Introduction to Arrays

Space Complexity

↳ max space reqd. to run algo.

```
func(int N) {   // 4B
    int x;      // 4B
    int y;      // 4B
    long z;     // 8B
}
```

$\rightarrow 4+4+8 = 16B$

$\Rightarrow O(1)$

SC → Auxilliary space taken
extra

$O(10^5) \rightarrow O(1)$

Quiz1

```
func(int N) {
    int a[10];             // 40B
    int x, y, z;           // 4*3B
    int[] a = new int[N];  // 4*N B
}
```

$SC = 40+12+4N$

$\Rightarrow O(N)$

```
func (int N){

    . . . .

    int[] arr = int [N];

    long[][] arr = long [N][N] ;

}
```

$SC = N + N^2$

$= O(N^2)$

```
for (i=0; i<n; ++i) {

    int x = i;

}
```

$\Rightarrow O(1)$ SC

$\boxed{\cancel{0}1}$

x

```
int maxArr (int a[], int n ){
    int am = a[0];
    for ( i = 1 to n-1 ){
        am = max(am, a[i]);
    }
    return am
}
```

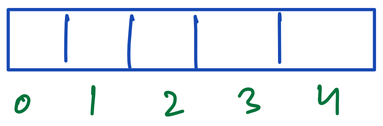
$SC = O(1)$

Array

Sequential collection of same type of data.

int arr[N];

N = 5



0, 1, 2, ... N-1
N elements

TC of accessing any value of array
= O(1)

arr[i] ⇒ O(1)

TC to access all elements of array
= O(N)

```
func printArray(a[], n){
    for(i=0 to n-1)
        print(a[i])
}
```

⇒ O(N) time
O(1) space

Question 1

Given an array arr of size N, reverse it.
w/o extra space.

eg $N = 5$

$[1, 2, 3, 4, 5]$
0 1 2 3 4

$\xrightarrow{\text{reverse}}$ $[5, 4, 3, 2, 1]$
0 1 2 3 4

0th → 4th
1st → 3rd
2nd → 2nd
3rd → 1st
4th → 0th

swap

0th ⟷ 4th
swap

$a[0] \longleftrightarrow a[n-1]$ $0 + n-1 = n-1$

$a[1] \longleftrightarrow a[n-2]$ $1 + n-2 = n-1$

⋮

$a[i] \longleftrightarrow a[j]$ $i + j = n-1$

$\downarrow$
$n-1-i$ $j = n-1-i$

```
for (i=0; i<n; ++i) {
    swap(a[i], a[n-1-i]);
}
```

NOT WORK

a =	1	2	3	4
i=0	4	2	3	1
i=1	4	3	2	1
i=2	4	2	3	1
i=3	1	2	3	4

```
i=0, j=n-1
while (i<j) {
    swap(a[i], a[j]);
    i++
    j--
}
```

iteration = N/2

TC : O(N)

SC : O(1)

```
for (i=0; i<n/2; ++i) {
    swap(a[i], a[n-1-i]);
}
```

```
swap( int x, int y ){
    int t = x;
    y = x;
    x = t;
}
```

Question 2

Given an array arr and integers l and r.

Reverse array from l to r. $l<r$

$N=5$ $a=[1,2,3,4,5]$

$\overset{\uparrow}{l=1}$ $\overset{\uparrow}{r=1}$

$l=1$, $r=3$

$ans=[1\ 4\ 3\ 2\ 5]$

```
i=l, j=r
while (i<j){
    swap(a[i], a[j]);
    i++
    j--
}
```

TC: $O(N)$ $O(r-l)$

SC: $O(1)$

BREAK: ~~10:02~~ - 10:12

Question 3

Given an array of size N. Rotate the array from left to right 'K' times.

clockwise

$\boxed{K=3}$

eg N=5

	1	2	3	4	5
	0	1	2	3	4
K=1	5	1	2	3	4
K=2	4	5	1	2	3
K=3	3	4	5	1	2

```
func rotateK ( a[], n, K ) {

    for ( i=0; i<K; ++i ) {  → O(K)
        temp = a[n-1];
        for ( j=n-1; j>0; --j ) {  → N-1 times O(N)

            a[j] = a[j-1];

        }
        a[0] = temp;

    }
}
```

TC: $O(K*N)$

SC: $O(1)$

$\boxed{K=3}$

$N=5$

1	2	3	4	5
0	1	2	3	4

3	4	5	1	2
0	1	2	3	4

$(0+\overset{K}{3})\%5 = 3$

$(1+3)\%5 = 4$

$(2+3)\%5 = 0$

$(3+3)\%5 = 1$

$(4+3)\%5 = 2$

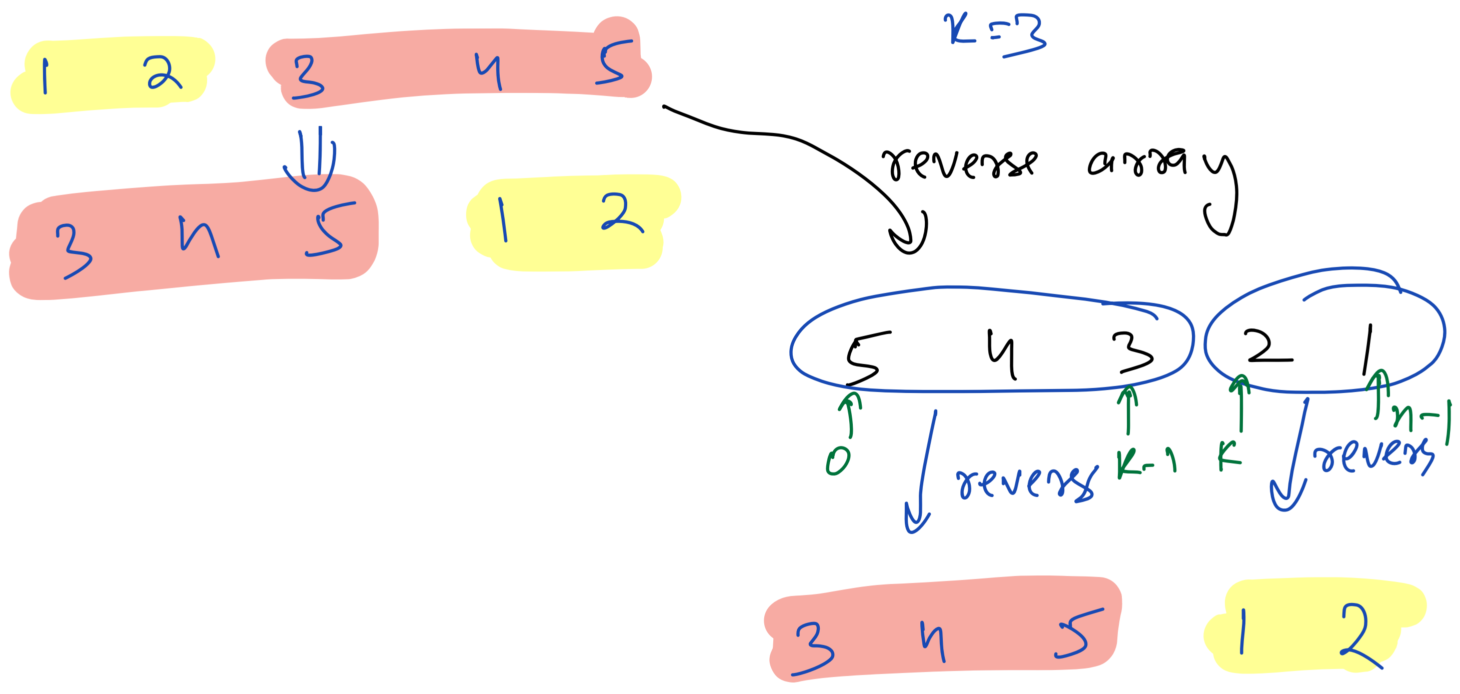
```
int temp[n]

for (i=0; i<n; ++i) {
        j= (i+K) % n
        temp[j] = a[i]
}

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

2n iterations

TC : O(N)

SC : O(N)

Final approach



To shift elements K times clockwise $\Rightarrow$

	iteration
reverse (a , 0 , n-1)	$N/2$ O(N)
reverse (a , 0 , K-1)	$K/2$ O(N)
reverse (a , K , n-1)	$\frac{N-K}{2}$ O(N)

total iteration $= \frac{N}{2} + \frac{\cancel{K}}{2} + \frac{N-\cancel{K}}{2} = N$

TC : O(N)

N + N + N = O(N)

SC : O(1)

```
void reverse( a[], l, r) {
    i= l, j= r
    while ( i<j ){
        swap( a[i], a[j] );
        i++
        j--
    }
}
```

Edge case: $K \geq n$

$K = K \% n$

Dynamic arrays

- random access O(1) like array
- variable size

Java	C++	Python	JS/Ruby
Arraylist	vector	list	array

Doubt

```
for ( — . . . . ) {
    for ( - . . . ) {
        . . . → O(1)
    }
}
```

$O(n^2)$

```
for( . . .. ) {
    func();
}
```

$n \times n = O(n^2)$

```
func() {
    for( . .. ) {
    }
}
```

a[log(n)]