Rotate Right

n=7

```
        0  1  2  3  4  5  6
arr = [ 1  2  3  4  5  6  7 ]
```

Rotate right by 1.

arr = [7 1 2 3 4 5 6

Rotate right by 2

arr = [6 7 1 2 3 4 5]

Two Pointer.

↳ Reverse the array

[7 6 5 4 3 2 1]

[7 1 2 3 4 5 6]

[7 1 2 3 4 5 6]

[1 2 3 4 5 6 7]

↳ Reverse array

[7 6 5 4 3 2 1]

Rotate by 2.

[6 7 1 2 3 4 5]

Example.

[1 2 3 4 5 6 7]

Reverse array

[7 6 5 4 3 2 1]

Reverse first 3 elements

[5 6 7 1 2 3 4]

We can reverse using two pointer.

1. left 2. right

To reverse entire array

left = 0, right = n-1

Then reverse first K elements

left = 0, right = K-1,

reverse , elements after K

left = K, right = n-1

$K = 2$, $n = 7$

$K = K \% n$

$= 2 \% 7 = \underline{2}$.

$K = 9$

$= 9 \% 7 = \underline{2}$.

Brute Force Approach

$$\text{arr} = \begin{array}{c} \\ \\ \end{array}\begin{array}{ccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array}$$

temp[] 0 1 2 3 4 5 6

= [7 1 2 3 4 5 6]

0 1 2 3 4 5 6

arr = [1 2 3 4 5 6 7]

0 1 2 3 4 5 6

temp = [6 7 0 1 3 4 5

6+2 = 8 = 1

5+2 = 7 = 0

temp[i+k%n] = arr[i]

```
for(int i=0; i<n; i++) {
    temp[i+k%n] = arr[i];
}

for(int i=0; i<n; i++){
    arr[i] = temp[i];
}
```

Zeroes and Ones

n=6

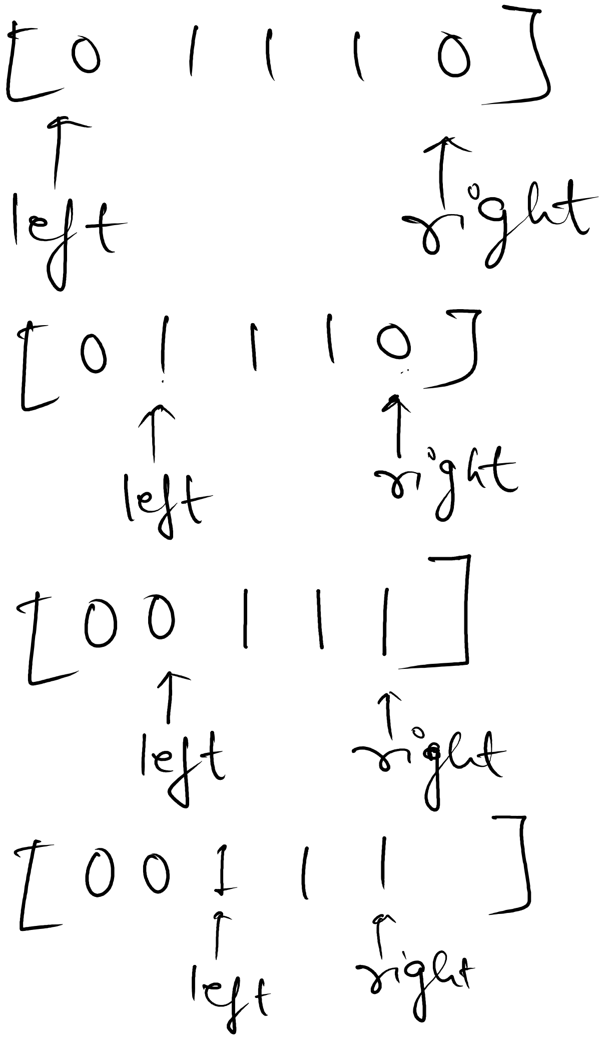
arr=[0 1 1 1 0]

left=0;

right=n-1;

increment left until arr[left] is not 1

decrement right until arr[right] is not 0



```
[0 0 1 1 1]
     ↑ ↑
  left right
```

Example2.

```
[0 1 0 1 0 1]
 ↑         ↑
left      right
```

```
[0 1 0 1 0 1]
   ↑     ↑
  left  right
```

```
[0 0 0 1 1 1]
   ↑       ↑
  left    righ
```

```
[0 0 0 1 1 1]
     ↑ ↑
  left right
```

Code.

```
int left=0, right=n-1;
```

```
while (left < right) {
   while(arr[left]==0 && left<right)
        left++;
   while(arr[right]==1 && (left<right)
       right--;
   int temp = arr[left];
   arr[left] = arr[right];
   arr[right] = temp;
    left++;
   right--;

  }
}  Time Complexity -> O(n/2)
```

Brute Force Approach.

```
   int countzero=0

 for (int i=0; i<n; i++) {
    if (arr[i]==0)
       countzero++;
 }
```

```
}
for (int i=0; i<countzero; i++){
      arr[i]=0;
   }
for(int i=countzero ; i<n; i++){
         arr[i]=1;
      }
```

Time Complexity → O(n)

Sorting Approach (Time Complexity - O(nlogn))

↳ Do not use in case of zero one array