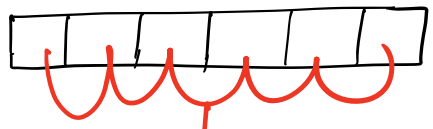
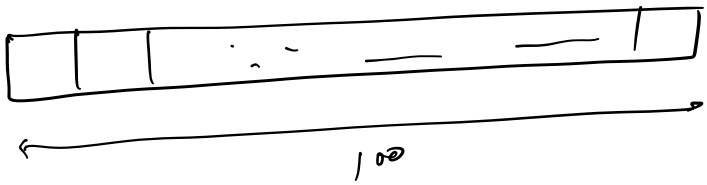
Array

Linear D.S. →



type is all same!

A

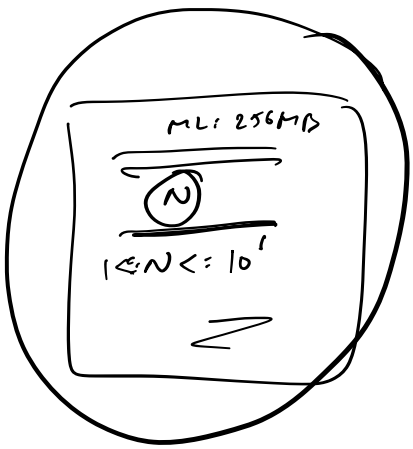



```
int A[100];
```

datatype Name Size!

→ N

```
int A[N];
```



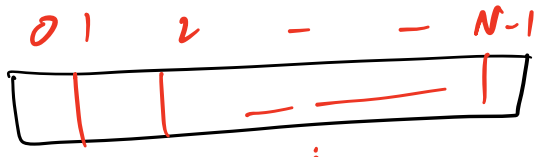
index → 0 1 2 ~ N-1

A

$\boxed{A[2] = 5}$ → put 5 at index 2

print (A[2]) → print val at index 2.

Q Given an array of size N.
Print the entire array!



$N = 5$

10^5

print(A[0]) 0 i
— (A[1]) 1
— (A[2]) 2
— (A[3]) 3
— (A[4]) 4

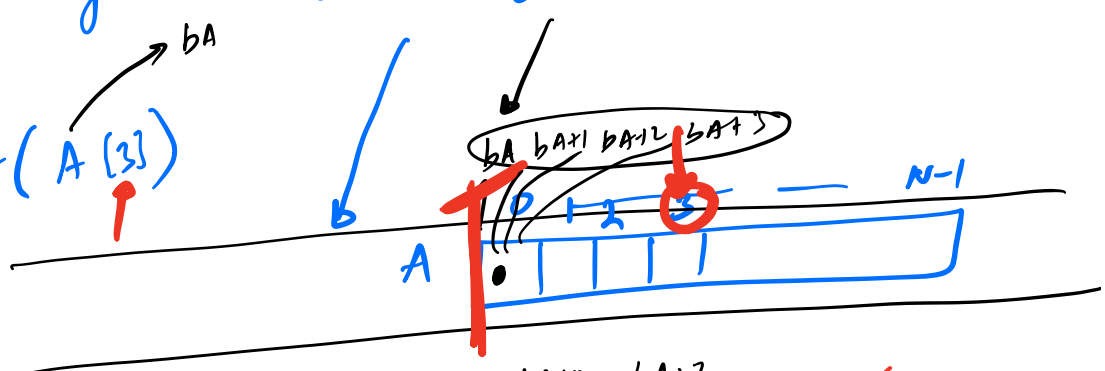
$N \times O(1)$

```
f( i=0; i<N; i++) {
    print( A[i]);
}
```

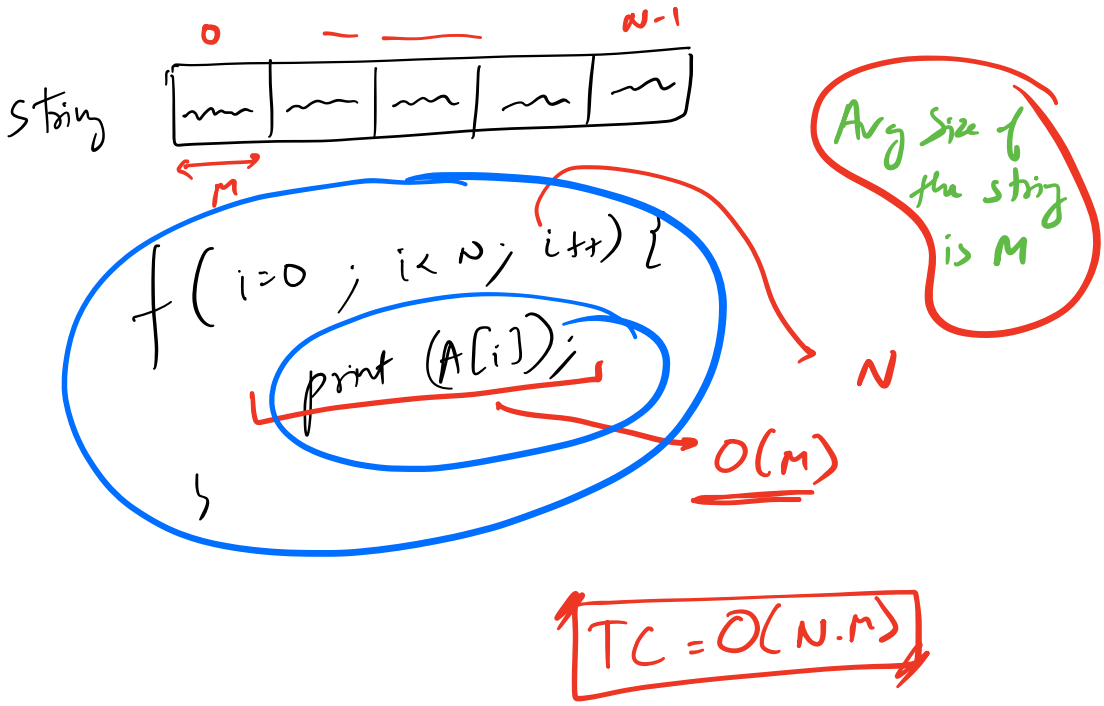
$\boxed{TC : O(N)}$

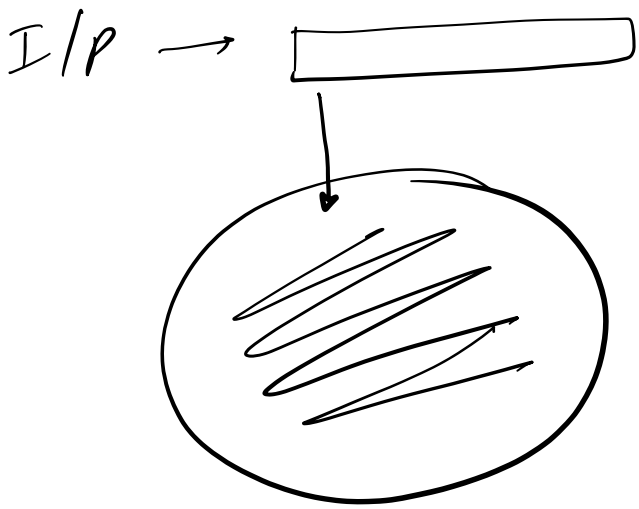
TC for accessing a single element of an array!

print(A[3])



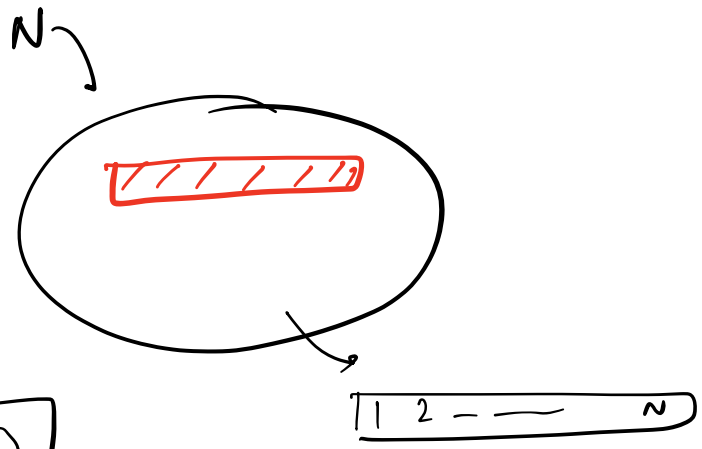
$\boxed{A[3] \rightarrow bA + 3 \times 4}$ $O(1)$


0
N-1
String
M
for (i=0 ; i<N; i++) {
print (A[i]);
}
O(M)
N
Avg Size of the string is M
TC = O(N.M)


I/P

$\boxed{SC : O(1)}$

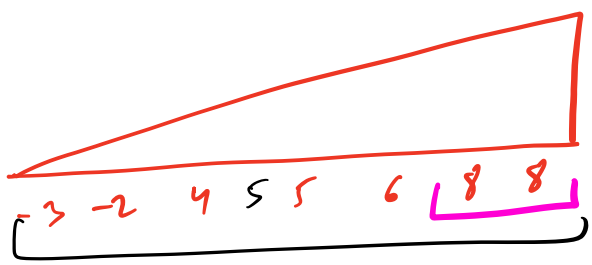
(A)



$\boxed{SC: O(1)}$

Q Given N elements → Array.
Count the no. of elements having atleast 1 element greater than itself!

A = [-3, -2, 6, 8, 4, 8, 5] → 5



ANS → N − # of occ. of MAX element

A : [-3, -2, 6, 8, 7, 8, 5] → MAX = 8

STEPS :

1. Find the MAX element
2. Find occ. of the MAX element.

1. A : [5, 7, 2, 3, 9, 8, 4, 8]

mC

9

mC = $-\infty$

1)

```
int mC = INT_MIN
f(i=0; i< N; i++) {
    if (A[i] > mC) {
        mC = A[i];
    }
}
```

→ $N \times O(1)$

$\boxed{TC : O(N)}$

$\boxed{SC : O(1)}$

2.

```
int cnt = 0;
f( i=0; i<N; i++){
    if( A[i] == mC){
        cnt++;
    }
}
```

mC = 5

3	5	2	2	5	4

↑ i

cnt

~~0~~ ~~1~~ 2

$\boxed{TC = O(N)}$

$\boxed{SC = O(1)}$

3. ANS → <u>N - cnt</u>

$TC = O(N) + O(N) = O(2N) = O(N)$

$\boxed{TC = O(N)}$

$\boxed{SC = O(1)}$

II

A : [2, 3, 10, 7, 3, 10, 2, 10, 5]

mC = -∞	2	3	10	10	10	10	10	10	10
cnt = 0	1	1	1	1	1	2	2	3	3

```
mC = -∞
cnt = 0
f( i=0; i<N; i++) {
    if( A[i] > mC) {
        mC = A[i];
        cnt = 1;
    }
    else if( A[i] == mC) {
        cnt++;
    }
}
```

$N \times O(1)$

$\boxed{TC: O(N)}$

$\boxed{SC: O(1)}$

ANS $\longrightarrow$ N - cnt

Q Given an array of size N & K

Check if there exists a pair (i, j)

Such that $A[i] + A[j] == K$ && $i != j$

	0	1	2	3	4	5	6
A:	3	-2	1	4	3	6	8

K = 10

$(i, j) \rightarrow (3, 5)$

	0	1	2	3
A:	2	4	-3	7

K = 8

i \ j	0	1	2	3
0	~~(0,0)~~	(0,1)	(0,2)	(0,3)
1	(1,0)	~~(1,1)~~	(1,2)	(1,3)
2	(2,0)	(2,1)	~~(2,2)~~	(2,3)
3	(3,0)	(3,1)	(3,2)	~~(3,3)~~

	j: [i+1 ... N-1]
i = 0	1, 2, 3
i = 1	2, 3
i = 2	3
i = 3	x

I)

```
f(i=0; i<N; i++){
    f(j=0; j<N; j++){
        if(i==j) continue;
        if(A[i]+A[j]==K){
            print(i,j);
        }
    }
}
```

TC:

$N \times N \times O(1)$

$\boxed{TC: O(N^2)}$

II

```
f(i=0; i<N-1; i++){
    f(j=i+1; j<N; j++){
        if(A[i]+A[j]==K){
            print(i,j);
        }
    }
}
```

i	j, [i+1 .. N-1]	
0	[1, N-1]	→ N-1
1	[2, N-1]	→ N-2
2	[3 - N-1]	→ N-3
3		
⋮		
N-2	(N-2, N-1)	→ 2

$\boxed{TC: O(N^2)}$

$\boxed{SC: O(1)}$

$2+3+4 \cdots N-1$

$\frac{N(N-1)}{2} - 1$

Q Given an Array. Reverse it!

IN PLACE!

A = [-1, 4, 7, 6, 2, 7, 8, 10]

[10 8 7 2 6 7 4 -1]

EVEN

[-1, 4, 7, 6, 2, 7, 8, 10]

i = 0 i i j i j j N-1

swap(i,j)

i < j ✓

ODD

[-1, 4, 7, 6, 2, 7, 8]

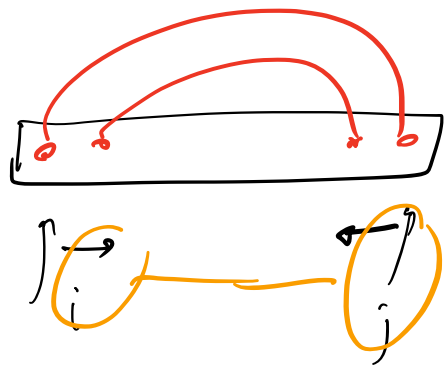
i = 0 i i j i,j j j j = N-1

i < j ✓

```
// A[], N
i = 0, j = N-1;
while( i < j) {
    swap(A[i], A[j]);
    i++;
    j--;
}
```

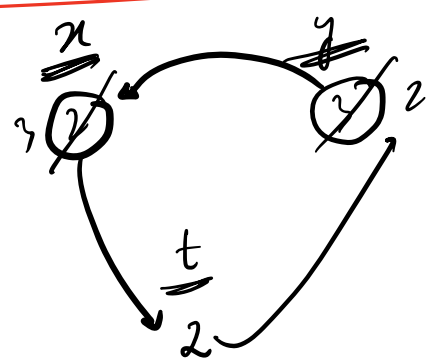
O(1)

2



#it → N/2

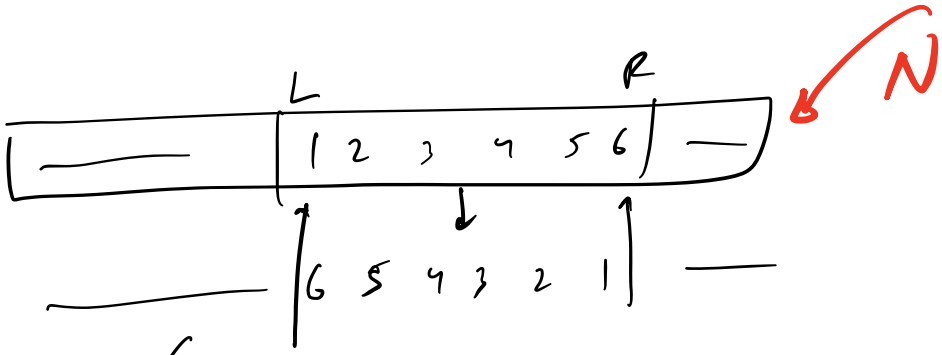
$\boxed{TC = O(N)}$



t = x
x = y
y = t

Q Given an Array.
Reverse the subarray [L, R].

R-L+1




```
rev(A[], L, R) {
    i = L; j = R;
    while (i < j) {
        swap(A[i], A[j]);
        i++;
        j--;
    }
}
```

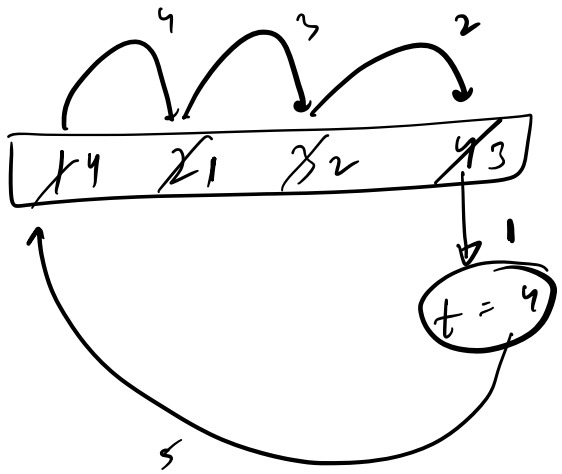
TC : O(N)

TC : O(R-L)

Q Given an Array.
Rotate the array from last to first K times

	0	1	2	3	4	5	6
A :	3	-2	1	4	6	9	8
K = 1	8	3	-2	1	4	6	9
K = 2	9	8	3	-2	1	4	6
K = 3	6	9	8	3	-2	1	4

1) BF



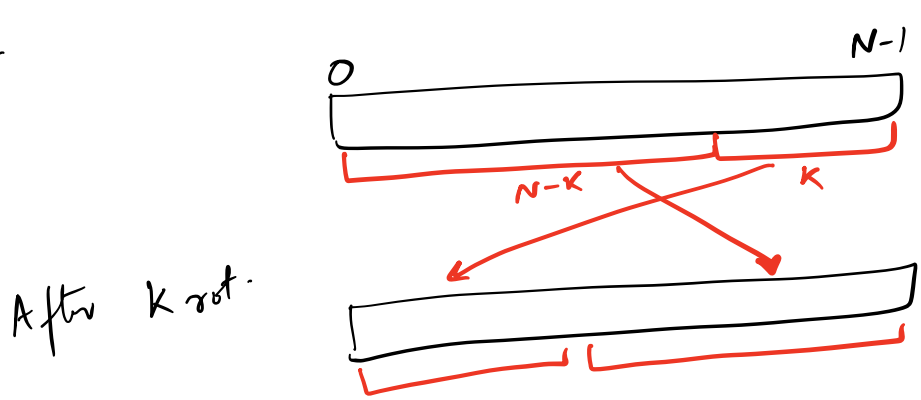
→ rot 1 time

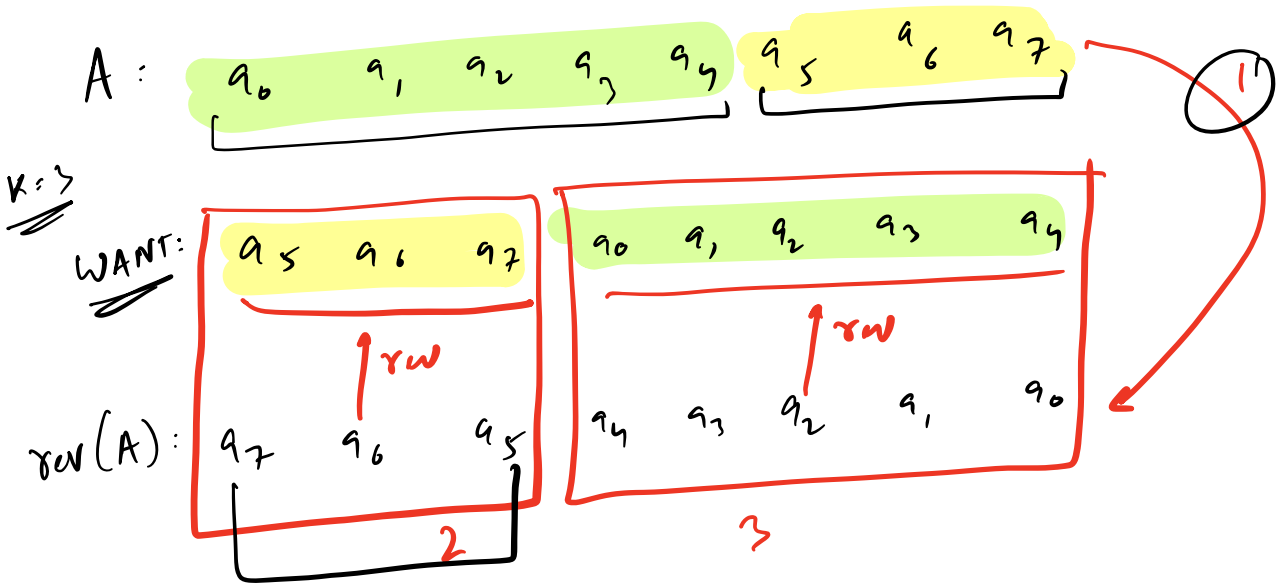
$O(N)$

rot K times

$\boxed{TC: O(K.N)}$

II





Steps

1) A $\longrightarrow$ Rev(A)

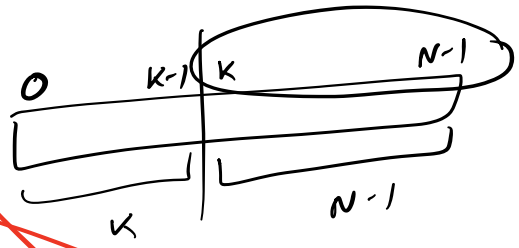
2) Rev the first K elements

3) Rev the last N-K element

✓ rev(A, 0, N-1)

✓ rev(A, 0, K-1)

✓ rev(A, K, N-1)



TC: N + K + N-K

$\boxed{TC: O(N)}$

$\boxed{SC: O(1)}$

K=0	1	2	3	K=3	K=6
K=1	3	1	2	K=4	K=7 10
K=2	2	3	1	K=5	

K%N=0

%N = 1

%N = 2

10%3 = 1

N=3

K=10

$\boxed{K = K\%N}$

K=0