Today's agenda

↳ Reverse a given Part of array

↳ Rotate array by K.

↳ greater than itself

↳ Two Sum

Q) Reverse a Part of array

↳ Given N array element and [s,e], reverse the array from [s,e].

[3,7] Ex: arr[]: {-3 4 2 8 3 9 6 2 8 10}

	0	1	2	3	4	5	6	7	8	9
arr[]:	-3	4	2	8	3	9	6	2	8	10

```
P S void reverse (int arr[ ], int s,int e){

        int sp= s;
        int ep = e;

        while (sp<ep) {

            int temp = arr[sp];
            arr[sp] = arr[ep];
            arr[ep] = temp;
              sp++;
              ep--;

        }

}
```

	0	1	2	3	4	5	6	7	8	9
arr[]:	-3	4	2	~~8~~	~~3~~	9	~~6~~	~~2~~	8	10
				2	6		3	8		

s: 3 e= 7

sp	ep	sp< ep
3	7	t
4	6	t
5	5	f

Q) Rotate the array

↳ Given N elements, Rotate array from last to first by K times. {google, meta, amazon}

K=3

ex: arr[7]: {3 -2 1 4 6 9 8}

↓ 1 rot.

{8 3 -2 1 4 6 9}

↓ 2nd rot.

{9 8 3 -2 1 4 6}

↓ 3rd rot.

{6 9 8 3 -2 1 4}

K=3

arr[7]: {3 -2 1 4 6 9 8}

↓

Reverse the whole array.

{8 9 6 | 4 1 -2 3}

↓ Reverse the first K elements

{6 9 8 | 4 1 -2 3}

↓ Reverse the elements after K elements

{6 9 8 | 3 -2 1 4}

{6 9 8 | 3 -2 1 4}

k=6

Q) arr[9]:

0 1 2 3 4 5 6 7 8

{4 1 6 9 2 14 7 8 3}

↓ reverse the whole array

{3 8 7 14 2 9 | 6 1 4}

↓ reverse the first k elements.

{9 2 14 7 8 3 | 6 1 4}

↓ reverse the rem. elements.

↓

{9 2 14 7 8 3 4 1 6}

{9 2 14 7 8 3 4 1 6}

$N: 10^8$ $K = 10^7$ → reverse ↓ 3

//Psuedo Code

```
P S void main ( ) {
    //input
    int n = . . —
    int [] arr = new int [n];
        for (            ) {
        }   arr[i] = . . —
    int K = . . — — ;
        k = k%n;
    //step1 : reverse whole array.  → [0,n-1]
        reverse (arr, 0, n-1);
    //step2 : reverse the first k elements.
        reverse (arr, 0, k-1);
    //step3 : revese the elements after kth
        reverse (arr, k, n-1);
}
```

```
P S void reverse (int arr[ ], int s, int e){
    int sp = s;
    int ep = e;
        while (sp<ep) {
            int temp = arr[sp];
            arr[sp] = arr[ep];
            arr[ep] = temp;
            sp++;
            ep--;
        }
}
```

K=1000

0 1 2 3

Q) arr[4]: {4 1 6 9}   n=4   K= 8

↓rot1

{9 4 1 6}

↓rot2

{6 9 4 1}

↓rot3

{1 6 9 4}

↓rot4

{4 1 6 9}

obs 1:

→ you will get same array if you do rotation in multiples of arr.length.

n	k	
5	50	→ Same array
5	45	→ Same array

ultimately   K % n

5   52   2 rotations → 52%5

% n → {0 1 2 n-1}

%5 → {0 1 2 3 4}

⇒ K = K % n → this much rotation you have to do.

n (arr.length)	K=13	
7	13-7=6	⇒ 6 rot.
7	31-7=24-7=17-7=10-7=3 31%7 = 3	
8	34	→ 34%8 = 2

K = K % arr.length

↑

8	4	→ 4%8 = 4
6	7	→ 7%6 = 1

Break till 9:55 Pm

Q) Given N array elements, Count total no. of elements having atleast 1 element greater than itself.

	0	1	2	3	4	5	6
ex: arr[7]:	{ -4	-3	7	9	3	9	4 }

↳ ans = 5

arr[8]: {3 4 11 8 2 10 9 11}

↳ ans=6

arr[5]: {7 7 7 7 7}

↳ ans=0

Obs1: max elements of the array are invalid.

Obs2: except for max element, all the elements are valid.

// find the occ. of max element → count.

ans = n - count

//PSuedo Code

```
int Countgreater (int arr[n]){

    int max= Integer.Min_VALUE;

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

    int Count=0;
    for (int i=0; i<arr.length; i++){
        if(arr[i] == max){
            Count ++;
        }
    }

    return arr.length -Count;
}
```

```
int max = Integer.min.VALUE;

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

    int Count = 0;
    for (int i=0; i<arr.length; i++){
            if(arr[i] == max){
                Count ++;
            }
    }
```

0	1	2	3	4	5	6

arr[7] = {3 4 11 8 2 10 9}

max = ~~-∞~~ ~~3~~ ~~4~~ 11

Count = ~~0~~ 1

ans = 7-1 = 6

Q) Two Sum

↳ Given N array elements, Check if there exists a Pair (i,j) Such that arr[i]+arr[j] == K and i != j.

Note: i & j are index value, K is given sum.

		0	1	2	3	4	5	6
ex:	arr[7]: {	2	-1	0	3	2	5	7 }

K=8

↳ true

	0	1	2	3
arr[4]: {	1	3	-2	6 }

K=5

↳ false

	0	1	2	3	4
arr[5]: {	2	4	-3	7	10 }

K=8

arr[1] + arr[1] = 8

4 + 4

↳ false

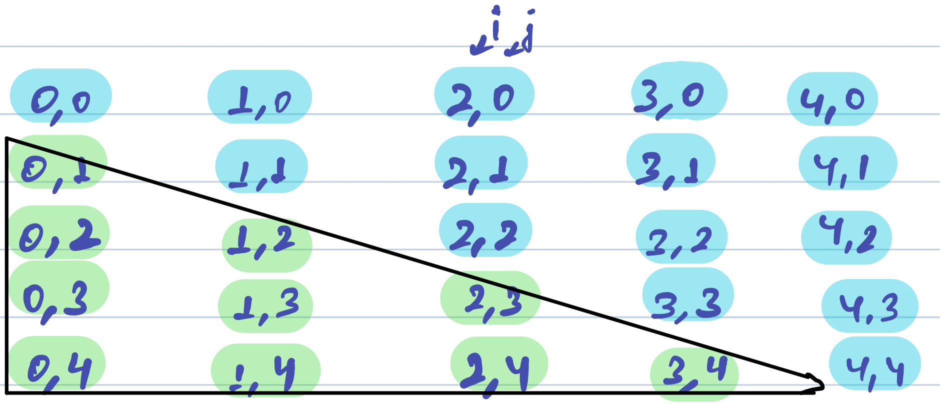
arr[6]: { 3 5 1 8 3 7 }

K=6

↳ true

$$\begin{array}{cccccc} & 0 & 1 & 2 & 3 & 4 \\ arr[5] : \{ & 3 & 5 & 2 & 7 & 5 \; \} \end{array}$$

k=12



0,0	1,0	2,0	3,0	4,0
0,1	1,1	2,1	3,1	4,1
0,2	1,2	2,2	2,2	4,2
0,3	1,3	2,3	3,3	4,3
0,4	1,4	2,4	3,4	4,4

n=5

i →	0	1	2	3
	j(1 2 3 4)	j(2 3 4)	j(3,4)	j(4)

```
Public Static boolean twoSum(int arr[],int k){
      int n=arr.length;   i < n-1
                            or
      for (int i=0; i<= n-2 ; i++){
                                   j<=n-1
                                     or
            for (int j=i+1; j<n ; j++){

                  if (arr[i] + arr[j] == k){
                        return true;
                  }
            }

      }
      return false;
}
```

n=5 k=12

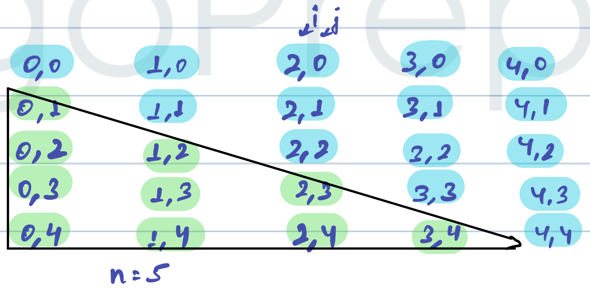
```
Public Static boolean twoSum(int arr[], int k){
    int n=arr.length;
    for (int i=0; i < n-1 ; i++){
        for (int j=i+1; j<n ; j++){
            if (arr[i] + arr[j] == k){
                return true;
            }
        }
    }
}
```

i	j	arr[i]+arr[j]
0	1	8
	2	5
	3	10
	4	8
	5 → exit	
1	2	7
	3	12
	4	
2		
3		

arr[5] : { 3 5 2 7 5 }

0	1	2	3	4
3	5	2	7	5

k=12



0,0	1,0	2,0	3,0	4,0
0,1	1,1	2,1	3,1	4,1
0,2	1,2	2,2	2,2	4,2
0,3	1,3	2,3	3,3	4,3
0,4	1,4	2,4	3,4	4,4

Active learning
↓
H.W

Passive learning