

sort color

R W B
0 1 2

nums = [2,0,2,1,1,0]

0 0 1 1 2 2

- -

.

j == 1

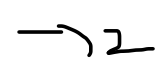
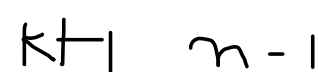
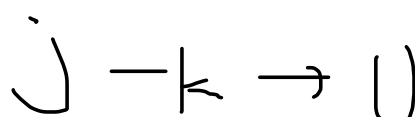
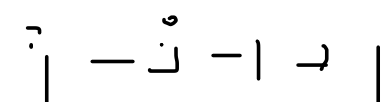
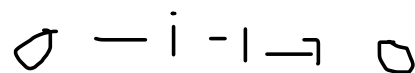
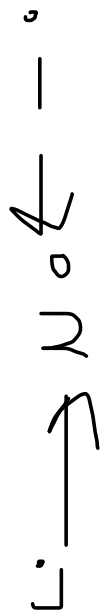
--> j++

j == 0

swap(i, j)
i++; j++

j == 2

swap(j, k)
k--



Move Zero's

0 1 2 3 4 5 6 7

1 8 7 3 0 0 6 0

↓
i

↓
j

1 8 7 3 0 0 0 0

0 - i - 1

→ n - 1

i - j - 1
→ 0

j → n - 1
→ 0

remove-duplicates-from-sorted-array/

O/P

0 0 1 1 1 2 2 3 3 4

$\textcircled{k} \rightarrow \text{let}$
Uniqw

I | P

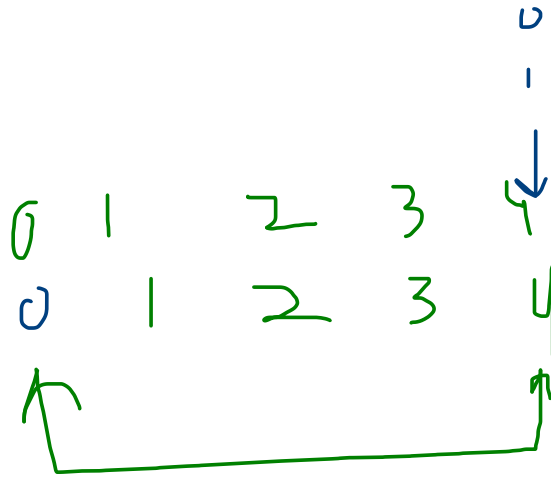


0 1 2 3 4

k unique elements

2 2 3 3 4

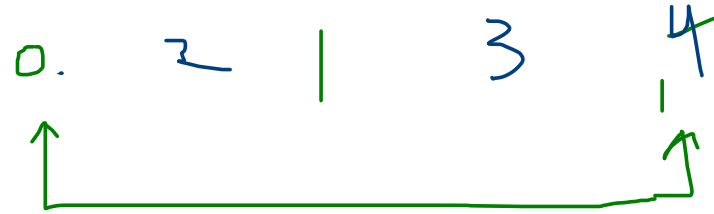
its ok if it is not as per input order



Unieke Elements

```
if (arr[i] == arr[j])
j++
```

```
if (arr[i] != arr[j])
arr[i+1] = arr[j]
i++; j++
```



Longest Substring Without Repeating Characters

Garbage

0-i

→ unique ele

$i+1 \rightarrow j-1$
garbage

$j - n-1$
unknown



Longest Substring Without Repeating Characters

0 1 2 3 4 5 6 7 8 9 10
a b c a b a c d e b c

↓ ↓
 s_i e_i

$s_i - e_i$
→ unique ele

Map char --> last index of char

a → 3
b → 4
c → 2
d → -1
e → -1

len of unique region = $e_i - s_i + 1$

maxUniqueLen = 3

($s_i = 4, 2$)

if char is not in map

1. cal len of unique region
- update
2. maxUniqueLen
3. update map
4. e_i++

if (char is present in map

$s_i = \max(s_i, (\text{map}[\text{ch}] + 1))$

1. cal len of unique region
- update
2. maxUniqueLen
3. update map
4. e_i++