Introduction to Arrays

(In class)

1.

```
void fun(int N){
    int x = N;
    int y = 2*x;
    long z = x+y;
}
```

→ Space Complexity

$x = 4B$

$y = 4B$ ⇒ $\leq$ space = 16B

$z = 8B$

⇒ O(1)

2.

```
func(int N){ // 4 bytes
    int arr[10]; //40 bytes
    int x; // 4 bytes
    int y; // 4 bytes
    long z; // 8 bytes
    int[] arr = new int[N]; // 4*N bytes
}
```

⇒ $a = 40B$

$x = 4B$

$y = 4B$ ⇒ $\leq$ space = $4N + 56$

$z = 8B$

$arr = 4 \times N\ B$ ⇒ O(N)

3.

```
void fun(int N){
    int x = N;
    int y = x*x;
    long z = y+y;
    int[] arr = new int[N];
    long[][] b = new long[N][N];
}
```

→ $x = 4B$

$y = 4B$

$z = 8B$

$arr = 4N\ B$

$b = 8 \times N \times N\ B = 8N^2\ B$

$\leq$ space = $8N^2 + 4N + 16$ ⇒ $O(N^2)$

4.

```
int maxArr( int arr[], int N){
    int ans = arr[0];
    for(i=0; i<N; i++){
        ans = Max(ans, arr[i]);
    }
    return ans;
}
```

i/p space

o/p space.

→ We need to calculate how much EXTRA space

→ Hence the space taken by input parameter variables is not accounted for

ans = 4B ⇒ O(1)

Rotate An Array

Given an array of size N & an integer K. Rotate arr[] by K.

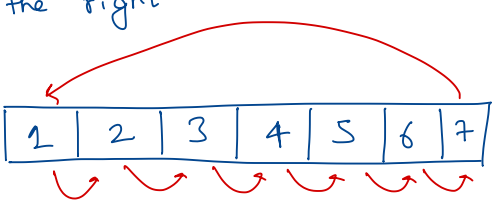
$1 \le N \le 10^6$

$0 \le K \le 10^9$

N = 7

1	2	3	4	5	6	7

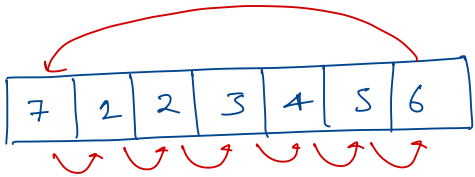
for K = 1 → The last element 7 needs to come up-front, and the rest of the array needs to shift one space towards the right



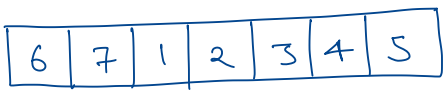
⇒

7	1	2	3	4	5	6

for k = 2 → Now 6 needs to come up front and shift the rest



⇒



You get the idea, so in the Brute Force approach, let's devise an algorithm which achieves the same

Step 1 → Save the value of last element

Step 2 → Shift the values of the initial elements 1 place to right

Step 3 → Set the value of the first element with the saved value of the last element

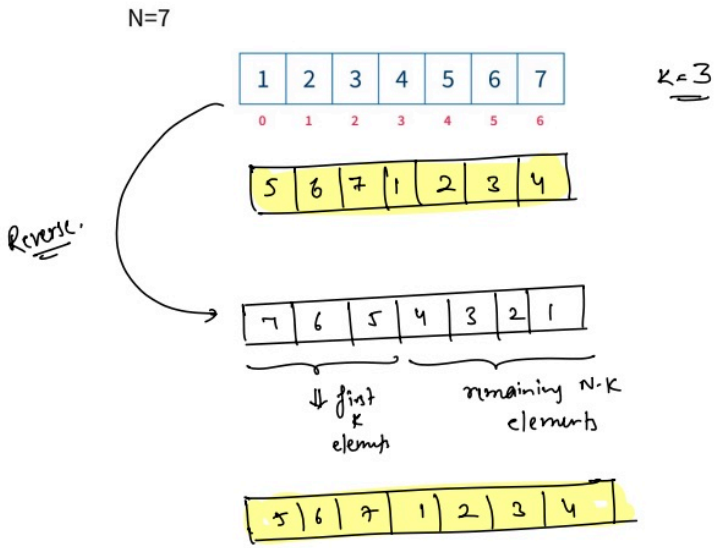
```
int length = arr.length;

// * Rotating array 'K' times
for (int i = 0; i < K; i++) {
    int last = arr[length - 1];
    for (int j = length - 2; j >= 0; j--) {
        arr[j + 1] = arr[j];
    }
    arr[0] = last;
}
```

→ Storing ahead

→ Saving current index value into the index after

Optimisation



0. $K = K \% N$
1. Reverse the whole array
2. Reverse first K elements.
3. Reverse remaining N-K elements