space

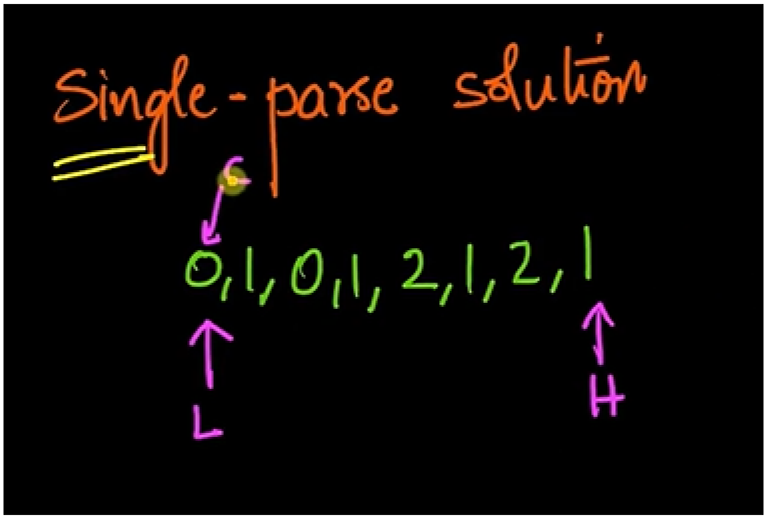
$C0 = 2$; $C1 = 4$; $C2 = 2$

$\begin{cases} \text{1st passe} \\ \text{2nd iteration/passe} \end{cases}$

2-passes/passes/iterations

8. Can we solve it in one pass?

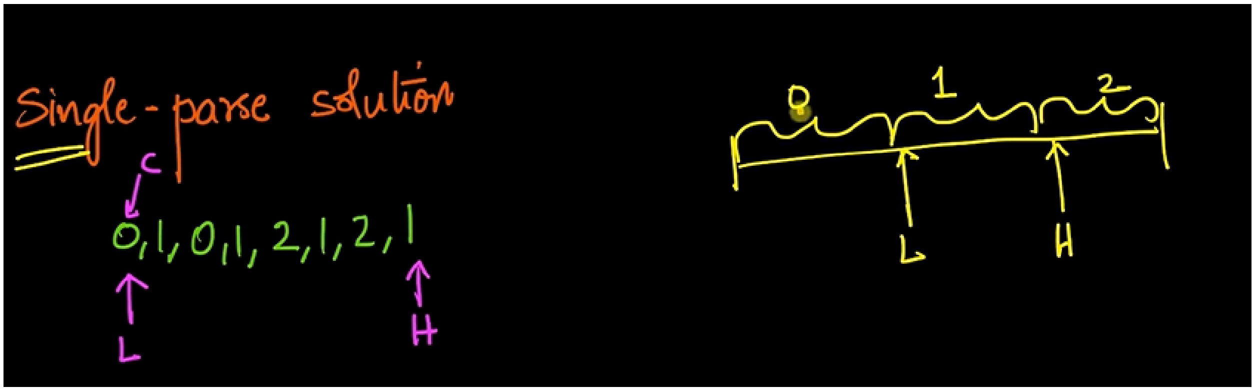
Single Pass Solution

1. 

Single-parse solution

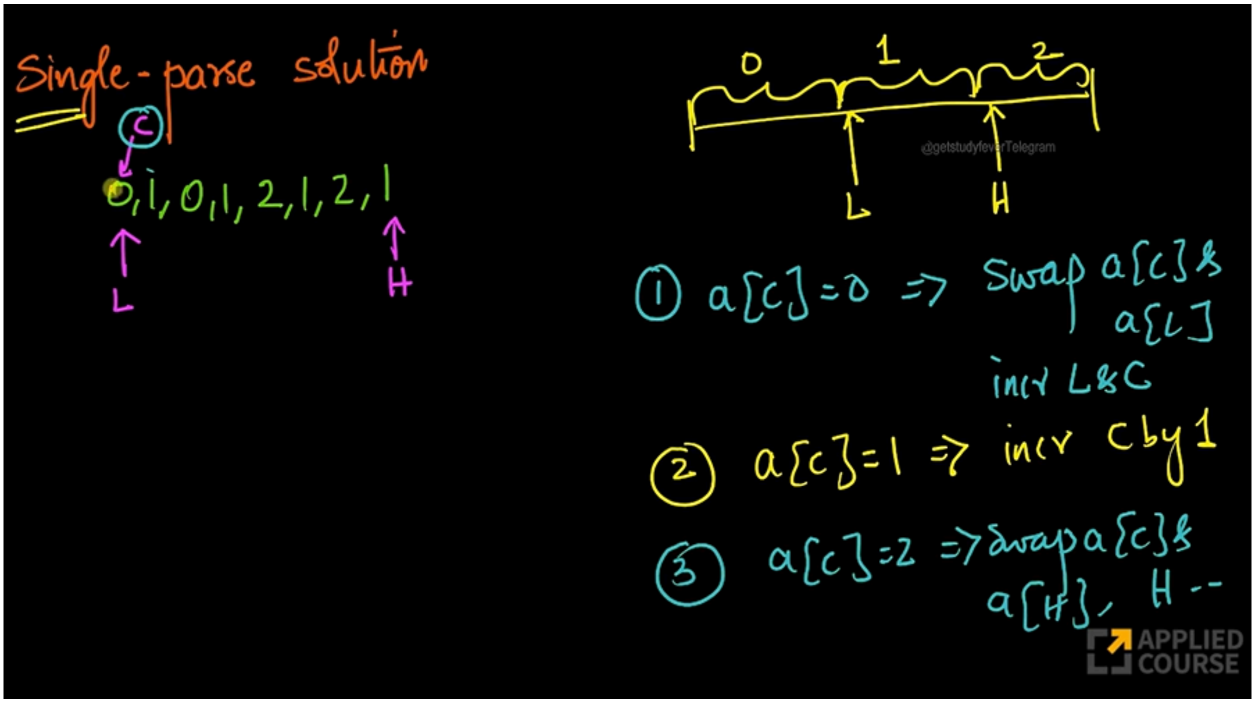
0, 1, 0, 1, 2, 1, 2, 1

2. So here we'll be using 3 pointers



Single-parse solution

3. L denotes the end of 0's and H denotes the start of 2's
4. This is the algo…we are implementing



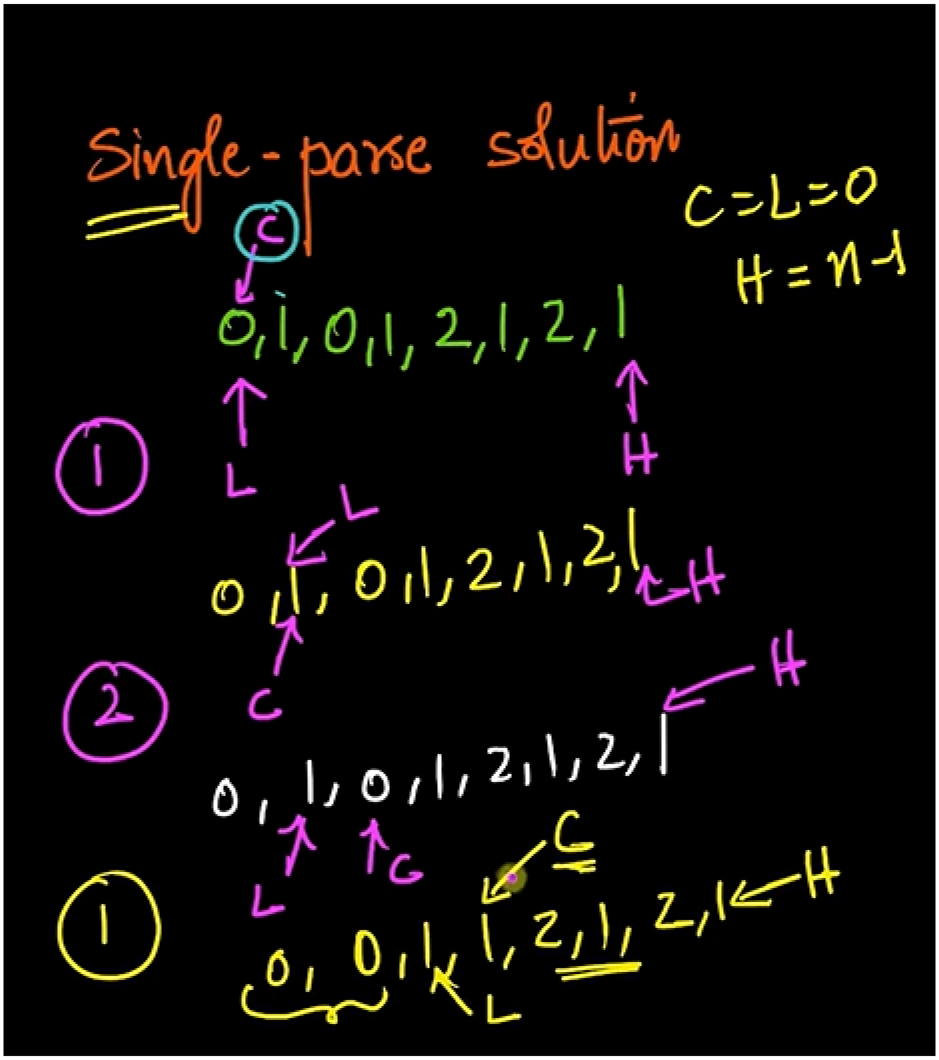
Single-parse solution

① a[c]=0 => Swap a[c] & a[L]
incr L&C

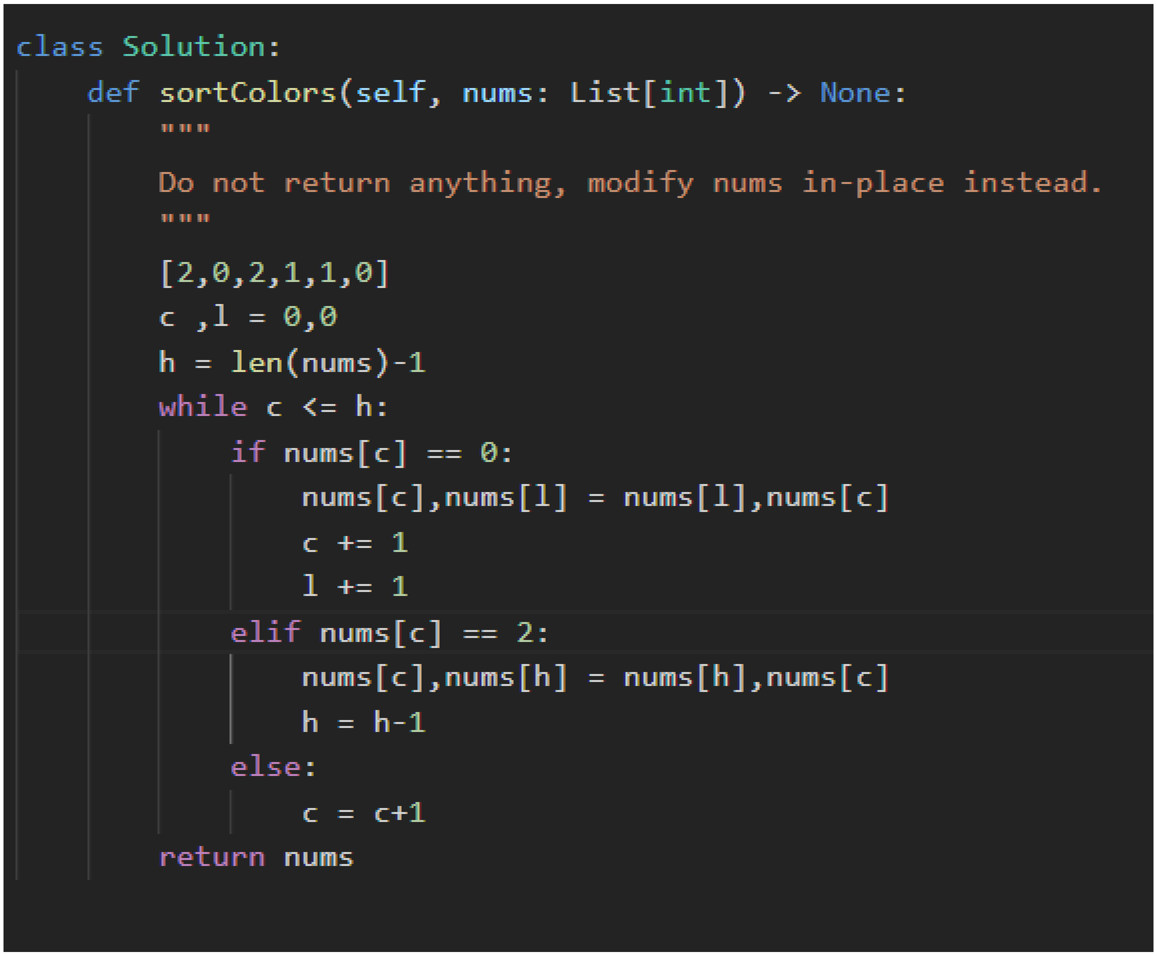
② a[c]=1 => incr c by 1

③ a[c]=2 => swap a[c] & a[H], H--

5. Now we'll iterate the array…and implement the above rules

6. 

Python Solution

1. 

```
class Solution:
    def sortColors(self, nums: List[int]) -> None:
        """
        Do not return anything, modify nums in-place instead.
        """
        [2,0,2,1,1,0]
        c ,l = 0,0
        h = len(nums)-1
        while c <= h:
            if nums[c] == 0:
                nums[c],nums[l] = nums[l],nums[c]
                c += 1
                l += 1
            elif nums[c] == 2:
                nums[c],nums[h] = nums[h],nums[c]
                h = h-1
            else:
                c = c+1
        return nums
```