Sort Color

use three pointer

1. left pointer: the first element not equals to 0

2. right pointer: the first element from right not equals 2

3. index: iterate from left to right.

There could be three case for left pointer:

0/1/2

if index pointer is 0, then swap needed, and left and index will move right.

  because we are scanning from right, so there is no 2 before index, so after swap, we can safely move right index. But swap with right not the same case

if index pointer is 1, then index move to right and look for next element

if index pointer is 2, then swap with right

  here could be two cases:

    current index is 0, then swap with left and move right

    current index is 1, then no swap needed, move right

| 2 | 1 | 0 | 0 | | 1 | 1 |

left

right

left
index

2	0

right

left
index

| 0 | 1 | 0 | 2 |

do not move index
because the new element could be 0

left
index

right

| 1 | 1 |

left   index

| 0 | 1 | 0 | 2 |

left
index

right

| 1 | 1 |

left

| 0 | 1 | 0 | 2 |

left
index

right

| 1 |

left

| 0 | 1 | 0 | 2 |

left

right
index

| 0 |

| 0 | 0 | 1 | 2 |

exit condition must index <= right

right
left

index

2	0

↑
right

2	0

↑ index

↑ right

1	0	2

↑
index
right

1	1	2

↑ left

↑ right

↑ index