Today's agenda

- Intro to arrays
  - Syntax
  - Storing values
  - Reading input
- Return sum of arr[] elements
- Return max of arr[] elements.
- Array with functions
- Swap 2 indexes
- Reverse array
- 1 more problem

// Intro to array

```
int a = 1;
int b = 2;
int c = 3;
int d = 4;
.
.
.
int j = 10;
```

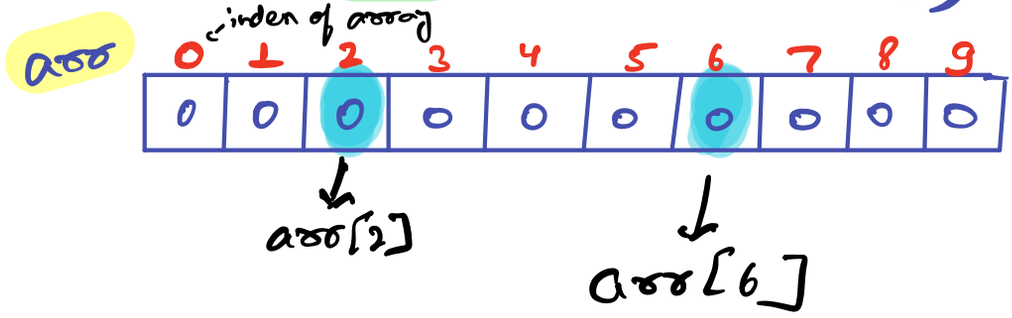
100 Students
!!
to store 100 variables
↓
Arrays

Arrays Syntax

```
type [] name = new type [size];
```

Q) Create an array of size 10 containing integers.

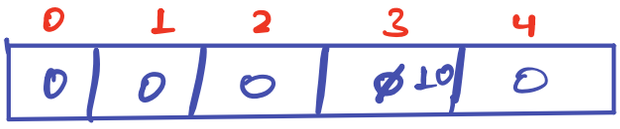
```
int [] arr = new int [10];
```



Indexing & Properties

↳ `int[] arr = new int[5];`

arr



→ `System.out.println(arr[3]);` → 0

`arr[3] = 10;`

→ `System.out.println(arr[3]);` → 10

→ `System.out.println(arr[5]);` → error (index out of bound)

→ If array is of length N.

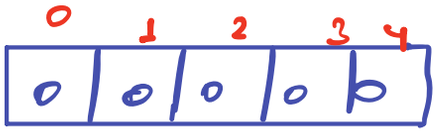
1st index = 0;
last index = N-1;

Q) Create an array of length 5 with values 10 20 30 40 50.

Way 1:

Step1: Create the array.

↳ int [] arr = new int [5];



Step2: assign the values.

```
arr[0] = 10;
arr[1] = 20;
arr[2] = 30;
arr[3] = 40;
arr[4] = 50;
```

Way 2:

```
int [] arr = {10, 20, 30, 40, 50};
```

(0 1 2 3 4)

```
System.out.println(arr[3]);
```

→ 40

→ to get size of array: arr.length;

Q) Sum of array

↳ Read an array of N length and Print the Sum of all elements.

Eg: arr[4]: 10 -1 3 -7 ⇒ 5 (indices: 0 1 2 3)

//Pseudo Code

```
void main( ) {
    Scanner scn = new Scanner (System.in);
    int n = scn.nextInt();
    int [] arr = new int[n];
    for (int i=0; i<n; i++){
        arr[i] = scn.nextInt();
    }

    int Sum=0;
    for (int i=0; i<n; i++ ){

        Sum = Sum + arr[i];
    }
    s.o.p(sum);
}
```

Tracing

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

```
int Sum = 0;
for (int i = 0; i<n; i++){
    Sum = Sum + arr[i];
}
S.O.P(Sum);
```

Sum = 0

i	i<n	Sum
0	t	10
1	t	9
2	t	12
3	t	5
4	f	

f → exit

45

Q) Max of array elements

↳ Read an array of N length and Print the max of all elements.

	0	1	2	3	
Ex: arr[4]:	10	-1	3	-7	→ 10
arr[4]:	-10	-20	-30	-40	→ -10

//Pseudo Code

```
void main( ) {
    Scanner scn = new Scanner (System.in);
    int n = scn.nextInt();
    int [] arr = new int[n];
    for (int i=0; i<n; i++) {
        arr[i] = scn.nextInt();
    }

    int max = 0;        → max = arr[0];
                        ↳ max = -∞ (Integer.MIN_VALUE)

    for (int i=0; i<n; i++) {
        if (arr[i] > max) {
            max = arr[i];
        }
        else {
            // nothing
        }
    }

    S.O.P (max);
}
```

```
int max = 0;

for(int i=0; i<n; i++){
    if(arr[i] > max){
        max = arr[i];
    }
    else{
        // nothing
    }
}
```

arr[5]:

0	1	2	3	4
10	-1	3	-7	20

max: 0

i	i<n	arr[i]>max	max
0	t	t	10
1	t	f	10
2	t	f	10
3	t	f	10
4	t	t	20
5	f		

↳ exit

max: -∞

arr[4]:

0	1	2	3
-10	-20	-30	-40

↳ 0

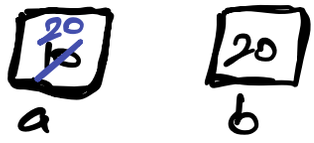
Break till 7:30 PM

Q) Swap the values of 2 variables

a = 10 b = 20 ⇒ a = 20 b = 10

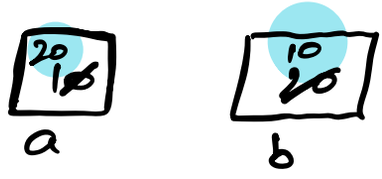
// incorrect way

```
void main() {
    int a = 10;
    int b = 20;

    a = b;
    b = a;
}
```



// Correct way

```
void main() {
    int a = 10;
    int b = 20;

    int temp = a;
    a = b;
    b = temp;
}
```



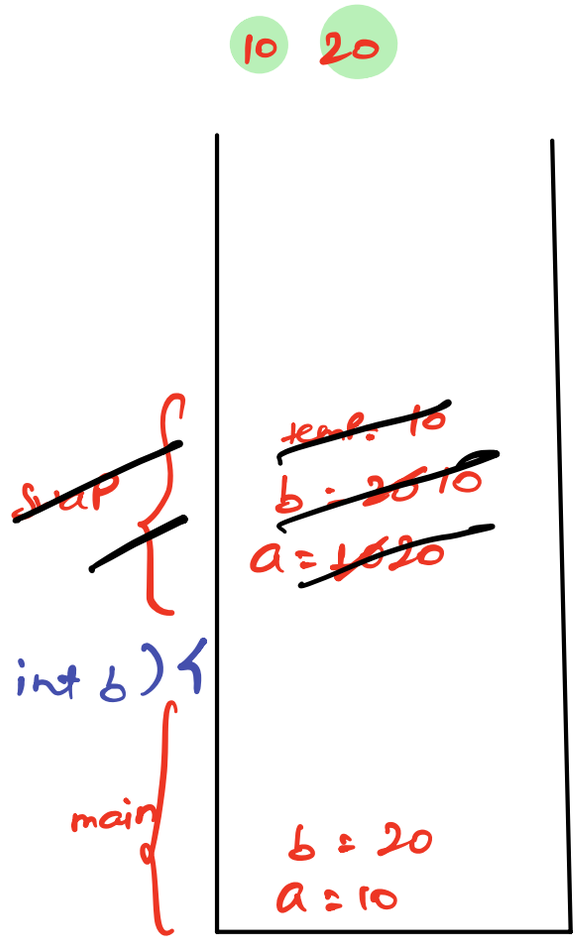
// Arrays with functions

```
main ( ) {
    int a = 10;
    int b = 20;
    swap (a, b);
    S.O.P (a);
    S.O.P (b);
}
```

→ 10

→ 20

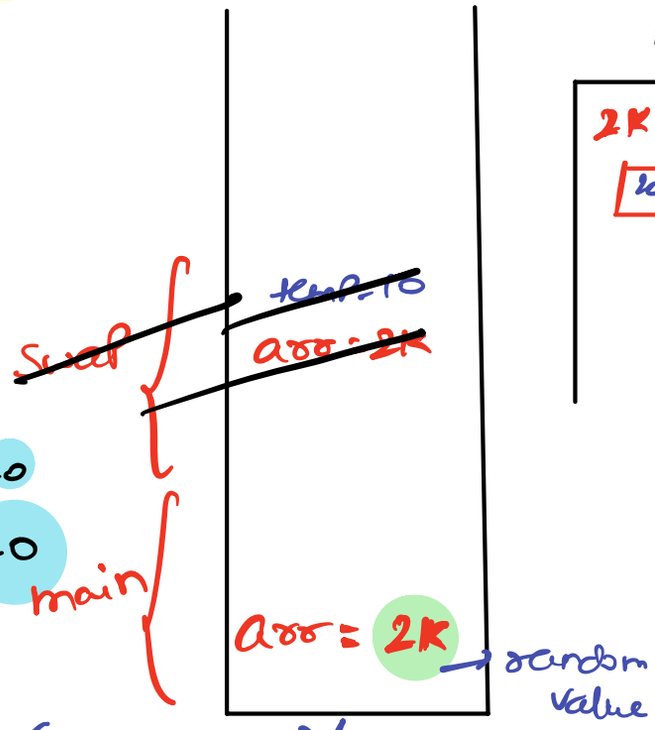
```
Public static void swap (int a, int b) {
    int temp = a;
    a = b;
    b = temp;
}
```



→ variables of 2 functions are not connected.

Q) → 20 10

```
void main() {
    int[] arr = {10, 20};

    swap(arr);

    s.o.p(arr[0]);   → 20
    s.o.p(arr[1]);   → 10
}
```



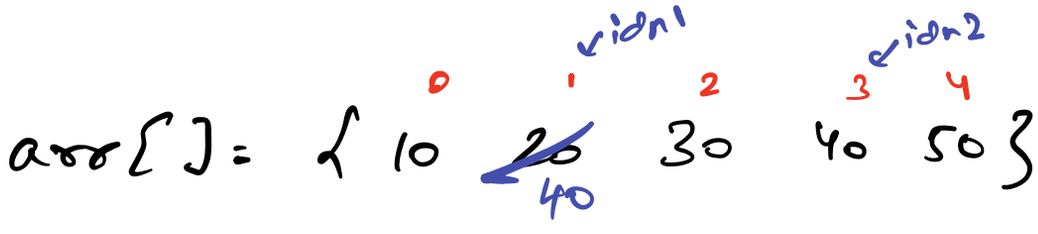

```
Public static void swap(int[] arr){

    int temp = arr[0];
    arr[0] = arr[1];
    arr[1] = temp;
}
```

→ arrays across functions are always connected.

Q) Swap indexes

↳ Given array of length N and two indexes idx1 and idx2, swap the element of those two indexes.

↳ done just before this page



idx1 = 1 idx2 = 3

```
int temp = arr[idx1];
arr[idx1] = arr[idx2];
arr[idx2] = temp;
```

Q) Reverse array

↳ Given array of length N, Reverse the whole array.

ex: arr[5]: { 10 20 30 40 50 }

0 1 2 3 4

↓

50 40 30 20 10

Combination of Swaps : reverse

arr[5]: { 10 20 30 40 50 }

0 1 2 3 4

4 3 1 0

↓

50 40 30 20 10

Swap (0, 4)

Swap (1, 3)

arr[10] :

0	1	2	3	4	5	6	7	8	9
10	20	30	40	50	60	70	80	90	100
9	8	7	6	5	4	3	2	1	0
100	90	80	70	60	50	40	30	20	10

[Swap (0, 9)
Swap (1, 8)
Swap (2, 7)
Swap (3, 6)
Swap (4, 5)] : reverse the array

//Pseudo code

```
int main() {
    // arr input

    reverse(arr);
}
```

P S void reverse (int arr[ ]){

```
int sP: 0
int eP: arr.length-1;
```

Swap (0, 9)
Swap (1, 8)
Swap (2, 7)
Swap (3, 6)
Swap (4, 5)

```
while (sP<eP) {

    int tempP = arr[sP];
    arr[sP] = arr[eP];
    arr[eP] : tempP;
      sP++;
      eP--;

}
```

}

```
int sp = 0
int ep = arr.length-1;

while (sp<ep) {
    int temp = arr[sp];
    arr[sp] = arr[ep];
    arr[ep] = temp;
    sp++;
    ep--;
}
```

arr[9] = 90 80 70 60 50 40 30 20 10

sp	ep	sp < ep
0	8	t
1	7	t
2	6	t
3	5	t
4	4	f → exit

```
int sp = 0
int ep = arr.length-1;

while (sp<ep) {
    int temp = arr[sp];
    arr[sp] = arr[ep];
    arr[ep] = temp;
    sp++;
    ep--;
}
```

arr[6] = 60 50 40 30 20 10

sp	ep	sp < ep	sp != ep (won't work)
0	5	t	t
1	4	t	t
2	3	t	t
3	2	f → exit	t

Sum of Array

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int[] arr = new int[n];
        for(int i=0;i<n;i++){
            arr[i] = scn.nextInt();
        }

        int sum = 0;

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

        System.out.println(sum);
    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;

int main() {
    int n;
    cin >> n;

    int* arr = new int[n];
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }

    int sum = 0;

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

    cout << sum << endl;

    delete[] arr;
    return 0;
}
```

Python Code:

```
def main():
    n = int(input())
    arr = list(map(int, input().split()))

    sum_ = 0
    for num in arr:
        sum_ += num

    print(sum_)

if __name__ == "__main__":
    main()
```

Max of Array

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int[] arr = new int[n];
        for(int i=0;i<n;i++){
            arr[i] = scn.nextInt();
        }

        int max = Integer.MIN_VALUE;

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

        System.out.println(max);
    }
}
```

C++ Code:

```
#include <iostream>
#include <limits>
using namespace std;

int main() {
    int n;
    cin >> n;
```

```
    int* arr = new int[n];
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }

    int max = numeric_limits<int>::min();

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

    cout << max << endl;

    delete[] arr;
    return 0;
}
```

Python Code:

```
def main():
    n = int(input())
    arr = list(map(int, input().split()))

    max_ = float('-inf')

    for num in arr:
        if num > max_:
            max_ = num

    print(max_)

if __name__ == "__main__":
    main()
```

Swap Indexes

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
       Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int[] arr = new int[n];
        for(int i=0;i<n;i++){
            arr[i] = scn.nextInt();
        }
        int idx1 = scn.nextInt();
        int idx2 = scn.nextInt();

        int temp = arr[idx1];
        arr[idx1] = arr[idx2];
        arr[idx2] = temp;

         for(int i=0;i<n;i++){
            System.out.print(arr[i]+" ");
        }
    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;

int main() {
    int n;
    cin >> n;

    int* arr = new int[n];
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
```

```
    }

    int idx1, idx2;
    cin >> idx1 >> idx2;

    int temp = arr[idx1];
    arr[idx1] = arr[idx2];
    arr[idx2] = temp;

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

    delete[] arr;
    return 0;
}
```

Python Code:

```
def main():
    n = int(input())
    arr = list(map(int, input().split()))

    idx1, idx2 = map(int, input().split())

    arr[idx1], arr[idx2] = arr[idx2], arr[idx1]

    for num in arr:
        print(num, end=" ")

if __name__ == "__main__":
    main()
```

Reverse Array

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int[]arr = new int[n];

        for(int i=0;i<n;i++){
            arr[i] = scn.nextInt();
        }

        int s = 0;
        int e = n-1;
        while(s < e){
            int temp = arr[s];
            arr[s] = arr[e];
            arr[e] = temp;
            s++;
            e--;
        }

        for(int i=0;i<n;i++){
            System.out.print(arr[i]+" ");
        }

    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;

int main() {
    int n;
    cin >> n;

    int* arr = new int[n];
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }

    int s = 0;
    int e = n - 1;
    while (s < e) {
        int temp = arr[s];
        arr[s] = arr[e];
        arr[e] = temp;
        s++;
        e--;
    }

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

    delete[] arr;
    return 0;
}
```

Python Code:

```
def main():
    n = int(input())
    arr = list(map(int, input().split()))

    s = 0
    e = n - 1
    while s < e:
        arr[s], arr[e] = arr[e], arr[s]
        s += 1
        e -= 1

    for num in arr:
        print(num, end=" ")

if __name__ == "__main__":
    main()
```

Largest Number at least twice_HW

Solution Vid: https://youtu.be/_Dj2BNXTzCY

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int[]arr = new int[n];

        for(int i=0;i<n;i++){
            arr[i] = scn.nextInt();
        }

        System.out.println(dominantIndex(arr));
    }

     public static int dominantIndex(int[] arr) {
```

```
        int max = Integer.MIN_VALUE;
        int index = -1;
        int second = -1;
         ]
        for (int i = 0; i < arr.length; i++) {
            if (arr[i] > max) {
                second = max;
                max = arr[i];
                index = i;
            } else if (arr[i] > second)
                second = arr[i];
        }
        return second * 2 <= max ? index : -1;
    }
}
```

C++ Code:

```
#include <iostream>
#include <vector>
using namespace std;

int dominantIndex(vector<int>& arr) {
    int max = INT_MIN;
    int index = -1;
    int second = -1;

    for (int i = 0; i < arr.size(); i++) {
        if (arr[i] > max) {
            second = max;
            max = arr[i];
            index = i;
        } else if (arr[i] > second)
            second = arr[i];
    }

    return second * 2 <= max ? index : -1;
}

int main() {
    int n;
    cin >> n;
    vector<int> arr(n);

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

```
    }

    cout << dominantIndex(arr) << endl;

    return 0;
}
```

Python Code:

```
def main():
    n = int(input())
    arr = list(map(int, input().split()))
    print(dominantIndex(arr))

def dominantIndex(arr):
    max_val = float('-inf')
    index = -1
    second = -1

    for i in range(len(arr)):
        if arr[i] > max_val:
            second = max_val
            max_val = arr[i]
            index = i
        elif arr[i] > second:
            second = arr[i]

    return index if second * 2 <= max_val else -1

if __name__ == "__main__":
    main()
```

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] Eg: arr[] = {-3 4 2 8 3 9 6 2 8 10}

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

arr[] = {-3 4 2 8 3 9 6 2 8 10}

s: 3 e: 7

int sP = s;
int eP = e;

sP	eP	sP < eP
3	7	t
4	6	t
5	5	f

```
while (sP<eP) {
    int temP = arr[sP];
    arr[sP] = arr[eP];
    arr[eP] : temP;
    sP++;
    eP--;
}
```

}

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]: { 4 1 6 9 2 14 7 8 3 }

0 1 2 3 4 5 6 7 8

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 sec. 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

// Pseudo 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 : reverse 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

Q) arr[4]: { 4 1 6 9 } (0 1 2 3) n = 4 K = 8

↓ rot 1

{ 9 4 1 6 }

↓ rot 2

{ 6 9 4 1 }

↓ rot 3

{ 1 6 9 4 }

↓ rot 4

{ 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

5 52 ultimately 2 rotations → 52 % 5 (K % n)

% 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 ans.

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.

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

↳ 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

//Pseudo 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 man = Integer.MIN_VALUE;

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

arr[7] = { 3 4 11 8 2 10 9 }
(0 1 2 3 4 5 6)

man: ~~-∞~~ 8 11

Count = 0 1

ans: 7 - 1 = 6

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

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.

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

K = 8 ↳ true

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

K = 5 ↳ false

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

K = 8

arr[1] + arr[2] = 8

4 + 4 ↳ false

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

K = 6

↳ true

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

k=12

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;
    for (int i=0; i <= n-2 ; i++){          // i < n-1
        for (int j=i+1; j < n ; j++){       // j <= n-1
            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;
            }
        }
    }
    return false;
}
```

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)

k:12

Active learning → H.W

Passive learning

Reverse Part of Array

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
         Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int[] arr = new int[n];
        for(int i=0;i<n;i++){
            arr[i] = scn.nextInt();
        }
        int s = scn.nextInt();
        int e = scn.nextInt();

        reversePart(arr,s,e);

        for(int i=0;i<n;i++){
            System.out.print(arr[i]+" ");
        }
    }

    public static void reversePart(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--;
        }

    }
}
```

C++ Code:

```
#include <iostream>
#include <vector>
using namespace std;

void reversePart(vector<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--;
    }
}

int main() {
    int n;
    cin >> n;
    vector<int> arr(n);
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }
    int s, e;
    cin >> s >> e;
    reversePart(arr, s, e);
    for (int i = 0; i < n; i++) {
        cout << arr[i] << " ";
    }
    return 0;
}
```

Python Code:

```
def reverse_part(arr, s, e):
    sp = s
    ep = e
    while sp < ep:
        arr[sp], arr[ep] = arr[ep], arr[sp]
        sp += 1
        ep -= 1
```

```
def main():
    n = int(input())
    arr = list(map(int, input().split()))

    s, e = map(int, input().split())

    reverse_part(arr, s, e)

    for num in arr:
        print(num, end=" ")

if __name__ == "__main__":
    main()
```

Rotate Array

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
         Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int[] arr = new int[n];
        for(int i=0;i<n;i++){
            arr[i] = scn.nextInt();
        }
        int k = scn.nextInt();

        k = k%n;
        reversePart(arr,0,n-1);
        reversePart(arr,0,k-1);
```

```
        reversePart(arr,k,n-1);

        for(int i=0;i<n;i++){
            System.out.print(arr[i]+" ");
        }
    }

    public static void reversePart(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--;
        }

    }
}
```

C++ Code:

```
#include <iostream>
#include <vector>
using namespace std;

void reversePart(vector<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--;
    }
}

int main() {
    int n;
    cin >> n;
    vector<int> arr(n);
```

```
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }
    int k;
    cin >> k;
    k = k % n;
    reversePart(arr, 0, n - 1);
    reversePart(arr, 0, k - 1);
    reversePart(arr, k, n - 1);
    for (int i = 0; i < n; i++) {
        cout << arr[i] << " ";
    }
    return 0;
}
```

Python Code:

```
def reverse_part(arr, s, e):
    sp = s
    ep = e
    while sp < ep:
        arr[sp], arr[ep] = arr[ep], arr[sp]
        sp += 1
        ep -= 1

def main():
    n = int(input())
    arr = list(map(int, input().split()))

    k = int(input())
    k = k % n

    reverse_part(arr, 0, n - 1)
    reverse_part(arr, 0, k - 1)
    reverse_part(arr, k, n - 1)

    for num in arr:
        print(num, end=" ")

if __name__ == "__main__":
    main()
```

Count Greater

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
         Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int[] arr = new int[n];
        for(int i=0;i<n;i++){
            arr[i] = scn.nextInt();
        }

        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<n;i++){
            if(arr[i] == max){
                count++;
            }
        }
```

```
        System.out.println(n - count);
    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;

int main() {
    int n;
    cin >> n;

    int* arr = new int[n];
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }

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

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

    cout << n - count << endl;

    delete[] arr;
    return 0;
}
```

Python Code:

```
def main():
    n = int(input())

    arr = list(map(int, input().split()))

    max_val = float('-inf')
    for num in arr:
        if num > max_val:
            max_val = num

    count = 0
    for num in arr:
        if num == max_val:
            count += 1

    print(n - count)

if __name__ == "__main__":
    main()
```

Two Sum Brute

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int[]arr = new int[n];

        for(int i=0;i<n;i++){
            arr[i] = scn.nextInt();
        }

        int k = scn.nextInt();

        System.out.println(twosum(arr,k));



    }

    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;
                }
            }
        }

        return false;
    }
}
```

C++ Code:

```
#include <iostream>
#include <vector>
using namespace std;

bool two_sum(const vector<int>& arr, int k) {
    int n = arr.size();
    for (int i = 0; i < n - 1; i++) {
        for (int j = i + 1; j < n; j++) {
            if (arr[i] + arr[j] == k) {
                return true;
            }
        }
    }
    return false;
}

int main() {
    int n;
    cin >> n;

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

    int k;
    cin >> k;

    cout << (two_sum(arr, k) ? "true" : "false") << endl;

    return 0;
}
```

Python Code:

```
def two_sum(arr, k):
    n = len(arr)
    for i in range(n - 1):
        for j in range(i + 1, n):
            if arr[i] + arr[j] == k:
                return True
    return False

def main():
    n = int(input())
    arr = list(map(int, input().split()))

    k = int(input())

    print("true" if two_sum(arr, k) else "false")

if __name__ == "__main__":
    main()
```

Max Difference 1

Solution Vid: https://youtu.be/vlbcMdMgY5o

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

// "static void main" must be defined in a public class.
public class Main {
    public static void main(String[] args) {

        Scanner scn = new Scanner(System.in);
         int n = scn.nextInt();
         int[]arr = new int[n];
```

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

        System.out.println(maxdifference_1(arr));

    }

    public static int maxdifference_1(int[]arr){
        int max = Integer.MIN_VALUE;
        int min = Integer.MAX_VALUE;

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

            if(arr[i] < min){
                min = arr[i];
            }

        }

        return max - min;
    }
}
```

C++ Code:

```
#include <iostream>
#include <vector>
using namespace std;

int max_difference_1(const vector<int>& arr) {
    int max_val = INT_MIN;
    int min_val = INT_MAX;

    for (int num : arr) {
        if (num > max_val) {
            max_val = num;
        }

        if (num < min_val) {
            min_val = num;
        }
    }
```

```
    return max_val - min_val;
}

int main() {
    int n;
    cin >> n;

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

    cout << max_difference_1(arr) << endl;

    return 0;
}
```

Python Code:

```
def max_difference_1(arr):
    max_val = float('-inf')
    min_val = float('inf')

    for num in arr:
        if num > max_val:
            max_val = num

        if num < min_val:
            min_val = num

    return max_val - min_val

def main():
    n = int(input())
    arr = list(map(int, input().split()))

    print(max_difference_1(arr))

if __name__ == "__main__":
    main()
```

Max Difference 2

Solution Vid: https://youtu.be/njkBFHgz05Q

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

// "static void main" must be defined in a public class.
public class Main {
    public static void main(String[] args) {

        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int[]arr = new int[n];

        for(int i=0;i<n;i++){
            arr[i] = scn.nextInt();
        }

        System.out.println(maxdifference_1(arr));

    }

    public static int maxdifference_1(int[]arr){
        for(int i=0;i<arr.length;i++){
            arr[i] = arr[i] + i;
        }

        int max = Integer.MIN_VALUE;
        int min = Integer.MAX_VALUE;

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

            if(arr[i] < min){
                min = arr[i];
            }
```

```
        }

        return max - min;
    }
}
```

C++ Code:

```
#include <iostream>
#include <vector>
using namespace std;

int max_difference_1(vector<int>& arr) {
    int n = arr.size();

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

    int max_val = INT_MIN;
    int min_val = INT_MAX;

    for (int num : arr) {
        if (num > max_val) {
            max_val = num;
        }

        if (num < min_val) {
            min_val = num;
        }
    }

    return max_val - min_val;
}

int main() {
    int n;
    cin >> n;

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

```
    cout << max_difference_1(arr) << endl;

    return 0;
}
```

Python Code:

```
def max_difference_1(arr):
    for i in range(len(arr)):
        arr[i] = arr[i] + i

    max_val = float('-inf')
    min_val = float('inf')

    for num in arr:
        if num > max_val:
            max_val = num

        if num < min_val:
            min_val = num

    return max_val - min_val

def main():
    n = int(input())
    arr = list(map(int, input().split()))

    print(max_difference_1(arr))

if __name__ == "__main__":
    main()
```

Max Difference 3

Solution Vid: https://youtu.be/MUFdnVghGkY

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

// "static void main" must be defined in a public class.
public class Main {
    public static void main(String[] args) {

        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int[]arr = new int[n];

        for(int i=0;i<n;i++){
            arr[i] = scn.nextInt();
        }

        System.out.println(maxdifference_1(arr));

    }

    public static int maxdifference_1(int[]arr){
        for(int i=0;i<arr.length;i++){
            arr[i] = arr[i] - i;
        }

        int max = Integer.MIN_VALUE;
        int min = Integer.MAX_VALUE;

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

```
            if(arr[i] < min){
                min = arr[i];
            }

        }

        return max - min;
    }
}
```

C++ Code:

```
#include <iostream>
#include <vector>
using namespace std;

int max_difference_1(vector<int>& arr) {
    int n = arr.size();

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

    int max_val = INT_MIN;
    int min_val = INT_MAX;

    for (int num : arr) {
        if (num > max_val) {
            max_val = num;
        }

        if (num < min_val) {
            min_val = num;
        }
    }

    return max_val - min_val;
}

int main() {
    int n;
    cin >> n;

    vector<int> arr(n);
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
```

```
    }

    cout << max_difference_1(arr) << endl;

    return 0;
}
```

Python Code:

```
def max_difference_1(arr):
    for i in range(len(arr)):
        arr[i] = arr[i] - i

    max_val = float('-inf')
    min_val = float('inf')

    for num in arr:
        if num > max_val:
            max_val = num

        if num < min_val:
            min_val = num

    return max_val - min_val

def main():
    n = int(input())
    arr = list(map(int, input().split()))

    print(max_difference_1(arr))

if __name__ == "__main__":
    main()
```

Max Difference 4

Solution Vid: https://youtu.be/SnzF5MAaO3w

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

// "static void main" must be defined in a public class.
public class Main {
    public static void main(String[] args) {

        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int[]arr = new int[n];

        for(int i=0;i<n;i++){
            arr[i] = scn.nextInt();
        }

        int ans1 = maxdifference_2(arr);
        int ans2 = maxdifference_3(arr);

        System.out.println(Math.max(ans1,ans2));

    }

     public static int maxdifference_3(int[]arr){
        for(int i=0;i<arr.length;i++){
            arr[i] = arr[i] - i;
        }

        int max = Integer.MIN_VALUE;
        int min = Integer.MAX_VALUE;

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

            if(arr[i] < min){
```

```
                min = arr[i];
            }

        }

         for(int i=0;i<arr.length;i++){
            arr[i] = arr[i] + i;
        }
        return max - min;
    }


    public static int maxdifference_2(int[]arr){
        for(int i=0;i<arr.length;i++){
            arr[i] = arr[i] + i;
        }

        int max = Integer.MIN_VALUE;
        int min = Integer.MAX_VALUE;

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

            if(arr[i] < min){
                min = arr[i];
            }

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

        return max - min;
    }
}
```

C++ Code:

```
#include <iostream>
#include <vector>
using namespace std;

int max_difference_2(vector<int>& arr) {
    for (int i = 0; i < arr.size(); i++) {
```

```
        arr[i] = arr[i] + i;
    }

    int max_val = INT_MIN;
    int min_val = INT_MAX;

    for (int num : arr) {
        if (num > max_val) {
            max_val = num;
        }

        if (num < min_val) {
            min_val = num;
        }
    }

    for (int i = 0; i < arr.size(); i++) {
        arr[i] = arr[i] - i;
    }

    return max_val - min_val;
}

int max_difference_3(vector<int>& arr) {
    for (int i = 0; i < arr.size(); i++) {
        arr[i] = arr[i] - i;
    }

    int max_val = INT_MIN;
    int min_val = INT_MAX;

    for (int num : arr) {
        if (num > max_val) {
            max_val = num;
        }

        if (num < min_val) {
            min_val = num;
        }
    }

    for (int i = 0; i < arr.size(); i++) {
        arr[i] = arr[i] + i;
    }

    return max_val - min_val;
}
```

```
int main() {
    int n;
    cin >> n;

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

    int max_val = INT_MIN;
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            if (abs(arr[i] - arr[j]) + i - j > max_val) {
                max_val = abs(arr[i] - arr[j]) + i - j;
            }
        }
    }

    int ans1 = max_difference_2(arr);
    int ans2 = max_difference_3(arr);

    cout << max(max_val, max(ans1, ans2)) << endl;

    return 0;
}
```

Python Code:

```
def max_difference_2(arr):
    for i in range(len(arr)):
        arr[i] = arr[i] + i

    max_val = float('-inf')
    min_val = float('inf')

    for num in arr:
        if num > max_val:
            max_val = num

        if num < min_val:
            min_val = num

    for i in range(len(arr)):
        arr[i] = arr[i] - i
```

```
    return max_val - min_val

def max_difference_3(arr):
    for i in range(len(arr)):
        arr[i] = arr[i] - i

    max_val = float('-inf')
    min_val = float('inf')

    for num in arr:
        if num > max_val:
            max_val = num

        if num < min_val:
            min_val = num

    for i in range(len(arr)):
        arr[i] = arr[i] + i

    return max_val - min_val

def main():
    n = int(input())
    arr = list(map(int, input().split()))

    max_val = float('-inf')
    for i in range(n):
        for j in range(n):
            if abs(arr[i] - arr[j]) + i - j > max_val:
                max_val = abs(arr[i] - arr[j]) + i - j

    ans1 = max_difference_2(arr.copy())
    ans2 = max_difference_3(arr.copy())

    print(max(max_val, max(ans1, ans2)))

if __name__ == "__main__":
    main()
```

Today's agenda

- Intro to 2D Arrays
- Print matrix row wise
- Print matrix Col wise
- Print matrix in wave form

// Intro to 2d array

↳ JEE Mains →

	Physics (0)	Chemistry (1)	Maths (2)
0th Student	80	85	65
1st Student	80	85	65
2nd Student	80	85	65
.			
.			
.			

↳ Excel Sheet → 2d array use

// Syntax

```
int[][] arr = new int[5][4];
```

(5 → row, 4 → col)

arr

	Column 0	1	2	3
0	0	0	0	0
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0
4	0	0	0	0

(row: 0 to 4)

⇒ arr[3][2]

↳ 5 * 4 = 20 elements

Q) Print matrix rowwise

↳ Print the given mat[N][m] rowwise.

Ex: arr[4][5] =

	0	1	2	3	4
0	10	20	30	40	50
1	60	70	80	90	100
2	110	120	130	140	150
3	160	170	180	190	200

→ 10 20 30 40 50
60 70 80 90 100
110 120 130 140 150
160 170 180 190 200

i, j

0 0 | 0 1 | 0 2 | 0 3 | 0 4

i, j

1 0 | 1 1 | 1 2 | 1 3 | 1 4

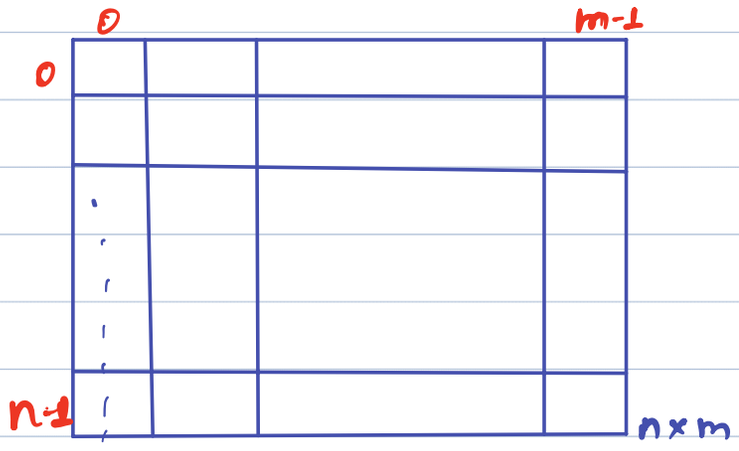
i, j

2 0 | 2 1 | 2 2 | 2 3 | 2 4

i, j

3 0 | 3 1 | 3 2 | 3 3 | 3 4

```
for(int i=0; i<n; i++){
    for(int j=0; j<m; j++){
        System.out.println(arr[i][j]);
    }
}
```



arr[4][5] =

	0	1	2	3	4
0	10	20	30	40	50
1	60	70	80	90	100
2	110	120	130	140	150
3	160	170	180	190	200

4 * 5

```
for(int i=0; i<n; i++){
    for(int j=0; j<m; j++){
        System.out.print(arr[i][j]+" ");
    }
    System.out.println();
}
```

```
10  20  30  40  50
60  70  80  90  100
110 120 130 140 150
160 170 180 190 200
```

i = 0
j: 0 1 2 3 4 5 → false

i = 1
j: 0 1 2 3 4 5 → false

i = 2
j: 0 1 2 3 4 5 → false

i = 3
j: 0 1 2 3 4 5 → false (exit)

i = 4 → false (exit)

Q) Print matrix Colwise

↳ Print the given mat[N][M] colwise.

Ex: arr[4][5] =

	0	1	2	3	4
0	10	20	30	40	50
1	60	70	80	90	100
2	110	120	130	140	150
3	160	170	180	190	200

4×5

10 60 110 160
20 70 120 170
30 80 130 180
40 90 140 190
50 100 150 200

i i

0 0 | 0 1 | 0 2 | 0 3 | 0 4
1 0 | 1 1 | 1 2 | 1 3 | 1 4
2 0 | 2 1 | 2 2 | 2 3 | 2 4
3 0 | 3 1 | 3 2 | 3 3 | 3 4

```
for (int j=0; j<m; j++){
    for (int i=0; i<n; i++){
        System.out.print(arr[i][j] + " ");
    }
    System.out.println();
}
```

```
for (int j=0; j<m; j++){
    for (int i=0; i<n; i++){
        System.out.print(arr[i][j]+" ");
    }
    System.out.println();
}
```

arr[4][5] =

	0	1	2	3	4
0	10	20	30	40	50
1	60	70	80	90	100
2	110	120	130	140	150
3	160	170	180	190	200

4x5

j: 0, 1, 2

i: 0, 1, 2, 3, 4 → exit

10 - 60 - 110 - 160

20 - 70 - 120 - 170

Break till 9:20 Pm

Q) Print matrix in wave form

↳ Print the given mat[N][m] in wave form.

Ex: arr[4][5] =

	0	1	2	3	4
0	10	20	30	40	50
1	60	70	80	90	100
2	110	120	130	140	150
3	160	170	180	190	200

N×m

L-R → 10 20 30 40 50

R-L → 100 90 80 70 60

L-R → 110 120 130 140 150

R-L → 200 190 180 170 160

L-R → 0, 2, ... (even no.)

R-L → 1, 3, ... (odd no.)

] row-wise

even no. that odd divisible by 2

```
for(int i=0; i<n; i++){
    if(i%2==0){
        for(int j=0; j<m; j++){
            System.out.print(arr[i][j]+" ");
        }
    }else{
        for(int j=m-1; j>=0; j--){
            System.out.print(arr[i][j]+" ");
        }
    }
}
```

```
for(int i=0; i<n; i++){
    if(i%2==0){
        for(int j=0; j<m; j++){
            System.out.print(arr[i][j]+" ");
        }
    }else{
        for(int j=m-1; j>=0; j--){
            System.out.print(arr[i][j]+" ");
        }
    }
    System.out.println();
}
```

10_ 20_ 30_ 40_ 50_

100_ 90_ 80_ 70_ 60_

arr[4][5] =

	0	1	2	3	4
0	10	20	30	40	50
1	60	70	80	90	100
2	110	120	130	140	150
3	160	170	180	190	200

n*m

i :

0 → if

1 → else

2

3

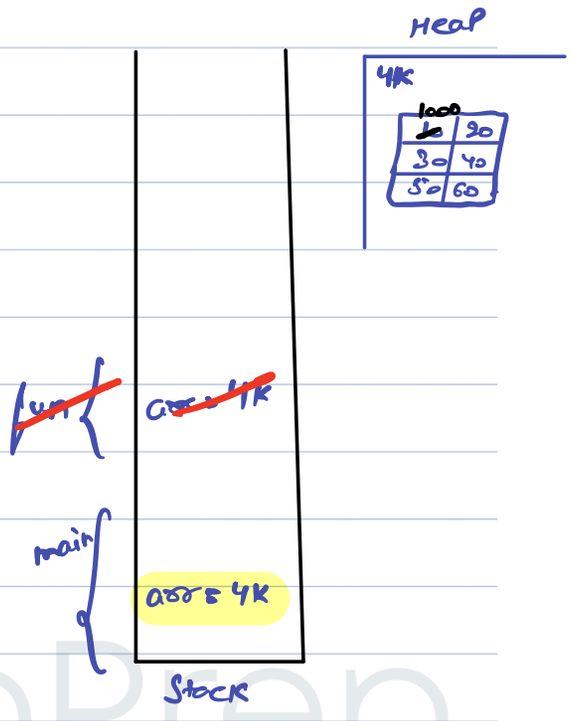
j :

0 1 2 3 4 5 → exit

4 3 2 1 0

//Pseudo code

```
void main() {
    //input 2d array -> arr[n][m]
        : { 10 20
            30 40
            50 60 }
    S.o.p(arr[0][0]); -> 10
    fun(arr);
-> S.o.p(arr[0][0]);
}

Public static void fun(int[][] arr){
    arr[0][0] = 1000;
}
```



→ java is Pass by value only. you Pass the stack value to the function.

Today's agenda

- Prefix Sum.
- Sum in a range for multiple query.
- Equilibrium index.
- Product of array except itself
- number of odd numbers in given range. ← easy ← HW
- Rain water trapping → Hard leetcode

Q) Given N array elements, return pf[] where pf[i] = Sum{arr[0], arr[1], ... arr[i]}, for all i.

Ex: arr[5] = { 4 1 6 -2 7 }
(0 1 2 3 4)

pf[5] = { 4 5 11 9 16 }

arr[10] = { -2 5 1 3 4 1 7 -8 2 0 }
(0 1 2 3 4 5 6 7 8 9)

pf[10] = { -2 3 4 7 11 12 19 11 13 13 }

// Brute force

↳ for every index i, run a loop from 0th to ith index and add all the elements.

arr[5] = { 4 1 6 -2 7 }
(0 1 2 3 4) ← i

pf[5] = { 4 5 11 9 16 }

//Pseudo Code

```
int[] PrefixSum (int arr[N]){
    int Pf[N];
    for (int i=0; i<N; i++){
        int Sum=0;
        for (int j=0; j<=i; j++){
            Sum = Sum + arr[j];
        }
        Pf[i] = Sum;
    }
    return Pf;
}
```

T.C: $O(N^2)$

S.C: $O(1)$

i

arr[5] = { 4 1 6 -2 7 }

Pf[5] : 4 5 11

Sum: 0+4+1+6

//Optimal approach

↳ arr[N]

pf[0] = arr[0]

pf[1] = arr[0] + arr[1] = pf[0] + arr[1]
↳ pf[0]

pf[2] = arr[0] + arr[1] + arr[2] = pf[1] + arr[2]
↳ pf[1]

pf[3] = arr[0] + arr[1] + arr[2] + arr[3]
↳ pf[2]
= pf[2] + arr[3]

∴ i

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

pf[5] : { 4 5 11 9 16 }

pf[i] = pf[i-1] + arr[i] for all i

//Psuedo Code

```
int[] PrefixOptimal(int arr[N]){
    int pf[N];
    pf[0] = arr[0];

    for(int i=1; i<N; i++){
        pf[i] = pf[i-1] + arr[i];
    }

    return pf;
}
```

T.C: O(N)

S.C: O(1)

Q) Given N array elements and Q queries on an array. for each query calculate sum of all elements in given range.

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

Q = 5

i	0	1	Sum
0	3	8	9
1	2	4	8
2	0	3	7
3	5	9	
4	6	6	

//Ideal (Brute force)

Iterate in the range for every query.

```
void SumQuery(int arr[N], int queries[Q][2]){
    for(int i=0; i<Q; i++){
        int L = Queries[i][0];
        int R = Queries[i][1];

        int Sum = 0;
        for(int j=L; j<=R; j++){
            Sum = Sum + arr[j];
        }

        Print(Sum);
    }
}
```

T.C : O(Q*N)

S.C : O(1)

Optimal approach

	0	1	2	3	4	5	6	7	8	9
arr[10]:	-2	5	1	3	4	1	7	-8	2	0
pf[10]:	-2	3	4	7	11	12	19	11	13	13

Sum(3, 8) = pf[8] - pf[2]
= Sum(0, 8) - Sum(0, 2)

Sum(2, 4) = pf[4] - pf[1]
= Sum(0, 4) - Sum(0, 1)

Sum(0, 3) = pf[3]
= Sum(0, 3)

Sum(L, R) = pf[R] - pf[L-1] if (L > 0)
= Sum(0, R) - Sum(0, L-1)

Sum(0, R) = pf[R] if (L == 0)
= Sum(0, R)

//Pseudo code

```
void SumQuery(int arr[N], int queries[Q][2]){
    int pf[] = PrefixOptimal(arr);
    for(int i=0; i<Q; i++){
        int L = Queries[i][0];
        int R = Queries[i][1];
        if (L>0) {
            Print(pf[R] - pf[L-1]);
        }
        else{ Print(pf[R]); }
    }
}
```

T.C : $O(N) + O(Q)$
$\simeq O(N+Q)$

S.C : $O(N)$

$O(N*Q)$ | $O(N+Q)$

$N == Q$

$O(N^2)$ | $O(N+N) = O(2N) \simeq O(N)$

Q) Equilibrium Index (pivot)

↳ Given N array elements, count no. of equilibrium index. An index i is said to be equilibrium index if?

Sum of all the elements before ith index = Sum of all the elements after ith index.

Sum(0, i-1) = Sum(i+1, N-1)

Ex: arr[4]:

	0	1	2	3
	-2	6	3	4
leftSum:	0	-2	4	7
rightSum:	13	7	4	0

⇒ ans = 1

arr[7]:

	0	1	2	3	4	5	6
	-7	1	5	2	-4	3	0
leftSum:	0	-7	-6	-1	1	-3	0
rightSum:	7	6	1	-1	3	0	0

⇒ ans: 2

//Optimal approach

	0	1	2	3 (N-1)
arr[4]:	-2	6	3	4
pf[4]:	-2	4	7	11

leftSum: Sum(0, i-1) = pf[i-1];

rightSum: Sum(i+1, N-1) = pf[N-1] - pf[i]
= pf[N-1] - pf[i]

//Pseudo code

```
int pivotIdx(int arr[N]){
    int pf[] = Prefix(arr);
    int ans = 0;

    for(int i=1; i<N-1; i++){
        int leftSum = pf[i-1];
        int rightSum = pf[N-1] - pf[i];
        if(leftSum == rightSum){ ans++; }
    }
```

T.C: O(N)+O(N)
O(2N) ≈ O(N)
S.C: O(N)

```
    //0th index Check
    if (0 == p[n-1] - p[0]){
        ans++;
    }

    //last index Check
    if (p[n-2] == 0) {ans++;}

    return ans;
}
```

Q) Product of Array Except itself

↳ Given arr[N], return an array answer such that answer[i] is equal to the product of all the elements of nums except nums[i].

Note: you can't use division operation.

Ex: arr[4]: { 1 2 3 4 }
(0 1 2 3)

ans[4]: 24 12 8 6

PrefixProd[i] = { arr[0] * arr[1] ... * arr[i] }

↳ arr[4]: { 1 2 3 4 }
(0 1 2 3) — i

PrefProd[4]: { ~~1~~ ~~2~~ ~~6~~ ~~24~~ }
24 12 8 6

Suffix = 1*4 = 4*3
= 12*2
= 24

PrefProd[i] = PrefProd[i-1] * Suffix

PrefProd[3] = PrefProd[2] * 1;
PrefProd[2] = PrefProd[1] * 4;
PrefProd[1] = PrefProd[0] * 12;

//Pseudo code

```
int[] ProductExceptItself (int arr[n]){
    int PrefProduct[n];
    PrefProduct[0] = arr[0];
    for (int i=1; i<n; i++){
        PrefProduct[i] = PrefProduct[i-1] * arr[i];
    }
    int Suffix = 1;
    for (int i=n-1; i>0; i--){
        PrefProduct[i] = PrefProduct[i-1] * Suffix;
        Suffix = Suffix * arr[i];
    }
    PrefProduct[0] = Suffix;
    return PrefProduct;
}
```

T.C: O(2n)

S.C: O(1)

Tracing

```
int PofProduct[n];
PofProduct[0] = arr[0];
for(int i=1; i<n; i++){
    PofProduct[i] = PofProduct[i-1] * arr[i];
}
int suffix = 1;
for(int i=n-1; i>0; i--){
    PofProduct[i] = PofProduct[i-1] * suffix;
    suffix = suffix * arr[i];
}
PofProduct[0] = suffix;
```

i: 0 1 2 3

arr[4]: { 1 2 3 4 }

PofProduct[4]: { ~~1~~ ~~2~~ ~~6~~ ~~24~~ }
24 12 8 6

suffix = 1 * 4 = 4 * 3 = 12 * 2 = 24

Prefixmax and Suffixmax

Prefixmax

Ex: arr[12]:

0	1	2	3	4	5	6	7	8	9	10	11
2	1	3	2	1	2	4	3	2	1	3	1

pmax[]:

2	2	3	3	3	3	4	4	4	4	4	4

Prefixmax[i] = max(0, ..., i)

Suffixmax

Suffixmax[i] = max(i, N-1)

arr[12]:

0	1	2	3	4	5	6	7	8	9	10	11
2	1	3	2	1	2	4	3	2	1	3	1

smax[12]:

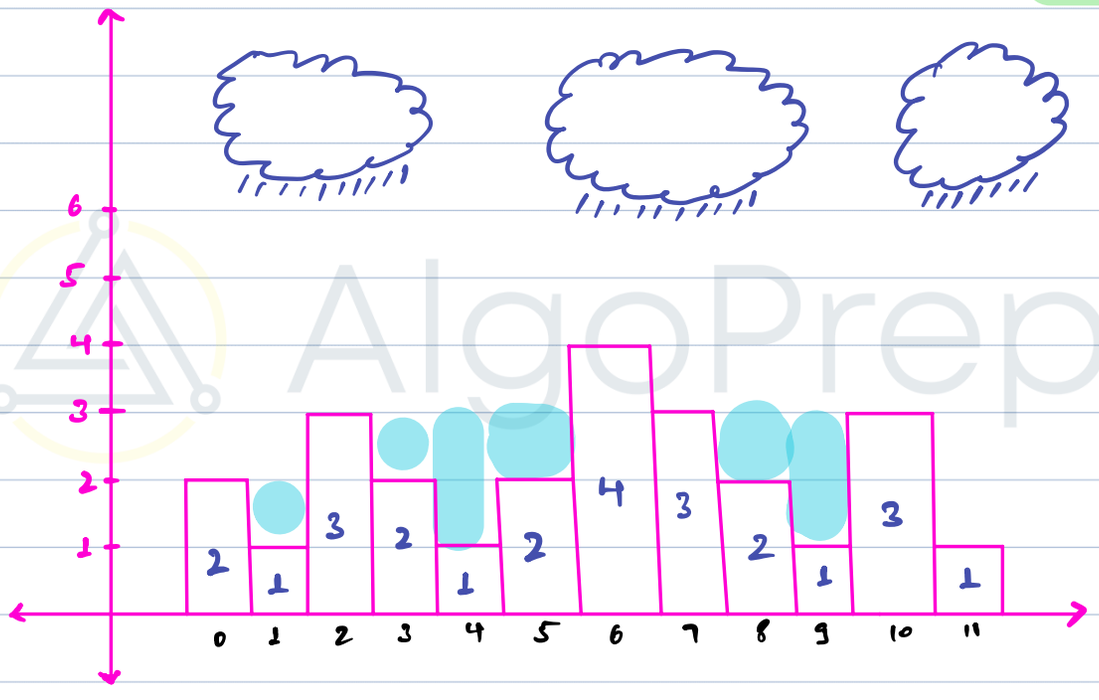
4	4	4	4	4	4	4	3	3	3	3	1

Q) Rain water trapping

↳ Given N non-negative integers representing an elevation map where the width of each bar is 1. Compute how much water it can trap after raining.

Eg: arr[12]: 2 1 3 2 1 2 4 3 2 1 3 1

↳ ans=8

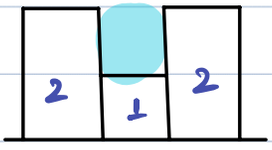


↳ Calculate amount at each wall and add them.

↓

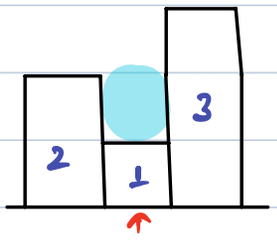
ans

Ex:



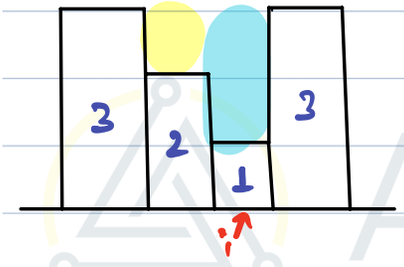
lb = 2
rb = 2

yb = 2
amount = 2 - 1 = 1



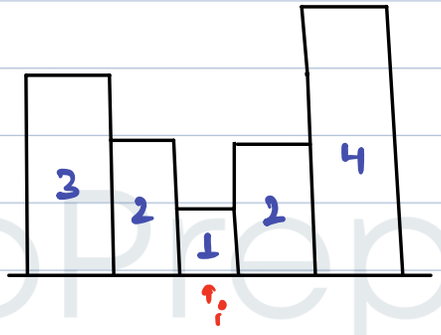
lb = 2
rb = 3

yb = min(lb, rb) = 2
amount = 2 - 1 = 1



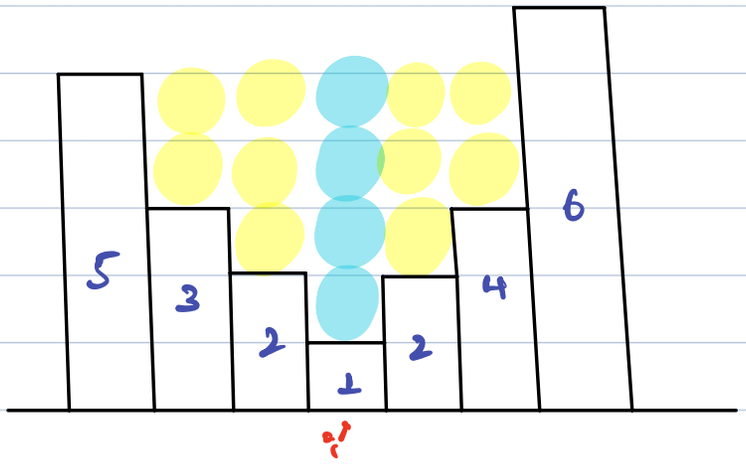
rb = 3
lb = 3

yb = min(lb, rb) = 3
amount = 3 - 1 = 2



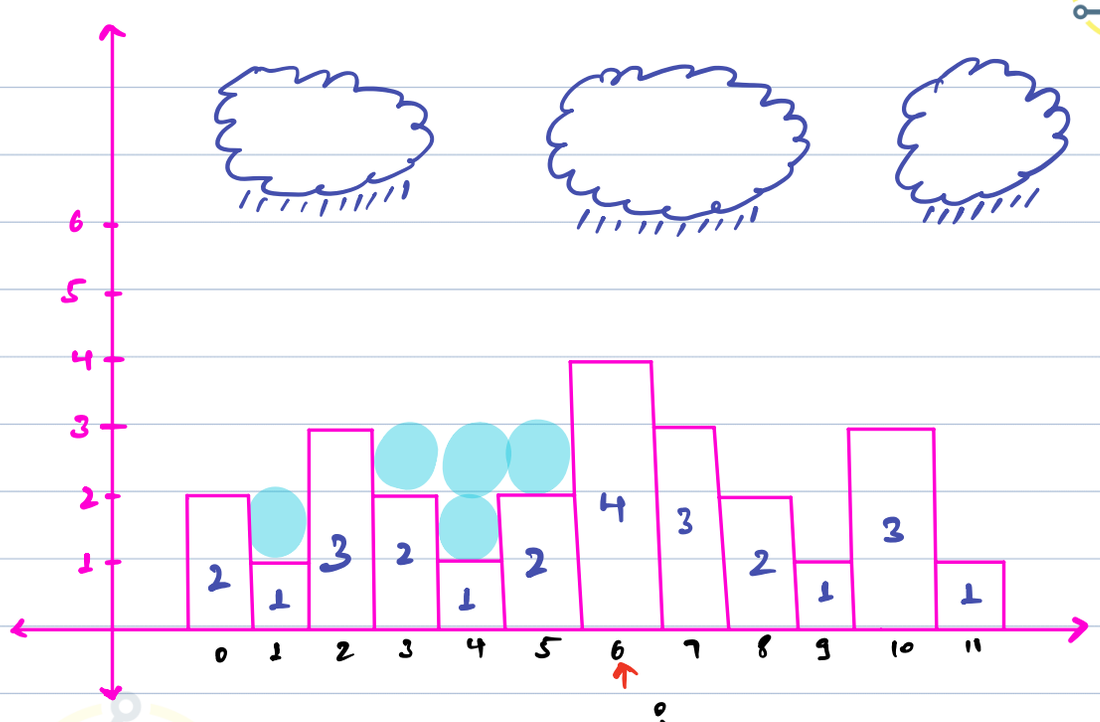
lb = 3
rb = 4

yb = min(lb, rb) = 3
amount = 3 - 1 = 2



lb = max height on left = 5
rb = max height on right = 6

yb = min(lb, rb) = min(5, 6) = 5
amount = 5 - 1 = 4



i

	0	1	2	3	4	5	6	7	8	9	10	11
arr[]:	2	1	3	2	1	2	4	3	2	1	3	1
Pmax[]:	2	2	3	3	3	3	4	4	4	4	4	4
Smax[]:	4	4	4	4	4	4	4	3	3	3	3	1

lb = Pmax[i-1] = 3

rb = Smax[i+1] = 4

yb = min(3,4) = 3

cont = 3 - 2 = 1

amount = 0+1 = 1+0 = 1+1 = 2+2 = 4+1 = 5

// Pseudo code

```
int Rainwater (int arr[N]) {

    int Pmax[] = func1(arr);   // HW
    int Smax[] = func2(arr);

    int amount = 0;
    for(int i = 1; i < N-1; i++) {
        int lb = Pmax[i-1];
        int rb = Smax[i+1];
        int yb = min(lb, rb);
        int conti = yb - arr[i];

        if (conti > 0) {
            amount = amount + conti;
        }
    }

    return amount;
}
```

T.C : O(3*N) : O(N)

S.C : O(2N) : O(N)

↓

O(1)

↓

Two Pointers

```
int[] func2(int arr[N]){

    int[] smax = new int[ N ];
    smax[N-1] = arr[N-1];

    for(int i=N-2 ; i>=0; i--){
        smax[i] = Math.max(smax[i+1],arr[i]);
    }

    return smax;

}
```

Running Sum of 1D Array

Java Code:

```
import java.util.Arrays;

public class RunningSum {
    public static int[] runningSum(int[] nums) {
        int n = nums.length;
        int[] pre = new int[n];
        pre[0] = nums[0];
        for (int i = 1; i < n; i++) {
            pre[i] = pre[i - 1] + nums[i];
        }
        return pre;
    }
}
```

C++ Code:

```
#include <iostream>
#include <vector>

std::vector<int> runningSum(std::vector<int>& nums) {
    int n = nums.size();
    std::vector<int> pre(n);
    pre[0] = nums[0];
    for (int i = 1; i < n; i++) {
        pre[i] = pre[i - 1] + nums[i];
    }
    return pre;
}

int main() {
    // Example usage
    std::vector<int> nums = {1, 2, 3, 4, 5};
    std::vector<int> result = runningSum(nums);

    // Print the result
    for (int num : result) {
        std::cout << num << " ";
    }
```

```
    std::cout << std::endl;

    return 0;
}
```

Python Code:

```
def runningSum(nums):
    n = len(nums)
    pre = [0] * n
    pre[0] = nums[0]
    for i in range(1, n):
        pre[i] = pre[i - 1] + nums[i]
    return pre

def main():
    # Example usage
    nums = [1, 2, 3, 4, 5]
    result = runningSum(nums)

    # Print the result
    print(result)

if __name__ == "__main__":
    main()
```

Range Sum Query - Immutable

Java Code:

Java

```java
class NumArray {
int[] psum;
public NumArray(int[] nums) {
    psum = new int[nums.length];
    psum[0] = nums[0];
    for(int i = 1; i < nums.length; i++)
```

```
            psum[i] = psum[i - 1] + nums[i];


    }

    public int sumRange(int i, int j) {
        if(i == 0)
            return psum[j];

        return psum[j] - psum[i - 1];
    }
    }
```

C++ Code:

```
#include <iostream>
#include <vector>

class NumArray {
public:
    std::vector<int> pref;

    NumArray(std::vector<int>& nums) {
        pref.push_back(0);
        for (int i = 1; i <= nums.size(); i++)
            pref.push_back(nums[i - 1] + pref[i - 1]);
    }

    int sumRange(int left, int right) {
        right++, left++;
        return pref[right] - pref[left - 1];
    }
};

int main() {
    std::vector<int> nums = {-2, 0, 3, -5, 2, -1};
    NumArray numArr(nums);

    // Example usage of the sumRange method
```

```
    std::cout << "Sum of elements from index 0 to 2: " << numArr.sumRange(0, 2) << std::endl; // Output: 1
    std::cout << "Sum of elements from index 2 to 5: " << numArr.sumRange(2, 5) << std::endl; // Output: -1
    std::cout << "Sum of elements from index 0 to 5: " << numArr.sumRange(0, 5) << std::endl; // Output: -3

    return 0;
}
```

Python Code:

```
class NumArray:
    def __init__(self, nums):
        self.pref = [0]
        for i in range(1, len(nums) + 1):
            self.pref.append(nums[i - 1] + self.pref[i - 1])

    def sumRange(self, left, right):
        right += 1
        left += 1
        return self.pref[right] - self.pref[left - 1]

# Example usage:
nums = [-2, 0, 3, -5, 2, -1]
numArr = NumArray(nums)

print("Sum of elements from index 0 to 2:", numArr.sumRange(0, 2))  # Output: 1
print("Sum of elements from index 2 to 5:", numArr.sumRange(2, 5))  # Output: -1
print("Sum of elements from index 0 to 5:", numArr.sumRange(0, 5))  # Output: -3
```

Find Pivot Index

Java Code:

```
import java.util.*;

class Solution {
    public int pivotIndex(int[] nums) {
        int n = nums.length;
```

```
        int totalSum = 0;
        for (int num : nums) {
            totalSum += num;
        }

        int idx = -1;
        int currSum = 0;

        for (int i = 0; i < n; i++) {
            if (currSum == totalSum - currSum - nums[i]) {
                idx = i;
                break;
            }
            currSum += nums[i];
        }
        return idx;
    }
}
}
```

C++ Code:

```
#include <iostream>
#include <vector>
using namespace std;
class Solution {
public:
    int pivotIndex(std::vector<int>& nums) {
        int n = nums.size();
        int sum = 0;
        for (auto x : nums) {
            sum += x;
        }

        int idx = -1;
        int curr_sum = 0;

        for (int i = 0; i < n; i++) {
            if (curr_sum == sum - curr_sum - nums[i]) {
                idx = i;
                break;
            }
            curr_sum += nums[i];
        }
        return idx;
    }
```

```
};

int main() {
    Solution solution;
    int n;
    cin>>n;
    std::vector<int> nums(n);
    for(int i=0;i<n;i++)
                    cin>>nums[i];
    int pivotIdx = solution.pivotIndex(nums);
    std::cout << "Pivot index: " << pivotIdx << std::endl; // Output: 3 (nums[3] = 6)

    return 0;
}
```

Python Code:

```
class Solution:
    def pivotIndex(self, nums):
        n = len(nums)
        total_sum = sum(nums)

        idx = -1
        curr_sum = 0

        for i in range(n):
            if curr_sum == total_sum - curr_sum - nums[i]:
                idx = i
                break
            curr_sum += nums[i]
        return idx

# Example usage:
solution = Solution()
nums = [1, 7, 3, 6, 5, 6]
pivot_idx = solution.pivotIndex(nums)
print("Pivot index:", pivot_idx)  # Output: 3 (nums[3] = 6)
```

Product of Array except Itself

Java Code:

```java
class Solution {
    public int[] productExceptSelf(int[] nums) {
    int n = nums.length;
    int[] res = new int[n];
    res[0] = 1;
    for (int i = 1; i < n; i++) {
        res[i] = res[i - 1] * nums[i - 1];
    }
    int right = 1;
    for (int i = n - 1; i >= 0; i--) {
        res[i] *= right;
        right *= nums[i];
    }
    return res;
}
}
```

C++ Code:

```cpp
#include <vector>

class Solution {
public:
    std::vector<int> productExceptSelf(std::vector<int>& nums) {
        int n = nums.size();
        std::vector<int> res(n, 1);

        int left = 1;
        for (int i = 0; i < n; i++) {
            res[i] *= left;
            left *= nums[i];
        }
```

```
        int right = 1;
        for (int i = n - 1; i >= 0; i--) {
            res[i] *= right;
            right *= nums[i];
        }

        return res;
    }
};
```

Python Code:

```
from typing import List

class Solution:
    def productExceptSelf(self, nums: List[int]) -> List[int]:
        n = len(nums)
        res = [1] * n

        left = 1
        for i in range(n):
            res[i] *= left
            left *= nums[i]

        right = 1
        for i in range(n - 1, -1, -1):
            res[i] *= right
            right *= nums[i]

        return res
```

Trapping Rain Water

Java Code:

```java
class Solution {
    public int trap(int[] arr) {
        int n = arr.length;

        int[] prefixmax = new int[n];
        int[] suffixmax = new int[n];

        prefixmax[0] = arr[0];
        for(int i=1;i<n;i++){
            prefixmax[i] = Math.max(prefixmax[i-1],arr[i]);
        }

        suffixmax[n-1] = arr[n-1];
        for(int i=n-2;i>=0;i--){
            suffixmax[i] = Math.max(suffixmax[i+1],arr[i]);
        }

        int ans = 0;
        for(int i =1; i<n;i++){
            ans+= Math.min(prefixmax[i],suffixmax[i]) - arr[i];
        }

        return ans;

    }
}
```

C++ Code:

#include <vector>

class Solution {

```
public:
    int trap(std::vector<int>& arr) {
        int n = arr.size();

        std::vector<int> prefixmax(n);
        std::vector<int> suffixmax(n);

        prefixmax[0] = arr[0];
        for (int i = 1; i < n; i++) {
            prefixmax[i] = std::max(prefixmax[i - 1], arr[i]);
        }

        suffixmax[n - 1] = arr[n - 1];
        for (int i = n - 2; i >= 0; i--) {
            suffixmax[i] = std::max(suffixmax[i + 1], arr[i]);
        }

        int ans = 0;
        for (int i = 1; i < n; i++) {
            ans += std::min(prefixmax[i], suffixmax[i]) - arr[i];
        }

        return ans;
    }
};
```

Python Code:

```
class Solution:
    def trap(self, arr):
        n = len(arr)

        prefix_max = [0] * n
        suffix_max = [0] * n

        prefix_max[0] = arr[0]
        for i in range(1, n):
            prefix_max[i] = max(prefix_max[i - 1], arr[i])

        suffix_max[n - 1] = arr[n - 1]
        for i in range(n - 2, -1, -1):
            suffix_max[i] = max(suffix_max[i + 1], arr[i])

        ans = 0
        for i in range(1, n):
```

```
        ans += min(prefix_max[i], suffix_max[i]) - arr[i]

    return ans
```

Count Odd Numbers_HW

Java Code:

```java
public int countOdds(int low, int high) {
        return (high + 1) / 2 - low / 2;
 }
```

C++ Code:

```cpp
int countOdds(int low, int high) {
    return (high + 1) / 2 - low / 2;
}
```

Python Code:

```python
def countOdds(low, high):
    return (high + 1) // 2 - low // 2
```

Range Addition_HW

Solution Vid:

https://youtu.be/XWkB_p0RBjc

Java Code:

```
Java
class Solution {
```

```java
    public int[] getModifiedArray(int length, int[][] updates) {
        int[] result = new int[length];

        for(int i = 0 ; i < updates.length ; i++){
            int st = updates[i][0];
            int end = updates[i][1];
            int incr = updates[i][2];
            result[st] += incr;

            if(end + 1 < length){
                result[end + 1] -= incr;
            }
        }

        int[] psum = new int[length];
        psum[0] = result[0];
        for(int i = 1 ; i < length ; i++){
            psum[i] = psum[i-1] + result[i];
        }

        return psum;
    }
}
```

C++ Code:

```cpp
class Solution {
public:
    vector<int> getModifiedArray(int length, vector<vector<int>>& updates) {
        vector<int> result(length, 0);

        for (int i = 0; i < updates.size(); i++) {
            int start = updates[i][0];
            int end = updates[i][1];
            int increment = updates[i][2];
            result[start] += increment;

            if (end + 1 < length) {
                result[end + 1] -= increment;
```

```
        }
    }

    vector<int> prefixSum(length, 0);
    prefixSum[0] = result[0];
    for (int i = 1; i < length; i++) {
        prefixSum[i] = prefixSum[i - 1] + result[i];
    }

    return prefixSum;
  }
};
```

Python Code:

```
class Solution:
    def getModifiedArray(self, length: int, updates: List[List[int]]) -> List[int]:
        result = [0] * length

        for update in updates:
            start, end, increment = update[0], update[1], update[2]
            result[start] += increment

            if end + 1 < length:
                result[end + 1] -= increment

        prefix_sum = [0] * length
        prefix_sum[0] = result[0]
        for i in range(1, length):
            prefix_sum[i] = prefix_sum[i - 1] + result[i]

        return prefix_sum
```

Today's agenda

- Subarrays recap.
- Subarrays questions.
- Contribution technique
- Number of subarray with bound
- Kadane's algo

Subarrays:

↳ Continuous Part of an array is Subarray.

(i) single element of an array? ✔

(ii) Complete array is also a subarray? ✔

(iii) empty array? ✗✗

(iv) {1, 2, 3, 4} → {3, 2} ??

↳ No, you can't reverse the order in subarray.

* Total no. of subarray in an array of length N.

↳ $\frac{N*(N+1)}{2}$

→ arr[5] = { 10 (0) 20 (1) 30 (2) 40 (3) 50 (4) }

Start 0th idx	Start 1st idx	Start 2nd idx	Start at last idx
10	20	30	
10 20	20 30	30 40	50
10 20 30	20 30 40	30 40 50	↓
10 20 30 40	20 30 40 50	3 : N-2	!
10 20 30 40 50	4 : N-1		...
5 : N			

Total Count : $N + (N-1) + (N-2) \ldots\ldots\ldots + 1$

$\frac{N*(N+1)}{2}$

Q) Given an arr[N], Print subarray from [s, e].

ex: arr[4] : { 5, 3, -1, 8 }
0 1 2 3

s = 1 e = 3

O/P 3 -1 8

```
for (int i=s; i<=e; i++){
    Print(arr[i]);
}
```

T.C: O(N)

Q) Given N array elements Print each and every subarray.

Ex: arr[4]: { 5, 3, -1, 8 }
(0, 1, 2, 3)

S=0		S=1		S=2		S=3
5	{0,0}	3	{1,1}	-1	{2,2}	8
5 3	{0,1}	3 -1	{1,2}	-1 8	{2,3}	
5 3 -1	{0,2}	3 -1 8	{1,3}			
5 3 -1 8	{0,3}					

```
void PrintSubArrays (int arr[N]){

    for(int s=0; s<N; s++){
        for(int e=s; e<N; e++){
            //[s,e] subarray
            for(int i=s; i<=e; i++){
                Print(arr[i]);
            }
        }
    }

}
```

T.C: O(N^3)

S.C: O(1)

Tracing

```
for (int s=0; s<N; s++){
    for (int e=s; e<N; e++){
        //[s,e] Sub array
        for (int i=s; i<=e; i++){
            Print(arr[i]);
        }
    }
}
```

arr[4]: { 5, 3, -1, 8 }
(0, 1, 2, 3)

s→0 → e=0 ⇒ 5
↳ e=1 ⇒ 5,3
↳ e=2 ⇒ 5,3,-1
↳ e=3 ⇒ 5,3,-1,8

s→1 → e=1 ⇒ 3

Q) Given N array elements, Print each subarray sum

Ex: arr[4] : { 5, 3, -1, 8 }

S=0	S=1	S=2	S=3
5 {0,0} → 5	3 {1,1} → 3	-1 {2,2} → -1	8 {3,3} → 8
5 3 {0,1} → 8	3 -1 {1,2} → 2	-1 8 {2,3} → 7	
5 3 -1 {0,2} → 7	3 -1 8 {1,3} → 10		
5 3 -1 8 {0,3} → 15			

```
void PrintSubArrays (int arr[N]){
    for(int s=0; s<N; s++){
        for(int e=s; e<N; e++){
            //[s,e] subarray
            int sum=0;
            for(int i=s; i<=e; i++){
                sum = sum+arr[i];
            }
            Print(sum);
        }
    }
}
```

T.C: O(N^3)

S.C: O(1)

//idea2

↳ Prefix Sum idea

Ex: arr[4] : { 5, 3, -1, 8 }

S=0	S=1	S=2	S=3
5 {0,0} → 5	3 {1,1} → 3	-1 {2,2} → -1	8 {3,3} → 8
5 3 {0,1} → 8	3 -1 {1,2} → 2	-1 8 {2,3} → 7	
5 3 -1 {0,2} → 7	3 -1 8 {1,3} → 10		
5 3 -1 8 {0,3} → 15			

Pf[4] : { 5, 8, 7, 15 }

Sum(i,j) : Psum[j] - Psum[i-1]

//Pseudo code

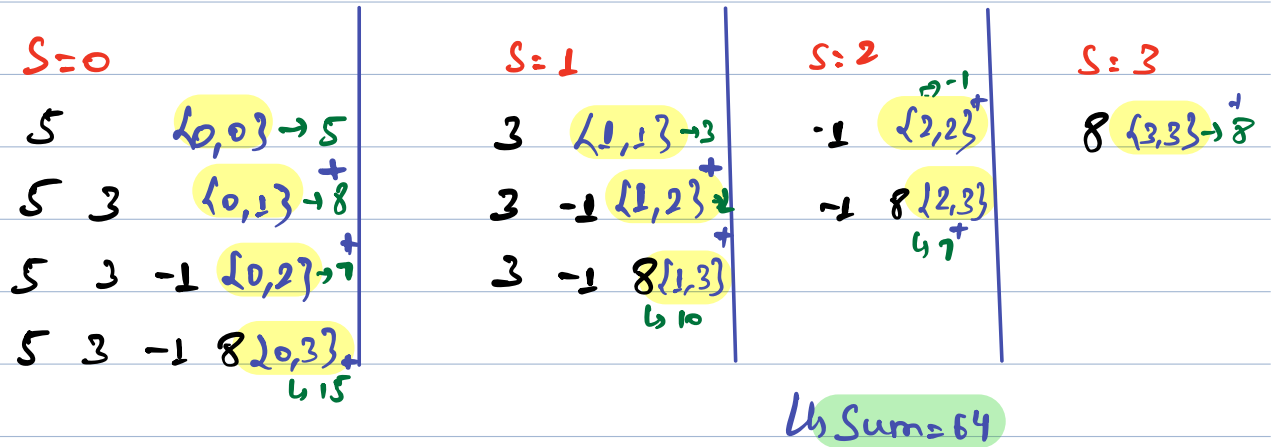
```
void PrintSubArrays(int arr[N]){
    int Psum[] = PrefixSum(arr);

    for(int s=0; s<N; s++){
        for(int e=s; e<N; e++){
            //[s,e] subarray
            if(s==0){ Print(Psum[e]); }

            else{ Print(Psum[e] - Psum[s-1]); }
        }
    }
}
```

T.C : O(N) + O(N²) = O(N²)

S.C = O(N)

Q) Given arr[N] elements, return Sum of all subarray sums

ex: arr[4] : {5 3 -1 8} (0 1 2 3)



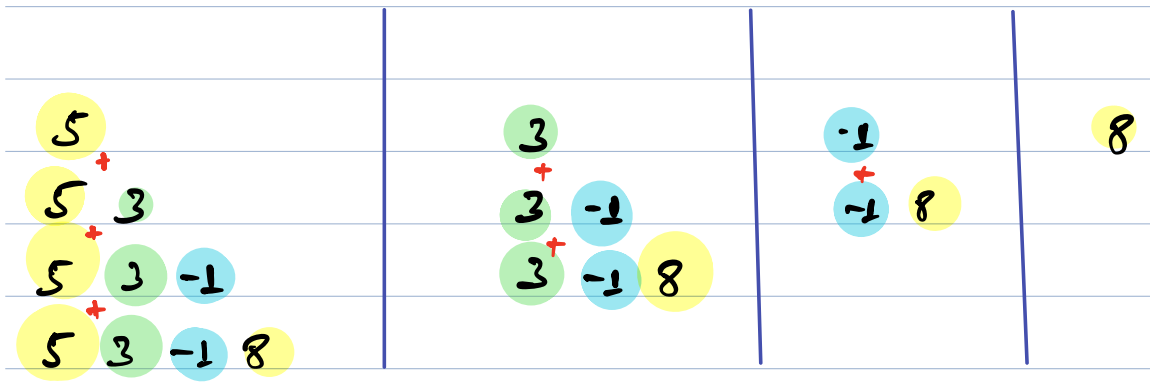

```
void PrintSubArrays(int arr[N]){
    int Psum[] = PrefixSum(arr);
    int ans=0;
    for(int s=0; s<N; s++){
        for(int e=s; e<N; e++){
            //[s,e] subarray
            if(s==0){ans=ans+Psum[e];}
            else{ans= ans+Psum[e]-Psum[s-1];}
        }
    }
    Print(ans);
}
```

T.C: O(N) + O(N²) = O(N²)

S.C = O(N)

//idea2

arr[4] : { 5 (0) 3 (1) -1 (2) 8 (3) }



↳ ans : 0

occ.

↳ arr[0] : 5 ⟶ 4 times

↳ 5 * 4 = 20

↳ arr[1] : 3 ⟶ 6 times

↳ 3 * 6 = 18

↳ arr[2] : -1 ⟶ 6 times

↳ -1 * 6 = -6

↳ arr[3] : 8 ⟶ 4 times

↳ 8 * 4 = 32

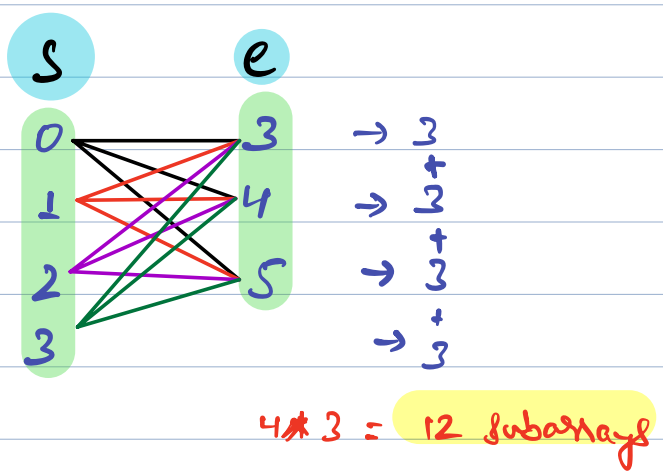
ans : 64

arr[4] : { 5 3 -1 8 }
0 1 2 3
↓ ↓ ↓ ↓
n0 n1 n2 n3

ans = (arr[0] * n0) + (arr[1] * n1) + (arr[2] * n2)
+ (arr[3] * n3)

→ Finding occurence

ex: { 3 -2 4 -1 2 6 }
0 1 2 3 4 5



4 * 3 = 12 subarrays

ex: [3 (0) | -2 (1) | 4 (2) | -1 (3) | 2 (4) | 6 (5)]

S	e
0	1
1	2
	3
	4
	5

⇒ 2 * 5 = 10 subarrays

general

ex: [3 (0) | -2 (1) | 4 (2) ← i | -1 (3) | 2 (4) | 6 (5)]

No. of occ = (No. of valid S) * (No. of valid e)

(No. of valid S) → [0, i] → (i+1) elements

(No. of valid e) → [i, N-1] → N-1-i+1 = (N-i) elements

Total no. of occ of ith index = (i+1) * (N-i)

//Pseudo code

```
int TotalSum(int arr[N]){
    int ans=0;
    for(int i=0; i<N; i++){
        int occ=(i+1)*(N-i);
        ans = ans + (occ*arr[i]);
    }
    return ans;
}
```

T.C: O(N)

S.C: O(1)

Tracing

arr[4]: {5 3 -1 8} (0 1 2 3)

```
int ans=0;
for(int i=0; i<N; i++){
    int occ=(i+1)*(N-i);
    ans = ans + (occ*arr[i]);
}
return ans;
```

occ = 4*1

ans = 0 + (5*4) + (3*6) + (-1*6) + 8*4

= 0 + 20 + 18 - 6 + 32

= 64

II/ Contribution technique

Q) number of subarrays with bounded maximum

Ls Given arr[N] and two integers left and right. Return the number of subarrays such that value of the maximum array element in that subarray is in the range {left, right}.

Ex1: arr[4]: { 2 1 4 3 } (0 1 2 3) left=2 Right=3

2
2 1
2 1 4
2 1 4 3

1
1 4
1 4 3

4
4 3

3

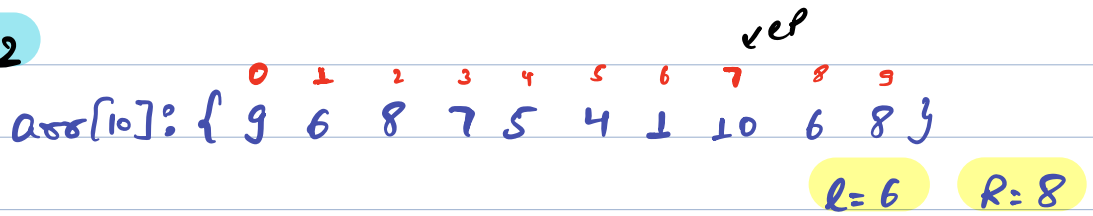
Ls ans=3

//Idea1

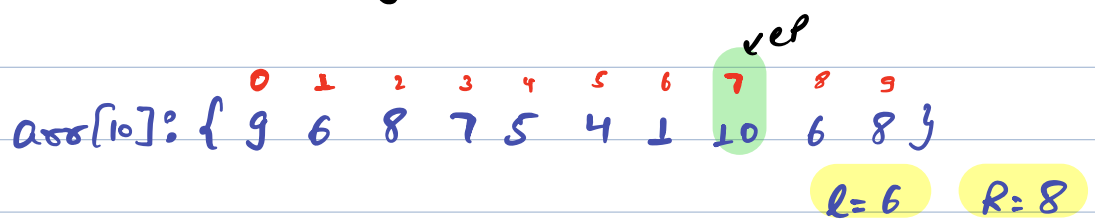
Ls create all the subarrays, Calculate the max and Pick the valid once.

T.C : $O(N^2) * O(N) \approx O(N^3)$

//idea2

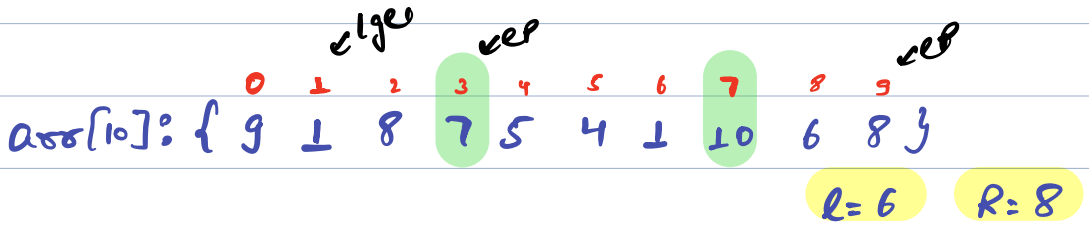


Case 1: arr[ep] > R



validStartingPoints = 0

Case 2: arr[ep] >= L && arr[ep] <= R

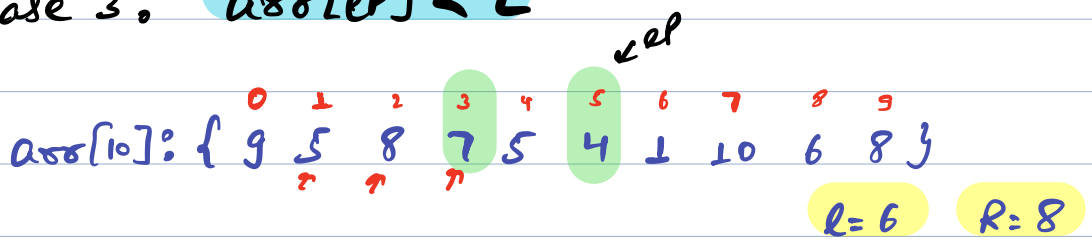


validStartingPoints = [lgei, ep]

↓ last greater than Right element index + 1.

ep - lgei + 1

Case 3: arr[ep] < L



validStartingPoints: no. of valid Starting Points for the element in the range of [L,R] just before the current element.

//Pseudocode

```
int validSubarray(arr[N], int left, int right){
    int ans = 0;
    int lgei = 0;
    int PervalidCount = 0;
    for(int ep=0; ep<N; ep++){
        if(arr[ep] > R){
            ans = ans + 0;        PervalidCount=0;
            lgei = ep+1;
        }
        else if(arr[ep] >=L && arr[ep]<=R){
            ans = ans + (ep - lgei + 1);
            PervalidCount = ep - lgei + 1;
        }
        else{ // arr[ep] < L
            ans = ans + PervalidCount;
        }
    }
}
```

T.C: O(N)

S.C: O(1)

Tracing

```
int validSubArray (arr[N], int left, int right){
    int ans = 0;
    int lgei = 0;
    int PervalidCount = 0;
    for (int ep=0; ep<N; ep++){
        if (arr[ep] > R){
            ans = ans + 0;        PervalidCount=0;
            lgei = ep+1;
        }
        else if (arr[ep] >= L && arr[ep]<=R){
            ans = ans + (ep - lgei + 1);
            PervalidCount = ep - lgei + 1;
        }
        else{ // arr[ep] < L
            ans = ans + PervalidCount;
        }
    }
```

arr[10]: { 9 6 8 7 5 4 1 10 6 8 }

indices: 0 1 2 3 4 5 6 7 8 9

ep → 3 (value 7)

lgei → 7 (value 10)

L = 6 R = 8

ans = 0+0+1+2+3+3+3+3+0+1+2

PerValidCount = 0 1 2 3 0 1 2

Q) max subarray sum → {Kadane's algo}

↳ Given N array elements, calculate max subarray sum.

Ex1: arr[7]: {3 2 -6 8 2 9 4}

↳ ans: 23

Ex2: arr[7]: {-3 2 4 -1 3 -4 3}

↳ ans: 8

//idea1

↳ generate all subarray sum and find max out of them.

T.C: $O(N^2)$ S.C: $O(N)$

//idea2 → {Kadane's algo}

{ +ve +ve +ve +ve +ve } → Sum of all these

{ -ve -ve +ve +ve +ve } → Sum of last three

{ +ve -ve +ve +ve +ve }

(-ve: < 0)

	0	1	2	3	4	5	6
arr[7] :	3	4	2	-14	16	-20	5
Sum : 0	3	7	9	-5	16	-4	5
ans = -∞	3	7	9	9	16	16	16

//Pseudo code

```
int Kadanes (int arr[N]) {
    int sum = 0;
    int ans = Integer.MIN_VALUE;

    for (int i=0; i<N; i++) {
        if (sum >= 0) {
            sum = sum + arr[i];
        }
        else {
            sum = arr[i];
        }
        ans = math.max (ans, sum);
    }
    return ans;
}
```

T.C : O(N)

S.C : O(1)

```
for (int i=0; i<N; i++) {
    if (sum >= 0) {
        sum = sum + arr[i];
    }
    else {
        sum = arr[i];
    }
    ans = Math.max(ans, sum);
}
return ans;
```

arr[7]: 3 4 2 -14 16 -20 5 (i: 0 1 2 3 4 5 6)

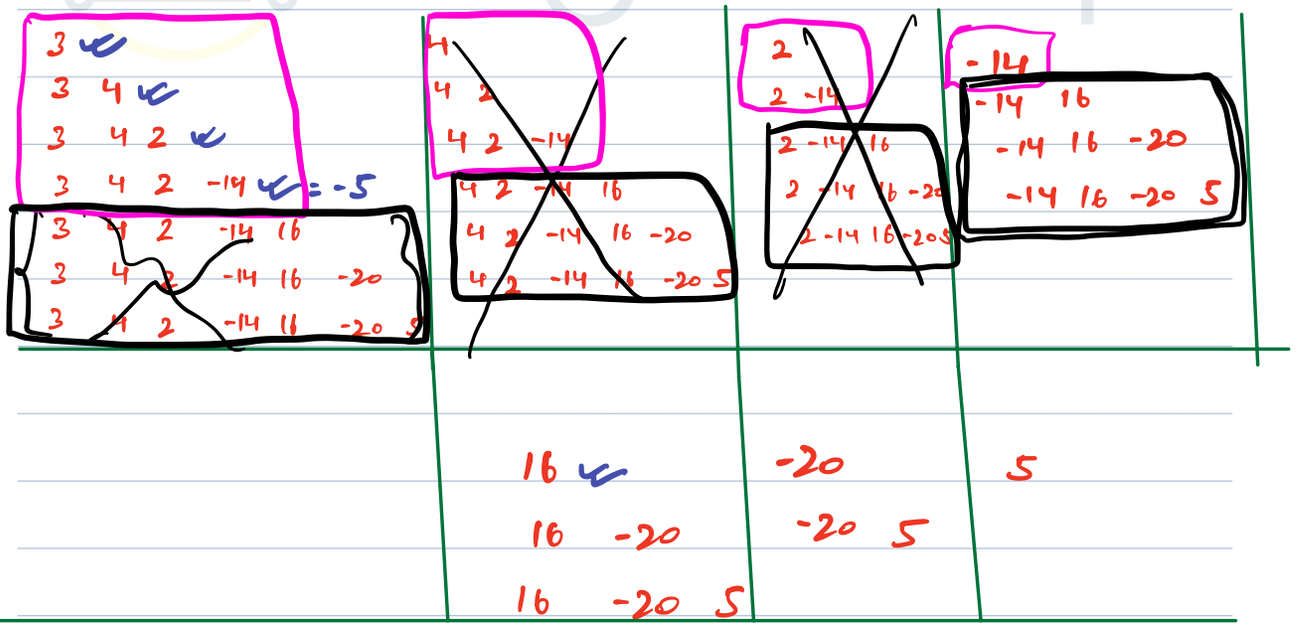
Sum = ~~0~~ 3 ~~7~~ ~~9~~ ~~-5~~ 16 ~~-4~~ 5

ans = ~~-∞~~ ~~3~~ ~~7~~ ~~9~~ ~~9~~ ~~16~~ ~~16~~ 16

Sum: 0 3 7 9 -5 16 -4 5

ans: -∞ 3 7 9 9 16 16 16

arr[7]: 3 4 2 -14 16 -20 5 (i: 0 1 2 3 4 5 6)



Print Subarray

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int[] arr = new int[n];

        for(int i=0;i<n;i++){
            arr[i] = scn.nextInt();
        }

        int si = scn.nextInt();
        int ei = scn.nextInt();

        for(int i=si;i<=ei;i++){
            System.out.print(arr[i]+" ");
        }
    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;

int main() {
    int n;
    cin >> n;
    int arr[n];
```

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

    int si, ei;
    cin >> si >> ei;

    for (int i = si; i <= ei; i++) {
        cout << arr[i] << " ";
    }

    return 0;
}
```

Python Code:

```
n = int(input())
arr = list(map(int, input().split()))

si, ei = map(int, input().split())

for i in range(si, ei + 1):
    print(arr[i], end=' ')
```

Print all the subarrays

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int[] arr = new int[n];
```

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

        for(int sp =0;sp < n;sp++){
            for(int ep = sp;ep<n;ep++){

                for(int i=sp;i<=ep;i++){
                    System.out.print(arr[i]+" ");
                }

                System.out.println();
            }
        }
    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;

int main() {
    int n;
    cin >> n;
    int arr[n];

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

    for (int sp = 0; sp < n; sp++) {
        for (int ep = sp; ep < n; ep++) {

            for (int i = sp; i <= ep; i++) {
                cout << arr[i] << " ";
            }

            cout << endl;
        }
    }

    return 0;
}
```

Python Code:

```
n = int(input())
arr = list(map(int, input().split()))

for sp in range(n):
    for ep in range(sp, n):
        for i in range(sp, ep + 1):
            print(arr[i], end=" ")
        print()
```

Sum of all SubArrays

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int[] arr = new int[n];

        for(int i=0;i<n;i++){
            arr[i] = scn.nextInt();
        }

        int sum = 0;

        for(int i=0; i<n;i++){
            int occ = (i+1)*(n-i);
            sum = sum + occ*arr[i];
        }

        System.out.println(sum);
```

```
    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;

int main() {
    int n;
    cin >> n;
    int arr[n];

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

    int sum = 0;

    for (int i = 0; i < n; i++) {
        int occ = (i + 1) * (n - i);
        sum = sum + occ * arr[i];
    }

    cout << sum << endl;

    return 0;
}
```

Python Code:

```
n = int(input())
arr = list(map(int, input().split()))

sum_val = 0

for i in range(n):
    occ = (i + 1) * (n - i)
    sum_val += occ * arr[i]

print(sum_val)
```

Number of Subarray with Bounded Maximum

Java Code:

```java
class Solution {
  public int numSubarrayBoundedMax(int[] A, int L, int R) {
          int lgei = 0;
        int lastvalidcount = 0;
        int ans = 0;

           for (int ep = 0; ep < A.length; ep++) {
                if (A[ep] >= L && A[ep] <= R) {
                     ans += ep - lgei + 1;
                     lastvalidcount = ep - lgei + 1;
                } else if (A[ep] < L) {
                     ans += lastvalidcount;
                } else {
                     lgei = ep + 1;
                     lastvalidcount = 0;
                }
           }
           return ans;
      }
}
```

C++ Code:

```
class Solution {
public:
   int numSubarrayBoundedMax(vector<int>& A, int L, int R) {
      int lgei = 0;
      int lastvalidcount = 0;
      int ans = 0;

      for (int ep = 0; ep < A.size(); ep++) {
         if (A[ep] >= L && A[ep] <= R) {
            ans += ep - lgei + 1;
```

```
                lastvalidcount = ep - lgei + 1;
            } else if (A[ep] < L) {
                ans += lastvalidcount;
            } else {
                lgei = ep + 1;
                lastvalidcount = 0;
            }
        }
        return ans;
    }
};
```

Python Code:

```
class Solution:
    def numSubarrayBoundedMax(self, A: List[int], L: int, R: int) -> int:
        lgei = 0
        lastvalidcount = 0
        ans = 0

        for ep in range(len(A)):
            if L <= A[ep] <= R:
                ans += ep - lgei + 1
                lastvalidcount = ep - lgei + 1
            elif A[ep] < L:
                ans += lastvalidcount
            else:
                lgei = ep + 1
                lastvalidcount = 0

        return ans
```

Maximum Sum SubArray

Java Code:

Java

```java
class Solution {
    public int maxSubArray(int[] nums) {
        int sum= 0;
        int ans=Integer.MIN_VALUE;

        for(int i=0;i<nums.length;i++){
            if(sum<0){
                sum=nums[i];
            }else{
                sum = sum +nums[i];
            }

            ans= Math.max(sum, ans);
        }
        return ans;
    }
}
```

C++ Code:

```cpp
class Solution {
public:
    int maxSubArray(vector<int>& nums) {
        int sum = 0;
        int ans = INT_MIN;

        for(int i = 0; i < nums.size(); i++) {
            if(sum < 0) {
                sum = nums[i];
            } else {
                sum += nums[i];
            }

            ans = max(sum, ans);
        }
        return ans;
    }
};
```

Python Code:

```
class Solution:
    def maxSubArray(self, nums: List[int]) -> int:
        sum_val = 0
        ans = float('-inf')

        for num in nums:
            if sum_val < 0:
                sum_val = num
            else:
                sum_val += num

            ans = max(sum_val, ans)

        return ans
```

Maximum Product SubArray

Solution vid:

https://youtu.be/0ab7Lid7tYA

Java Code:

Java

```java
class Solution {
    public int maxProduct(int[] nums) {
        int n = nums.length;

        int pf1 = 1;
        int max1 = Integer.MIN_VALUE;
        int pf2 = 1;
        int max2 = Integer.MIN_VALUE;

        for(int i=0;i<n;i++){
            if(pf1 == 0){
                pf1 = nums[i];
            }else{
                pf1 = pf1 * nums[i];
```

```
            }

            max1 = Math.max(max1,pf1);
        }

        for(int i = n-1;i>=0;i--){
            if(pf2 == 0){
                pf2 = nums[i];
            }else{
                pf2 = pf2 * nums[i];
            }

            max2 = Math.max(max2,pf2);
        }

        return Math.max(max1,max2);



    }
}
```

C++ Code:

```
class Solution {
public:
    int maxProduct(vector<int>& nums) {
        int n = nums.size();
        int pf1 = 1;
        int max1 = INT_MIN;
        int pf2 = 1;
        int max2 = INT_MIN;

        for (int i = 0; i < n; i++) {
            if (pf1 == 0) {
                pf1 = nums[i];
            } else {
                pf1 = pf1 * nums[i];
            }
```

```
            max1 = max(max1, pf1);
        }

        for (int i = n - 1; i >= 0; i--) {
            if (pf2 == 0) {
                pf2 = nums[i];
            } else {
                pf2 = pf2 * nums[i];
            }
            max2 = max(max2, pf2);
        }

        return max(max1, max2);
    }
};
```

Python Code:

```
class Solution:
    def maxProduct(self, nums: List[int]) -> int:
        n = len(nums)
        pf1, max1 = 1, float('-inf')
        pf2, max2 = 1, float('-inf')

        for i in range(n):
            if pf1 == 0:
                pf1 = nums[i]
            else:
                pf1 = pf1 * nums[i]
            max1 = max(max1, pf1)

        for i in range(n - 1, -1, -1):
            if pf2 == 0:
                pf2 = nums[i]
            else:
                pf2 = pf2 * nums[i]
            max2 = max(max2, pf2)

        return max(max1, max2)
```

Today's agenda

- Transpose
- Rotate matrix by 90 degrees
- Range sum query 2D
- Sum of all submatrix sum

Q) Transpose of a matrix

↳ Given a mat[N][N], calculate transpose of mat[][]
(convert every col into corresponding row.)

Note: you can't use a new matrix.

Ex:

	0	1	2	3
0	10	20	30	40
1	50	60	70	80
2	90	100	110	120
3	130	140	150	160

→

	0	1	2	3
0	10	50	90	130
1	20	60	100	140
2	30	70	110	150
3	40	80	120	160

(0,1) (1,0)
(1,0) (0,1)

(1,2) (2,1)
(2,1) (1,2)

$(i,j) \rightleftarrows (j,i)$

↳ Transpose is equivalent to swapping (i,j) & (j,i).

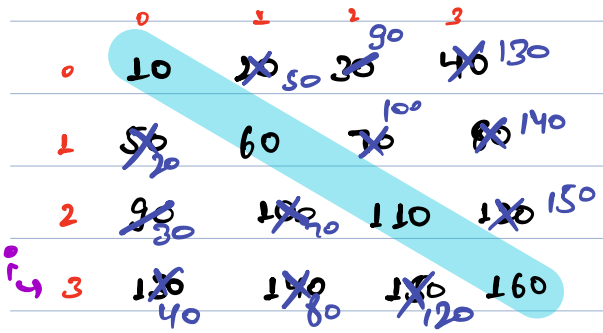
//Pseudo code

```
void transpose (int arr[N][N]){
    for (int i=1; i<N; i++){
        for (int j=0; j<i; j++){
            //Swap (i,j) with (j,i)
            int temp = arr[i][j]
            arr[i][j] = arr[j][i];
            arr[j][i] = temp;
        }
    }
}
```

T.C: $O(n^2)$

S.C: $O(1)$

//Tracing




```
for(int i=1; i<N; i++){
    for(int j=0; j<i; j++){
        //Swap (i,j) with (j,i)
        int temp = arr[i][j]
        arr[i][j] = arr[j][i];
        arr[j][i] = temp;
    }
}
```

i=1 → j=0 Swap (1,0) with (0,1)
↳ j=1 → exit

i=2 → j=0 Swap (2,0) with (0,2)
↳ j=1 Swap (2,1) with (1,2)
↳ j=2 → exit

i=3 → j=0 Swap (3,0) with (0,3)
↳ j=1 Swap (3,1) with (1,3)
↳ j=2 Swap (3,2) with (2,3)
↳ j=3 → exit

i=4 → exit

// what if matrix is rectangle?

mat

	0	1	2
0	1	2	3
1	4	5	6

2×3

transpose

	0	1
0	1	4
1	2	5
2	3	6

3×2

↳ matrix changed → It is not possible to take transpose within the same matrix.

↳ transpose[i][j] = mat[j][i];

Q) Rotate the matrix → {Google, Amazon}

↳ Rotate the given mat[N][N] clockwise by 90°

Ex:

	0	1	2	3
0	10	20	30	40
1	50	60	70	80
2	90	100	110	120
3	130	140	150	160

→

	0	1	2	3
0	130	90	50	10
1	140	100	60	20
2	150	110	70	30
3	160	120	80	40

Step1: Transpose of the matrix

	0	1	2	3
0	10	50	90	130
1	20	60	100	140
2	30	70	110	150
3	40	80	120	160

Step2: reverse every row →

	0	1	2	3
0	130	90	50	10
1	140	100	60	20
2	150	110	70	30
3	160	120	80	40

// Pseudo code

```
void Rotate 90° clockwise (int arr[N][N]){
    //Step1
    transpose (arr);

    //Step2
    for (int i=0; i<N; i++) {
        int[] temp = arr[i];
        reverse (temp); → {TODO}
    }
}
```

T.C: $O(N^2)$

S.C: $O(1)$

Q) Rotate the given mat[N][N] anti-clockwise by 90°

Ex:

	0	1	2	3
0	10	20	30	40
1	50	60	70	80
2	90	100	110	120
3	130	140	150	160

Step1: transpose

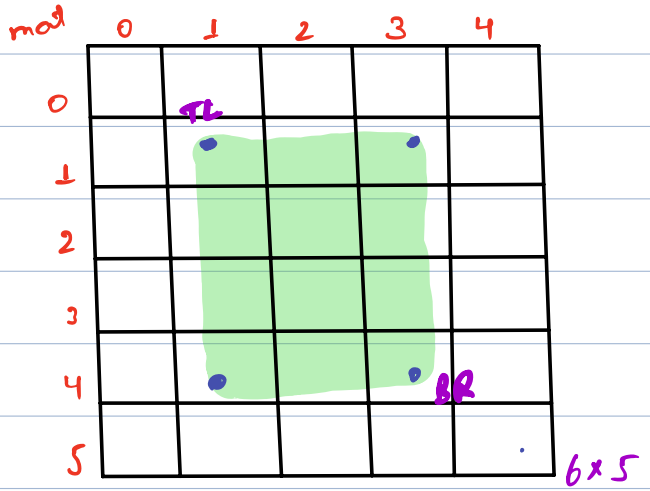
	0	1	2	3
0	10	50	90	130
1	20	60	100	140
2	30	70	110	150
3	40	80	120	160

Step2: reverse Column

	0	1	2	3
0	40	80	120	160
1	30	70	110	150
2	20	60	100	140
3	10	50	90	130

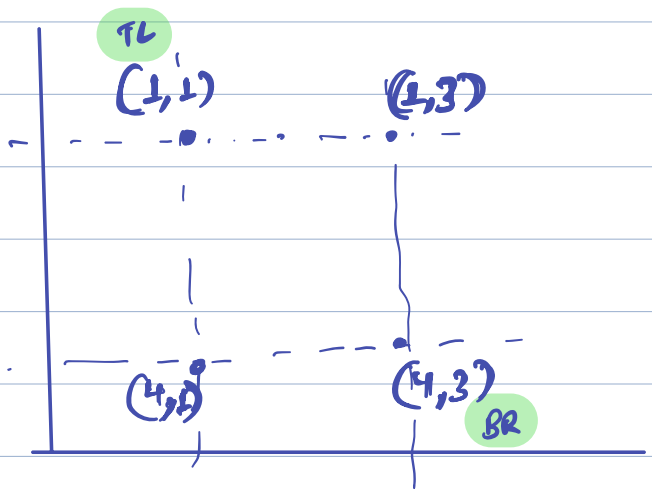
Representing submatrix:

→ mat[6][5]



TL (1,1) TR (1,3)

BL (4,1) BR (4,3)



{Leetcode 304}

Q) Range Sum Query - 2D

↳ Given a mat[N][m] and Q queries, for each query find submatrix sum.

Ex:

	0	1	2	3	4
0	7	1	-6	3	12
1	10	5	1	0	9
2	6	4	-3	8	11
3	13	-8	-5	12	4
4	3	2	1	9	8
5	4	3	-2	6	3

(TL)		(BR)		
x1	y1	x2	y2	
2	1	4	3	: 20
3	2	5	4	: 35

Idea:

↳ for every query iterate on the submatrix and get the sum.

T.C: O(N*m*Q) S.C: O(1)

Optimal approach

Psum[i] = Sum(0,i) | Sum(i,j) = Psum[j] - Psum[i-1]

↓

Psum[i][j] = sum{(0,0) - (i,j)}

(0,0) ↑ TL, (i,j) ↑ BR

arr

	0	1	2	3	4
0	7	1	-6	3	12
1	10	5	1	0	9
2	6	4	-3	8	11
3	13	-8	-5	12	4
4	3	2	1	9	8
5	4	3	-2	6	3

psum

	0	1	2	3	4
0	7			5	
1			18		
2					
3			35		
4					
5		50			

//2D Prefix Sum

arr

	0	1	2
0	a	b	c
1	d	e	f
2	g	h	i

① Apply row-wise Prefix Sum →

	0	1	2
0	a	a+b	a+b+c
1	d	d+e	d+e+f
2	g	g+h	g+h+i

Apply Col-wise Prefix Sum ↓

Psum

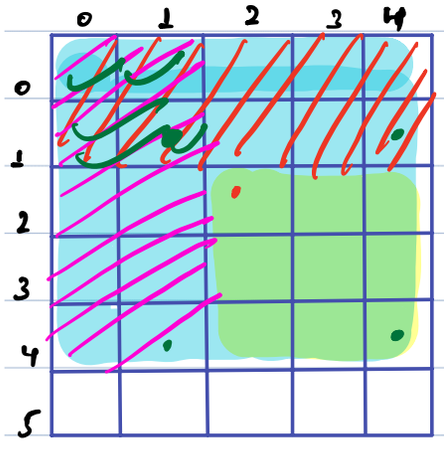
	0	1	2
0	a	a+b	a+b+c
1	a+d	a+b+d+e	a+b+c+d+e+f
2	a+d+g	a+b+d+e+g+h	a+b+c+d+e+f+g+h+i

=

	0	1	2
0	a	a+b	a+b+c
1	a+d	a+b+d+e	a+b+c+d+e+f
2	a+d+g	a+b+d+e+g+h	a+b+c+d+e+f+g+h+i

T.C: O(N*M)

//Solving the query



query

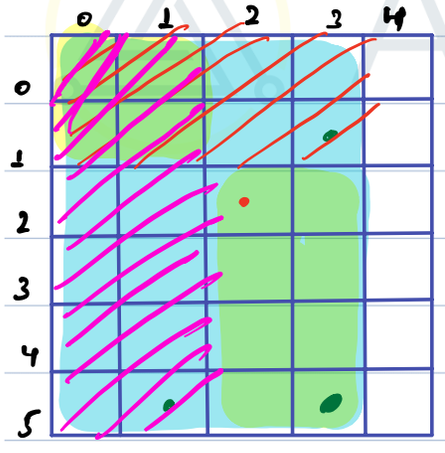
TL (2,2)

BR (4,4)

Psum[4][4] - Psum[1][4] - Psum[4][1]

+ Psum[1][1]

Sum[(0,0) - (4,4)]



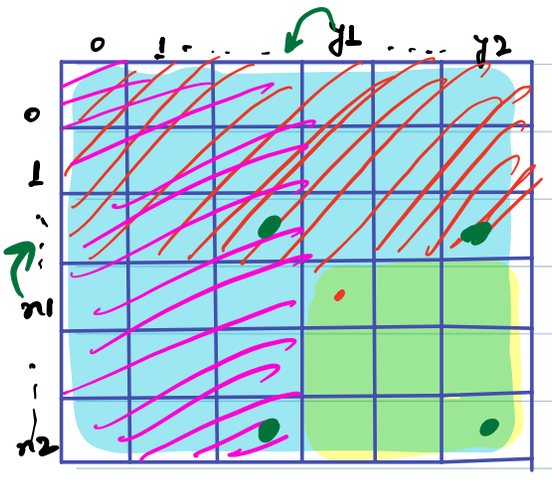
query

TL (2,2)

BR (5,3)

Psum[5][3] - Psum[1][3] - Psum[5][1]

+ Psum[1][1]



Query

(x_1, y_1) TL

(x_2, y_2) BR

$Psum[x_2][y_2] - Psum[x_1-1][y_2]$

$Psum[x_2][y_1-1] + Psum[x_1-1][y_1-1]$

Psum

i \ j	0	1	2
0	a	a+b	a+b+c
1	d	d+e	d+e+f
2	g	g+h	g+h+i

	0	1	2
0	a	b	c
1	d	e	f
2	g	h	i

a	a+b	a+b+c

//Pseudo Code

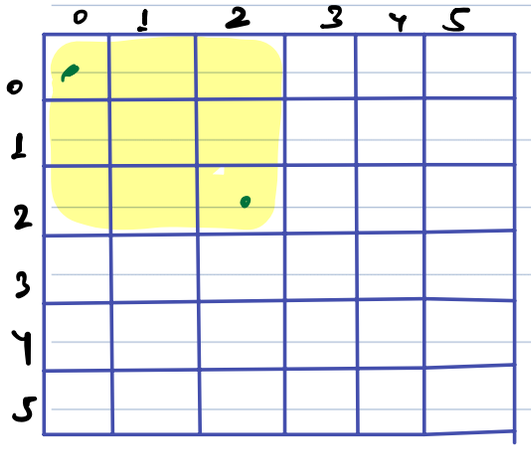
```
Class NumMatrix {
    int[][] Psum;                                   // creating the pref in sum
    Public NumMatrix (int[][] matrix) {
        Psum = new int[matrix.length][matrix[0].length];
        int n = matrix.length;
        int m = matrix[0].length;

        for(int i=0; i<n; i++) {
            for(int j=0; j<m; j++){
                if (j==0){ Psum[i][j] = matrix[i][j]
                else {
                Psum[i][j] = Psum[i][j-1] + mat[i][j];
                }
            }
        }

        for(int j=0; j<m; j++){
            for(int i=1; i<n; i++){
                Psum[i][j] = Psum[i-1][j] + Psum[i][j]
            }
        }
    }
}
```

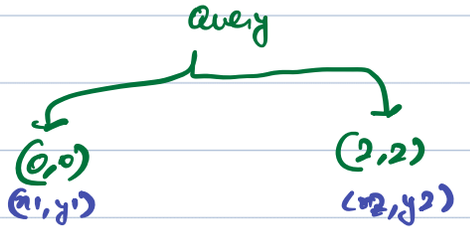
T.C: $O(N*M) + O(Q)$

$\Downarrow$

$O(N*M+Q)$

```
Public int sumRegion (int x1, int y1, int x2,
                      int y2 ){
    int sum = 0;
    sum = sum + Psum[x2][y2];
    if (x1-1 >= 0){
        sum = sum - Psum[x1-1][y2];
    }
    if (y1-1 >= 0){
        sum = sum - Psum[x2][y1-1];
    }
    if (x1-1 >= 0 && y1-1 >= 0){
        sum = sum + Psum[x1-1][y1-1];
    }
    return ans;
}
```





int ans = Psum[x2][y2] - Psum[x1-1][y2]
- Psum[x2][y1-1] + Psum[x1-1][y1-1]

Psum[2][2] - Psum[-1][2] - Psum[2][-1]
+ Psum[-1][-1]

```
for(int j=0; j<m; j++){
    for(int i=1; i<n; i++){
        psum[i][j] = psum[i-1][j] + psum[i][j-1]
    }
}
```

psum

	0	1	2
0	a	b	c
1	d	e	f
2	g	h	i

	0	1	2
0	a	a+b	a+b+c
1	a+d	a+b+d+e	a+b+c+d+e+f
2	a+d+g	a+b+d+e+g+h	a+b+c+d+e+f+g+h+i

Q) Given mat[n][m], find sum of all submatrix sum.

Ex: mat[6][5] → TL BR

	0	1	2	3	4
0	TL	TL	TL		
1	TL	TL	TL		
2	TL	TL	TL BR	BR	BR
3			BR	BR	BR
4			BR	BR	BR
5			BR	BR	BR

6*5

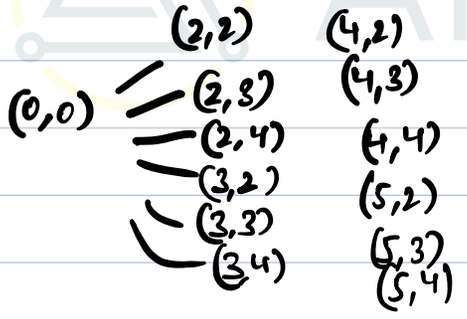
Idea → Contribution technique

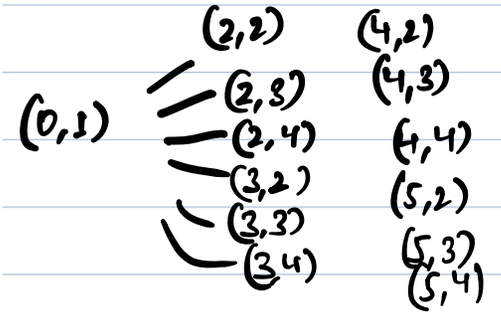
Σ (occ * mat[i][j])

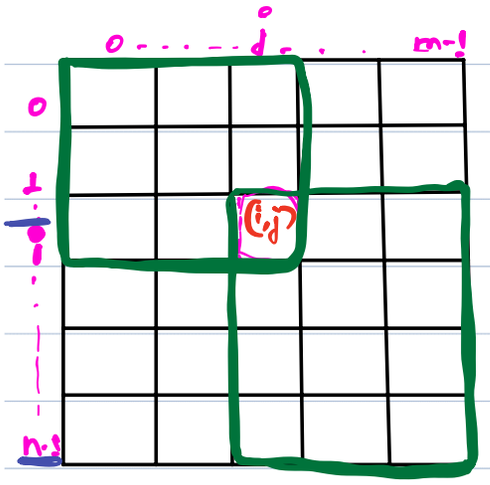
valid TL = 9

valid BR = 12

No. of submatrix in which (2,2) is present = 9x12 = 108







valid TL = (i+1) * (j+1)

valid BR = (n-i) * (m-j)

Total occ of (i,j) = (i+1) * (j+1) * (n-i) * (m-j)

// Pseudo code

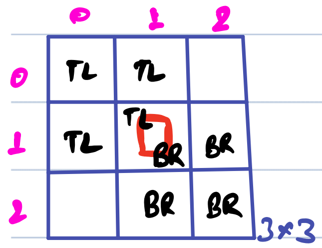
```
int SubmatrixSum(int arr[N][M]){
    int ans = 0;
    for(int i=0; i<N; i++){
        for(int j=0; j<M; j++){
            int occ = (i+1)*(j+1)*(n-i)*(m-j);
            ans = ans + (occ * arr[i][j]);
        }
    }
    return ans;
}
```

T.C: O(N*M)

S.C: O(1)

$2*2 * 2*2 = 16$

$(i+1)*(j+1)*(n-i)*(m-j);$



$(1,1)$ occ $= 4*4 = 16$

Contribution technique on 2D Arrays

Transpose of a matrix

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();

        int[][] arr = new int[n][n];
        for(int i=0;i<n;i++){
            for(int j=0;j<n;j++){
                arr[i][j] = scn.nextInt();
            }
        }

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

         for(int i=0;i<n;i++){
            for(int j=0;j<n;j++){
                System.out.print(arr[i][j]+" ");
            }

            System.out.println();
        }


    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;

int main() {
    int n;
    cin >> n;

    int arr[n][n];

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

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

    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            cout << arr[i][j] << " ";
        }
        cout << endl;
    }

    return 0;
}
```

Python Code:

```
n = int(input())
arr = []

for i in range(n):
    row = list(map(int, input().split()))
    arr.append(row)

for i in range(1, n):
    for j in range(i):
        arr[i][j], arr[j][i] = arr[j][i], arr[i][j]

for i in range(n):
    for j in range(n):
        print(arr[i][j], end=" ")
    print()
```

Rotate Image

Java Code:

Java

```java
class Solution {
    public void rotate(int[][] matrix) {
           int n = matrix.length;
               int m = matrix[0].length;

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

```
            }


            for (int i = 0; i < n; i++) {
                  int[] temp = matrix[i];

              int sp = 0;
              int ep = temp.length - 1;

              while(sp < ep){
                  int t = temp[sp];
                  temp[sp] = temp[ep];
                  temp[ep] = t;
                  sp++;
                  ep--;
              }

              matrix[i] = temp;
            }
        }
    }
```

C++ Code:

```
class Solution {
public:
    void rotate(vector<vector<int>>& matrix) {
        int n = matrix.size();
        int m = matrix[0].size();

        // Transpose the matrix
        for (int i = 0; i < n; i++) {
            for (int j = i; j < m; j++) {
                swap(matrix[i][j], matrix[j][i]);
            }
        }

        // Reverse each row
        for (int i = 0; i < n; i++) {
            int sp = 0;
```

```
            int ep = m - 1;

            while (sp < ep) {
                swap(matrix[i][sp], matrix[i][ep]);
                sp++;
                ep--;
            }
        }
    }
};
```

Python Code:

```
class Solution:
    def rotate(self, matrix):
        n = len(matrix)
        m = len(matrix[0])

        # Transpose the matrix
        for i in range(n):
            for j in range(i, m):
                matrix[i][j], matrix[j][i] = matrix[j][i], matrix[i][j]

        # Reverse each row
        for i in range(n):
            sp = 0
            ep = m - 1

            while sp < ep:
                matrix[i][sp], matrix[i][ep] = matrix[i][ep], matrix[i][sp]
                sp += 1
                ep -= 1
```

Range Sum Query 2D

Java Code:

```java
class NumMatrix {
    int[][] psum;
    public NumMatrix(int[][] mat) {
        int n = mat.length;
        int m = mat[0].length;

        psum = new int[n][m];

        for(int i=0;i<n;i++){
            for(int j=0;j<m;j++){
                if(j == 0){
                    psum[i][j] = mat[i][j];
                }else{
                    psum[i][j] = psum[i][j-1] + mat[i][j];
                }
            }
        }

        for(int j=0;j<m;j++){
            for(int i=1;i<n;i++){
                psum[i][j] = psum[i-1][j] + psum[i][j];
            }
        }
    }

    public int sumRegion(int row1, int col1, int row2, int col2)
{
        int ans = 0;

        ans = ans+ psum[row2][col2];
        if(row1 > 0){
            ans -= psum[row1-1][col2];
        }

        if(col1 > 0){
            ans -= psum[row2][col1-1];
```

```
            }

            if(row1 > 0 && col1 > 0){
                ans += psum[row1-1][col1-1];
            }

            return ans;
        }
    }
```

C++ Code:

```
class NumMatrix {
    vector<vector<int>> psum;

public:
    NumMatrix(vector<vector<int>>& mat) {
        int n = mat.size();
        int m = mat[0].size();

        psum = vector<vector<int>>(n, vector<int>(m, 0));

        for (int i = 0; i < n; i++) {
            for (int j = 0; j < m; j++) {
                if (j == 0) {
                    psum[i][j] = mat[i][j];
                } else {
                    psum[i][j] = psum[i][j - 1] + mat[i][j];
                }
            }
        }

        for (int j = 0; j < m; j++) {
            for (int i = 1; i < n; i++) {
                psum[i][j] = psum[i - 1][j] + psum[i][j];
            }
        }
    }

    int sumRegion(int row1, int col1, int row2, int col2) {
```

```
        int ans = psum[row2][col2];

        if (row1 > 0) {
            ans -= psum[row1 - 1][col2];
        }

        if (col1 > 0) {
            ans -= psum[row2][col1 - 1];
        }

        if (row1 > 0 && col1 > 0) {
            ans += psum[row1 - 1][col1 - 1];
        }

        return ans;
    }
};
```

Python Code:

```
class NumMatrix:

    def __init__(self, matrix):
        n = len(matrix)
        m = len(matrix[0])

        self.psum = [[0] * m for _ in range(n)]

        for i in range(n):
            for j in range(m):
                if j == 0:
                    self.psum[i][j] = matrix[i][j]
                else:
                    self.psum[i][j] = self.psum[i][j - 1] + matrix[i][j]

        for j in range(m):
            for i in range(1, n):
                self.psum[i][j] = self.psum[i - 1][j] + self.psum[i][j]

    def sumRegion(self, row1, col1, row2, col2):
        ans = self.psum[row2][col2]

        if row1 > 0:
            ans -= self.psum[row1 - 1][col2]
```

```
        if col1 > 0:
            ans -= self.psum[row2][col1 - 1]

        if row1 > 0 and col1 > 0:
            ans += self.psum[row1 - 1][col1 - 1]

        return ans
```

Sum of all Submatrix

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int m = scn.nextInt();

        int[][] arr = new int[n][m];

        for(int i=0;i<n;i++){
            for(int j=0;j<m;j++){
                arr[i][j] = scn.nextInt();
            }
        }

        int ans = 0;

        for(int i=0;i<n;i++){
            for(int j=0;j<m;j++){
                int occ = (i+1)*(j+1)*(n-i)*(m-j);

                ans = ans + (occ * arr[i][j]);
            }
        }
```

```
        System.out.println(ans);

    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;

int main() {
    int n, m;
    cin >> n >> m;

    int arr[n][m];

    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            cin >> arr[i][j];
        }
    }

    int ans = 0;

    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            int occ = (i + 1) * (j + 1) * (n - i) * (m - j);

            ans = ans + (occ * arr[i][j]);
        }
    }

    cout << ans << endl;

    return 0;
}
```

Python Code:

```
n, m = map(int, input().split())

arr = []
```

```
for i in range(n):
    row = list(map(int, input().split()))
    arr.append(row)

ans = 0

for i in range(n):
    for j in range(m):
        occ = (i + 1) * (j + 1) * (n - i) * (m - j)
        ans += occ * arr[i][j]

print(ans)
```

Max Submatrix Sum of Sorted Matrix

Solution Vid:

https://youtu.be/gZLfQSnhXLE

Java Code:

```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        /* Enter your code here. Read input from STDIN. Print output to STDOUT. Your class
should be named Solution. */
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int m = scn.nextInt();

        int[][] arr = new int[n][m];

        for(int i=0;i<n;i++){
            for(int j =0;j<m;j++){
                arr[i][j] = scn.nextInt();
            }
        }

        int[][] pf = prefix(arr);
```

```
    int ans = Integer.MIN_VALUE;
    for(int i=0;i<n;i++){
        for(int j=0;j<m;j++){
            int temp = sumRegion(i,j,n-1,m-1,pf);
            ans = Math.max(ans, temp);
        }
    }

    System.out.println(ans);
}

public static int sumRegion(int row1, int col1, int row2, int col2, int[][]psum){
    int ans = 0;

    ans = ans+ psum[row2][col2];
    if(row1 > 0){
        ans -= psum[row1-1][col2];
    }

    if(col1 > 0){
        ans -= psum[row2][col1-1];
    }

    if(row1 > 0 && col1 > 0){
        ans += psum[row1-1][col1-1];
    }

    return ans;
}

public static int[][] prefix(int[][] arr){
    int n = arr.length;
    int m = arr[0].length;

    int[][] psum = new int[arr.length][arr[0].length];

    for(int i=0;i<n;i++){
        for(int j=0;j<m;j++){
            if(j == 0){
                psum[i][j] = arr[i][j];
            }else{
                psum[i][j] = psum[i][j-1] + arr[i][j];
            }
        }
    }

    for(int j=0;j<m;j++){
```

```
            for(int i=1;i<n;i++){
                psum[i][j] = psum[i-1][j] + psum[i][j];
            }
        }

        return psum;
    }
}
```

C++ Code:

```
#include <iostream>
#include <vector>

using namespace std;

int sumRegion(int row1, int col1, int row2, int col2, vector<vector<int>>& psum) {
    int ans = 0;

    ans = ans + psum[row2][col2];
    if (row1 > 0) {
        ans -= psum[row1 - 1][col2];
    }

    if (col1 > 0) {
        ans -= psum[row2][col1 - 1];
    }

    if (row1 > 0 && col1 > 0) {
        ans += psum[row1 - 1][col1 - 1];
    }

    return ans;
}

vector<vector<int>> prefix(vector<vector<int>>& arr) {
    int n = arr.size();
    int m = arr[0].size();

    vector<vector<int>> psum(n, vector<int>(m, 0));

    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            if (j == 0) {
                psum[i][j] = arr[i][j];
            } else {
```

```
                psum[i][j] = psum[i][j - 1] + arr[i][j];
            }
        }
    }

    for (int j = 0; j < m; j++) {
        for (int i = 1; i < n; i++) {
            psum[i][j] = psum[i - 1][j] + psum[i][j];
        }
    }

    return psum;
}

int main() {
    int n, m;
    cin >> n >> m;

    vector<vector<int>> arr(n, vector<int>(m, 0));

    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            cin >> arr[i][j];
        }
    }

    vector<vector<int>> psum = prefix(arr);

    int ans = INT_MIN;
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            int temp = sumRegion(i, j, n - 1, m - 1, psum);
            ans = max(ans, temp);
        }
    }

    cout << ans << endl;

    return 0;
}
```

Python Code:

```
def sumRegion(row1, col1, row2, col2, psum):
    ans = psum[row2][col2]
```

```
    if row1 > 0:
        ans -= psum[row1 - 1][col2]

    if col1 > 0:
        ans -= psum[row2][col1 - 1]

    if row1 > 0 and col1 > 0:
        ans += psum[row1 - 1][col1 - 1]

    return ans

def prefix(arr):
    n = len(arr)
    m = len(arr[0])

    psum = [[0] * m for _ in range(n)]

    for i in range(n):
        for j in range(m):
            if j == 0:
                psum[i][j] = arr[i][j]
            else:
                psum[i][j] = psum[i][j - 1] + arr[i][j]

    for j in range(m):
        for i in range(1, n):
            psum[i][j] = psum[i - 1][j] + psum[i][j]

    return psum

n, m = map(int, input().split())
arr = []

for i in range(n):
    row = list(map(int, input().split()))
    arr.append(row)

psum = prefix(arr)

ans = float('-inf')
for i in range(n):
    for j in range(m):
        temp = sumRegion(i, j, n - 1, m - 1, psum)
        ans = max(ans, temp)

print(ans)
```

Today's agenda

- Max Chunks to make Sorted 1
- Max Chunks to make Sorted 2
- Print boundary
- Print Spiral Square matrix
- Print Spiral rectangle matrix

Q) max Chunks to make Sorted I

↳ Given an arr[N] containing permutation of the integers in the range [0, n-1]. you have to split array into maximum possible number of chunks such that after individually sorting each chunk, the array gets sorted.

Ex: arr[5]: {1 0 2 3 4}

↳ ans = 4

Ex: arr[9]: {2 0 1 4 3 6 7 5 8}

↳ ans: 4

Ex: arr[5]: {4 3 2 1 0}

↳ ans: 1

//idea

arr[i] = {0, n-1}

Ex: arr[9]: {2 0 1 | 4 3 | 6 7 5 | 8}

obs: if max(0...i) is equal to i → valid chunk.

//Pseudo code

```
Public int maxChunksSortedone (int arr[N]){

    int maxtillnow = -∞;
    int Count = 0;

    for(int i=0; i<N; i++){
        maxtillnow = mathmax(maxtillnow, arr[i]);

        if(maxtillnow == i){ Count++;}
    }

    return Count;
}
```

T.C: O(N)

S.C: O(1)

Tracing

arr[9]: { 2 0 1 | 4 3 | 6 7 5 | 8 }

(indices: 0 1 2 | 3 4 | 5 6 7 | 8)

```
int maxtillnow = -∞;
int Count = 0;

for(int i=0; i<N; i++){
    maxtillnow = mathmax(maxtillnow, arr[i]);

    if(maxtillnow == i){ Count++;}
}

return Count;
```

i	maxtillnow	Count
0	2	0
1	2	0
2	2	1
3	4	1
4	4	2
5	6	2
6	7	2
7	7	3
8	8	4

Q) max Chunks to make Sorted 2

Ls Given an arr[N] containing permutation of the integers. you have to split array into maximum possible number of chunks such that after individually sorting each Chunk, the array gets sorted.

arr[9] =	0	1	2	3	4	5	6	7	8
23	10	18	35	27	48	60	40	55

Ls ans = 3

arr[6] =	0	1	2	3	4	5
13	7	12	22	18	26

Ls ans = 3

//idea

```
if (max(0...i) <= min(i+1....n-1)){
    //valid chunk
}
```

	0	1	2	3	4	5	6	7	8 (i)
arr[9] =	23	10	18	35	27	48	60	40	55
Pref[inmax[9]]:	23	23	23	35	35	48	60	60	60
Suff[inmin[9]]:	10	10	18	27	27	40	40	40	55

+1

Count = 0 + 1 + 1 + 1

//Pseudo Code

```
int maxChunksSorted2 (int arr[N]){

    int [] PrefixMax = PrefixMax(arr);  → H.W
    int [] SuffixMin = SuffixMin(arr);  → H.W

T.C: O(N)       int count=0;
S.C: O(N)
    for(int i=0; i<N-1; i++){
        if(PrefixMax[i]<=SuffixMin[i+1]){
            count++;
        }
    }

    count++;
    return count;
}
```

Q) Print boundary

↳ Given `mat[N][N]`, Print boundary in Clockwise direction.

	0	1	2
0	1	2	3
1	4	5	6
2	7	8	9

↳ 1 2 3 6 9 8 7 4

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

1 2 3 4 5 10 15 20 25 24 23 22 21 16 11 6

//idea

	0	1	2	3	4
0	1	2	3	4	5
1	6	7	8	9	10
2	11	12	13	14	15
3	16	17	18	19	20
4	21	22	23	24	25

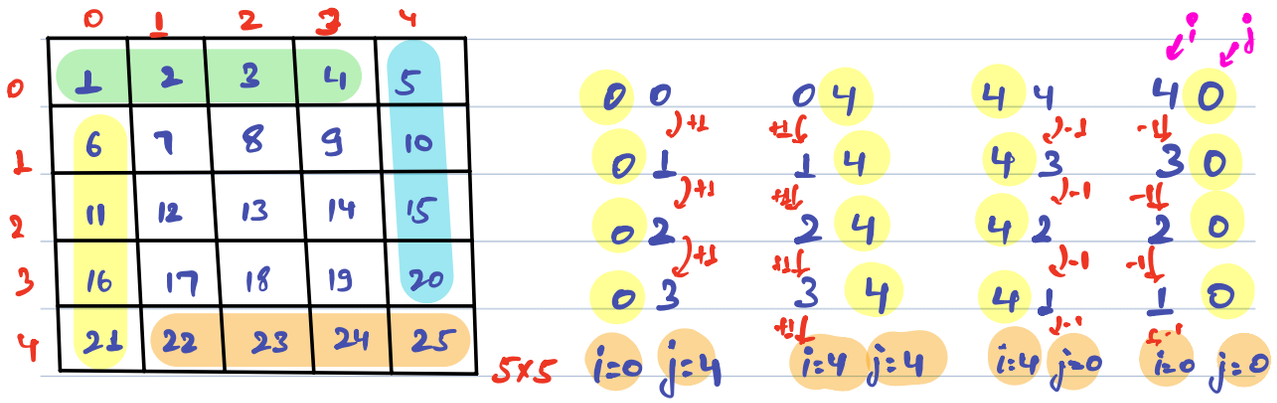
5x5

↳ 4 no. of 0th row
↳ 4 no. of last Col
↳ 4 no. of last row in reverse
↳ 4 no. of 0th Col in reverse

	0	1	2	3	4	5
0	1	2	3	4	5	6
1	7	8	9	10	11	12
2	13	14	15	16	17	18
3	19	20	21	22	23	24
4	25	26	27	28	29	30
5	31	32	33	34	35	36

6x6

↳ N-1 no. of 0th row
↳ N-1 no. of last Col
↳ N-1 no. of last row in reverse
↳ N-1 no. of 0th Col in reverse


0 1 2 3 4
0 1 2 3 4 5
1 6 7 8 9 10
2 11 12 13 14 15
3 16 17 18 19 20
4 21 22 23 24 25
5X5
0 0 0 4 4 4 4 0
0 1 1 4 4 3 3 0
0 2 2 4 4 2 2 0
0 3 3 4 4 1 1 0
i:0 j:4
i:4 j:4
i:4 j:0
i:0 j:0
i
j
+1
-1

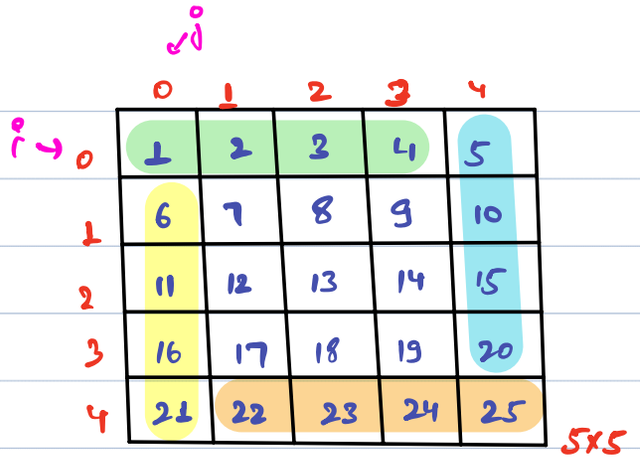
// Pseudo Code

```
void PrintBoundary (int mat[N][N]){
    int i=0;
    int j=0;

    for( k=0; k<N-1; k++){
        Print(mat[i][j]);
        j++;
    }

    for( k=0; k<N-1; k++){
        Print(mat[i][j]);
        i++;
    }

    for( k=0; k<N-1; k++){
        Print(mat[i][j]);
        j--;
    }

    for( k=0; k<N-1; k++){
        Print(mat[i][j]);
        i--;
    }
}
```

T.C: O(N)

S.C: O(1)

```
void PrintBoundary (int mat[N][N]){
    int i=0;
    int j=0;

    for( K=0; K<N-1; K++){
        Print(mat[i][j]);
        j++;
    }

    for( K=0; K<N-1; K++){
        Print(mat[i][j]);
        i++;
    }

    for( K=0; K<N-1; K++){
        Print(mat[i][j]);
        j--;
    }

    for( K=0; K<N-1; K++){
        Print(mat[i][j]);
        i--;
    }
}
```



1 2 3 4 5 10 15 20
25 24 23 22 22 16 11 6

Q) Square matrix Spiral

↳ Given mat[N][N], Print the Spiral form.

	0	1	2	3	4	5
0	1	2	3	4	5	6
1	7	8	9	10	11	12
2	13	14	15	16	17	18
3	19	20	21	22	23	24
4	25	26	27	28	29	30
5	31	32	33	34	35	36

6×6

i	j	steps
0	0	N-1=5
+1 ↓	↓ +1	↓ -2
1	1	3
+1 ↓	↓ +1	↓ -2
2	2	1

↳ do PrintBoundary till your steps >=1.

	0	1	2	3	4
0	1	2	3	4	5
1	6	7	8	9	10
2	11	12	13	14	15
3	16	17	18	19	20
4	21	22	23	24	25

5×5

i	j	steps
0	0	4
+1 ↓	↓ +1	↓ -2
1	1	2
2	2	0

//Pseudo code

```
void PrintSpiralSquare(int mat[N][N]){
    int i=0;
    int j=0;
    int steps = N-1;

    while(steps >= 1){
        for(k=0; k<steps; k++){
            Print(mat[i][j]);
            j++;
        }
        for(k=0; k<steps; k++){
            Print(mat[i][j]);
            i++;
        }
        for(k=0; k<steps; k++){
            Print(mat[i][j]);
            j--;
        }
        for(k=0; k<steps; k++){
            Print(mat[i][j]);
            i--;
        }
        i++;
        j++;
        steps = steps-2;
    }

    if(steps == 0){ Print mat[i][j]; }
}
```

T.C: $O(N^2)$

S.C: $O(1)$

```
void PrintSpiralSquare (int mat[N][N]){
    int i=0;
    int j=0;
    int steps = N-1;

    while (steps >= 1) {
        for (k=0; k<steps; k++){
            Print (mat[i][j]);
            j++;
        }
        for (k=0; k<steps; k++){
            Print (mat[i][j]);
            i++;
        }
        for (k=0; k<steps; k++){
            Print (mat[i][j]);
            j--;
        }
        for (k=0; k<steps; k++){
            Print (mat[i][j]);
            i--;
        }
        i++;
        j++;
        steps = steps-2;
    }
    if (steps == 0) { Print mat[i][j]; }
}
```

	0	1	2	3	4	5
0	1	2	3	4	5	6
1	7	8	9	10	11	12
2	13	14	15	16	17	18
3	19	20	21	22	23	24
4	25	26	27	28	29	30
5	31	32	33	34	35	36

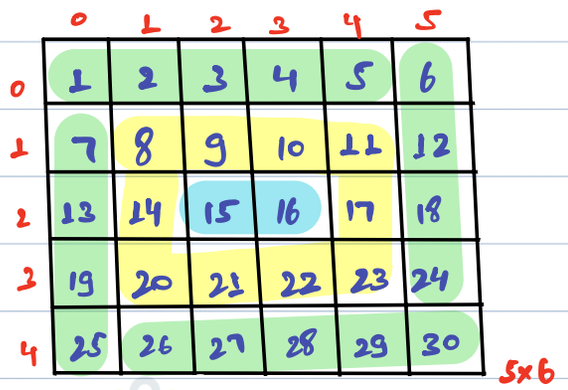
6x6

i	j	steps
0	0	5
1	1	3
2	2	1
3	3	-1

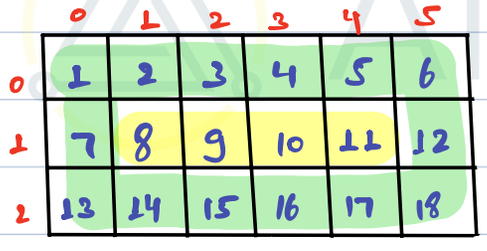
(, exit

Q) Spiral matrix

↳ Given `mat[n][m]`, return all elements of the matrix in spiral order.



i	j	oSteps	cSteps
0	0	4	0



i	j	oSteps	cSteps
0	0	4	2
1	1	2	0

```
void PrintSpiralSquare(int mat[n][m]){
    int i = 0;
    int j = 0;
    int rSteps = n-1;
    int cSteps = m-1;

    while(rSteps >= 1 && cSteps >= 1){
        for(k=0; k<cSteps; k++){
            Print(mat[i][j]);
            j++;
        }

        for(k=0; k<rSteps; k++){
            Print(mat[i][j]);
            i++;
        }

        for(k=0; k<cSteps; k++){
            Print(mat[i][j]);
            j--;
        }

        for(k=0; k<rSteps; k++){
            Print(mat[i][j]);
            i--;
        }
        i++;
        rSteps = rSteps-2;
        j++;
        cSteps = cSteps-2;
    }

    if(rSteps == 0){
        for(int k=0; k<cSteps+1; k++){
            Print(mat[i][j]);
            j++;
        }
    }
}
```

T.C: O(n*m)

S.C: O(1)

```
        else if (csteps==0){
            for(int k=0; k<csteps+1; k++){
                Print(mat[i][j]);
                i++;
            }
        }
    }
```

Q) Flip bits

↳ Given an arr[N] that contains 1 and 0 only. Find the subarray when flipped can give max 1's in the entire array. (flipping once)

ex: arr[5]:

	0	1	2	3	4	
	1	0	1	0	0	
1-4	1	1	0	1	1	: 4
0-2	0	1	0	3	4	: 1
2-4	1	0	0	1	1	: 3

arr[7]:

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

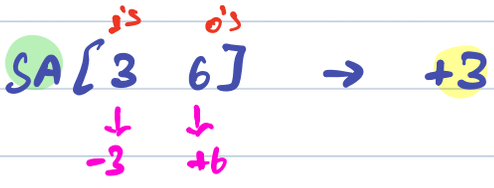
//idea 1

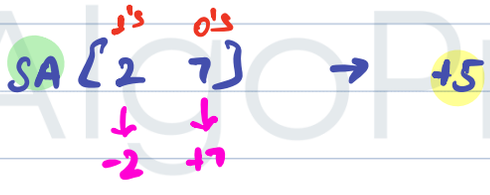
↳ Try flipping every subarray and choose the one which gives us the maximum.

T.C: $O(N^2) * O(N) \approx O(N^3)$

//idea2

Obs1:





→ flip subarray with max no. of net gain

Net gain:

↳ 1 → 0 : -1

↳ 0 → 1 : +1

	0	1	2	3	4	5	6
arr[i]:	0	0	1	0	0	1	0

⇓

arr[i]:	+1	+1	-1	+1	+1	-1	+1

↓

find the subarray with max gain

↓

max subarray sum

↳ Kadane's algo

arr[i]:	+1	+1	-1	+1	+1	-1	+1

cmax : 0 1 2 1 2 3 2 3

omax : 0 1 2 3

ans = 5

//Pseudo code

```
int maxones (int arr[N]){
    int count=0;
    for (int i=0; i<N; i++) {
        if (arr[i]==0) {
            arr[i]=1;
        }
        else {
            arr[i]=-1;
            count++;
        }
    }
    int csum=0;
    int osum=-∞;

    for (int i=0; i<N; i++) {
        if (csum>=0) { csum+=arr[i]; }
            else { csum=arr[i]; }

        osum = Math.max(osum, csum);
    }
    if (osum>0){ return osum+count; }
        else { return count; }
}
```

T.C: O(N)

S.C: O(1)

↳ you didn't come this far only to come this far.

First Missing Positive

Java Code:

Java

```java
class Solution {
    public int firstMissingPositive(int[] nums) {
        int n = nums.length;

        int i =0;
        while(i<n){
            if(nums[i] == i+1){
                i++;
                continue;
            }

            if(nums[i] <= 0 || nums[i] > n){
                i++;
                continue;
            }

            int idx1 = i;
            int idx2 = nums[i] - 1;

            if(nums[idx1] == nums[idx2]){
                i++;
                continue;
            }
            int temp = nums[idx1];
            nums[idx1] = nums[idx2];
            nums[idx2] = temp;

        }

        for(int j=0;j<n;j++){
```

```
                if(nums[j] != j+1){
                    return j+1;
                }
            }

            return n+1;
        }
    }
```

C++ Code:

```
class Solution {
public:
    int firstMissingPositive(vector<int>& nums) {
        int n = nums.size();

        int i = 0;
        while (i < n) {
            if (nums[i] == i + 1) {
                i++;
                continue;
            }

            if (nums[i] <= 0 || nums[i] > n) {
                i++;
                continue;
            }

            int idx1 = i;
            int idx2 = nums[i] - 1;

            if (nums[idx1] == nums[idx2]) {
                i++;
                continue;
            }
            int temp = nums[idx1];
            nums[idx1] = nums[idx2];
            nums[idx2] = temp;
        }

        for (int j = 0; j < n; j++) {
```

```
            if (nums[j] != j + 1) {
                return j + 1;
            }
        }

        return n + 1;
    }
};
```

Python Code:

```
class Solution:
    def firstMissingPositive(self, nums: List[int]) -> int:
        n = len(nums)

        i = 0
        while i < n:
            if nums[i] == i + 1:
                i += 1
                continue

            if nums[i] <= 0 or nums[i] > n:
                i += 1
                continue

            idx1 = i
            idx2 = nums[i] - 1

            if nums[idx1] == nums[idx2]:
                i += 1
                continue

            nums[idx1], nums[idx2] = nums[idx2], nums[idx1]

        for j in range(n):
            if nums[j] != j + 1:
                return j + 1

        return n + 1
```

Majority Element 1

Java Code:

```java
class Solution {
    public int majorityElement(int[] nums) {
        int val = nums[0];
        int count = 1;

        for(int i=1;i<nums.length;i++){
            if(count == 0){
                val = nums[i];
                count = 1;
            }else if(nums[i] == val){
                count++;
            }else{
                count--;
            }
        }

        return val;
    }
}
```

C++ Code:

```
class Solution {
public:
    int majorityElement(vector<int>& nums) {
        int val = nums[0];
        int count = 1;

        for (int i = 1; i < nums.size(); i++) {
            if (count == 0) {
                val = nums[i];
```

```
            count = 1;
        } else if (nums[i] == val) {
            count++;
        } else {
            count--;
        }
    }

    return val;
  }
};
```

Python Code:

```
class Solution:
    def majorityElement(self, nums: List[int]) -> int:
        val = nums[0]
        count = 1

        for i in range(1, len(nums)):
            if count == 0:
                val = nums[i]
                count = 1
            elif nums[i] == val:
                count += 1
            else:
                count -= 1

        return val
```

Majority Element 2

Java Code:

Java

```
class Solution {
```

```java
public List<Integer> majorityElement(int[] nums) {
    int val1 = nums[0];
    int count1 = 1;
    int val2 = nums[0];
    int count2 = 0;

    for(int i=1;i<nums.length;i++){
        if(nums[i] == val1){
            count1++;
        }else if(nums[i] == val2){
            count2++;
        }else if(count1 == 0){
            val1 = nums[i];
            count1 = 1;
        }else if(count2 == 0){
            val2 = nums[i];
            count2 = 1;
        }else{
            count1--;
            count2--;
        }
    }

    int c1 = 0;
    int c2 = 0;

    for(int val: nums){
        if(val == val1){
            c1++;
        }else if(val == val2){
            c2++;
        }
    }
    List<Integer> ans = new ArrayList<>();
    if(c1 > nums.length/3){
        ans.add(val1);
    }
```

```
            if(c2 > nums.length /3){
                ans.add(val2);
            }

            return ans;
        }
    }
```

C++ Code:

```
class Solution {
public:
    vector<int> majorityElement(vector<int>& nums) {
        int val1 = nums[0];
        int count1 = 1;
        int val2 = nums[0];
        int count2 = 0;

        for (int i = 1; i < nums.size(); i++) {
            if (nums[i] == val1) {
                count1++;
            } else if (nums[i] == val2) {
                count2++;
            } else if (count1 == 0) {
                val1 = nums[i];
                count1 = 1;
            } else if (count2 == 0) {
                val2 = nums[i];
                count2 = 1;
            } else {
                count1--;
                count2--;
            }
        }

        int c1 = 0;
        int c2 = 0;

        for (int val : nums) {
            if (val == val1) {
                c1++;
            } else if (val == val2) {
```

```
            c2++;
        }
    }

    vector<int> ans;
    if (c1 > nums.size() / 3) {
        ans.push_back(val1);
    }

    if (c2 > nums.size() / 3) {
        ans.push_back(val2);
    }

    return ans;
  }
};
```

Python Code:

```
class Solution:
    def majorityElement(self, nums):
        val1, count1 = nums[0], 1
        val2, count2 = nums[0], 0

        for i in range(1, len(nums)):
            if nums[i] == val1:
                count1 += 1
            elif nums[i] == val2:
                count2 += 1
            elif count1 == 0:
                val1, count1 = nums[i], 1
            elif count2 == 0:
                val2, count2 = nums[i], 1
            else:
                count1 -= 1
                count2 -= 1

        c1, c2 = 0, 0

        for val in nums:
            if val == val1:
                c1 += 1
            elif val == val2:
                c2 += 1
```

```
ans = []
if c1 > len(nums) // 3:
    ans.append(val1)

if c2 > len(nums) // 3:
    ans.append(val2)

return ans
```

Next Greater Element 3

Java Code:

Java

```java
class Solution {
    public int nextGreaterElement(int n) {
        char[] number = (n+"").toCharArray();

        int idx = -1;
        for(int i = number.length - 2; i>=0;i--){
            if(number[i] < number[i+1]){
                idx = i;
                break;
            }
        }

        if(idx == -1){
            return -1;
        }

        int val = number[idx];
        int smallest = idx+1;
```

```
            for(int j = idx+1;j<number.length;j++){
                if(number[j] > val && number[j] <= number[smallest]){
                    smallest = j;
                }
            }

            char temp = number[idx];
            number[idx] = number[smallest];
            number[smallest] = temp;

            Arrays.sort(number,idx+1,number.length);

            long ans = Long.parseLong(new String(number));

            if(ans > Integer.MAX_VALUE){
                return -1;
            }else{
                return (int)ans;
            }

        }


    }
```

C++ Code:

```
class Solution {
public:
    int nextGreaterElement(int n) {
        string strNum = to_string(n);

        int idx = -1;
        for (int i = strNum.length() - 2; i >= 0; i--) {
            if (strNum[i] < strNum[i + 1]) {
                idx = i;
                break;
            }
        }
```

```
        if (idx == -1) {
            return -1;
        }

        char val = strNum[idx];
        int smallest = idx + 1;

        for (int j = idx + 1; j < strNum.length(); j++) {
            if (strNum[j] > val && strNum[j] <= strNum[smallest]) {
                smallest = j;
            }
        }

        swap(strNum[idx], strNum[smallest]);

        sort(strNum.begin() + idx + 1, strNum.end());

        long long ans = stoll(strNum);

        if (ans > INT_MAX) {
            return -1;
        } else {
            return (int)ans;
        }
    }
};
```

Python Code:

```
class Solution:
    def nextGreaterElement(self, n: int) -> int:
        # Convert the integer to a string
        str_num = str(n)

        # Find the index of the first digit from the right that is smaller than the digit to its right
        idx = -1
        for i in range(len(str_num) - 2, -1, -1):
            if str_num[i] < str_num[i + 1]:
                idx = i
                break

        if idx == -1:
            return -1
```

```
# Find the smallest digit to the right of idx that is greater than str_num[idx]
val = str_num[idx]
smallest = idx + 1
for j in range(idx + 1, len(str_num)):
    if str_num[j] > val and str_num[j] <= str_num[smallest]:
        smallest = j

# Swap the digits at idx and smallest
str_num_list = list(str_num)
str_num_list[idx], str_num_list[smallest] = str_num_list[smallest], str_num_list[idx]

# Sort the digits to the right of idx in ascending order
str_num_list[idx + 1:] = sorted(str_num_list[idx + 1:])

# Convert the modified string back to an integer
ans = int(''.join(str_num_list))

# Check for overflow and return the result
return ans if ans <= 2**31 - 1 else -1
```

Maximize Distance to Closest Person_HW

Solution vid:

https://youtu.be/vuGRIull9z0

Java Code:

Java

```java
class Solution {
    public int maxDistToClosest(int[] seats) {
           int dist = Integer.MIN_VALUE;
           int i = -1;

           for (int j = 0; j < seats.length; j++) {
                 if (seats[j] == 1) {
```

```
                    if (i == -1) {
                        dist = Math.max(dist, j);
                    } else {
                        dist = Math.max(dist, (j - i) / 2);
                    }
                    i = j;
                }
            }

            dist = Math.max(dist, seats.length - 1 - i);

            return dist;
        }
    }
```

C++ Code:

```
class Solution {
public:
    int maxDistToClosest(vector<int>& seats) {
        int dist = INT_MIN;
        int j = -1;

        if (seats[0] == 1) {
            j = 0;
        }

        for (int i = 1; i < seats.size(); i++) {
            if (seats[i] == 1) {
                if (j == -1) {
                    dist = max(dist, i);
                } else {
                    dist = max(dist, (i - j) / 2);
                }
                j = i;
            }
        }

        if (seats[seats.size() - 1] == 0) {
            dist = max(dist, static_cast<int>(seats.size()) - 1 - j);
```

```
        }

        return dist;
    }
};
```

Python Code:

```
class Solution:
    def maxDistToClosest(self, seats: List[int]) -> int:
        dist = float('-inf')
        j = -1

        if seats[0] == 1:
            j = 0

        for i in range(1, len(seats)):
            if seats[i] == 1:
                if j == -1:
                    dist = max(dist, i)
                else:
                    dist = max(dist, (i - j) // 2)
                j = i

        if seats[-1] == 0:
            dist = max(dist, len(seats) - 1 - j)

        return dist
```

Today's agenda

- first missing Positive
- Majority element
- majority element - 2
- Next greater element 3

leetcode 41: First missing Positive → {Amazon, microsoft, APPle}

↳ Given arr[N], find first missing natural number.

{1, 2, 3, 4, 5 ...}

Ex: arr[5]: 3 -2 1 2 7 → 4

arr[7]: -8 2 6 4 -7 1 3 → 5

arr[6]: 2 1 6 4 3 5 → 7

arr[5]: -4 8 3 -1 0 → 1

arr[4]: 2 1 4 3 → 5

arr[3]: 1 2 1 → 3

arr[4]: -2 -3 -8 -7 → 1

//Ideal

↳ Sort the array

arr[5]: 3 -2 1 2 7

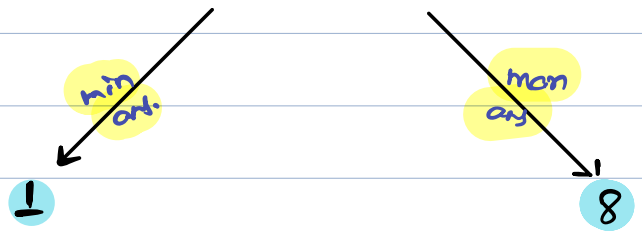
⇓

{-2 1 2 3 7} → 4 (i)

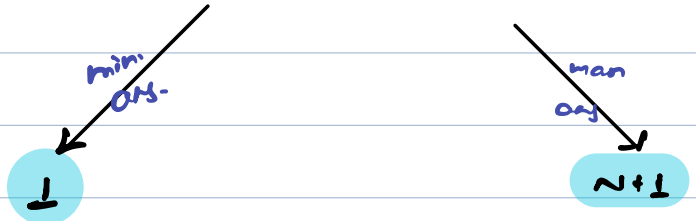
T.C: O(NlogN) + O(N) ≈ O(NlogN)

// idea2

obs1: arr[i]: 1 2 3 4 5 6 7

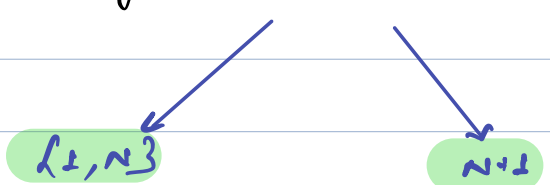


arr[N]: 1 2 3 _ _ _ _ N



→ ans range → {1, N+1}

obs2: Answer range → {1, N+1}



arr[N]: 1 2 ~~3~~ -3 _ _ _ _ N-1 N

Ex: arr[8]:

0	1	2	3	4	5	6	7
4	2	7	6	9	1	8	3

answer range: {1, 9}

myanswerange: {1, 8}

0	1	2	3	4	5	6	7
1	2	3	4	~~5~~	6	7	8

↳ How to do this mapping → Swapping

{1, 8}

↳ arr[8]:

0	1	2	3	4	5	6	7
~~4~~ ~~6~~ 1	2	~~-7~~ 3	~~6~~ 4	9	~~1~~ 6	-8	~~8~~ -7

i

↳ any: 5

arr[0] = 4 ⇒ index = 3, Swap(0, 3)

arr[0] = 6 ⇒ index = 5, Swap(0, 5)

arr[0] = 1 ⇒ index = 0, increment i

arr[1] = 2 ⇒ index = 1, increment i

arr[2] = -7 ⇒ irrelevant, increment i

arr[3] = 4 ⇒ index = 3, increment i

arr[4] = 9 ⇒ irrelevant, increment i

⋮

arr[7] = 3 ⇒ index = 2, Swap(7, 2)

// Pseudo code

→ {1, N+1} → {0, N}

```
int missingPositiveInteger(int arr[N]){
    int i=0;
    while (i < N){                                   // irrelevant | already solved
        if(arr[i]<1 || arr[i]>N || i==arr[i]-1)
            i++;
        }
        else{
            int idx = arr[i]-1;
            if(arr[i]==arr[idx]){i++;}
            else{ swap(i, idx); }
        }
    }

    for(int i=0; i<N; i++){
        if (i != arr[i]-1){return i+1;}
    }

    return N+1;
}
```

T.C: O(N)

S.C: O(1)

edge case: → {3, 5}

arr[5] = { 4 (i=0), 1, 3, 5 (i=3), 2 (i=4) }

→ {1, 8}

```
int missingPositiveInteger(int arr[N]){
    int i=0;

    while (i<N) {
        if(arr[i]<1 || arr[i]>N || i==arr[i]-1)
            i++;
        }
        else {
            int idx= arr[i]-1;
            swap(i, idx);
        }
    }

    for (int i=0; i<N; i++){
        if (i != arr[i]-1 ){return i+1;}
    }
    return N+1;
}
```

Annotations: "arr[i]<1 || arr[i]>N" → irrelevant; "i==arr[i]-1" → already solved. Below "i != arr[i]-1": 4, 8.

b arr[8]: 4 6 1 | 2 | -7 3 | 8 4 | 9 | 1 6 | -8 | 3 -7 (indices 0–7)

ans = 5

max no. swapping : N

edge case: arr[5] = { 4 3 3 | 1 | 3 5 3 | 3 4 | 2 } (indices 0–4)

idx = 2

i = 0

leetcode 169: Majority Element → { Amazon, Apple }

↳ Given arr[N], find majority element

↳ An ele with freq > N/2

↳ majority element is definitely present

Ex1: arr[6] = { 1 2 1 6 1 1 } ⇒ ans: 1

↳ 6/2 → >3 occ.

Ex2: arr[9] = { 3 4 4 8 4 9 4 3 4 } ⇒ ans: 4

↳ 9/2 = >4 occ.

Ex3: arr[12] = { 3 3 4 6 1 3 2 5 3 3 3 3 }

↳ 12/2 = >6 occ. ↳ ans: 3

Ex4: arr[10] = { 4 6 5 3 4 5 4 4 4 8 }

↳ 10/2 = >5 occ.

↙ invalid input

↳ no majority element

Note: majority element is present in the array.

//idea1

↳ use nested loop → T.C: $O(N^2)$

//idea2

↳ Sort the array → T.C: $O(N \log N)$

//idea3

↳ use Hashmap to count → T.C: $O(N)$, S.C: $O(N)$

//idea 4 → expected T.C: O(N), S.C: O(1)

↳ Boyer moore voting algo

Obs 1 :

↳ How many majority elements can be Present in one array? → Atmost 1

$freq(maj1) > N/2$

$freq(maj2) > N/2$

$> \frac{N}{2} + > N/2 = > N$ → In total you have only N elements in the array.

Comb! → 15 People to Participate → >7

Obs 2 :

Subhesh :

Nishant :

Abhishek :

Comb 2

→ 15 People to Participate → >7

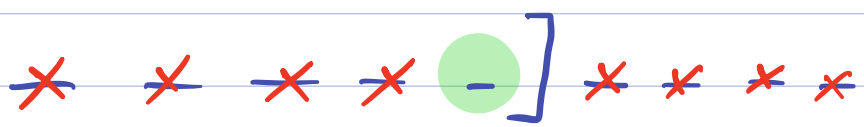
Subhesh: 

Nishant: 

Abhishek: 

>4 occ

→ arr[9] = { 6 6 3 8 8 6 9 6 6 }
0 1 2 3 4 5 6 7 8



↳ if we cancel out 1 majority element no. with 1 non-majority element number, the non-cancelled number at the end is going to be majority elem.

Obs 3:

Is you don't need to know majority element.

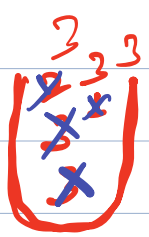
⇓

you Cancel out 2 distinct elements,

(majority, non-majority element)

Summary: Cancel out 2 distinct elements one by one, the uncancelled element is going to be majority element.

Ex: arr[11]: { 3 3 4 6 1 3 2 5 3 3 3 }



i	ele	freq	
0	3	1	
1	3	2	⇒ ans: 3
2	3	1	
3	3	0	
4	1	1	
5	1	0	
6	2	1	
7	2	0	
8	3	1	
9	3	2	

10 3 3

//Pseudo Code

```
int majorityElement(int arr[N]){

    int val = arr[0];
    int count = 1;

    for(int i=1; i<N; i++){
        if(arr[i] == val){
            count++;
        }
        else{
            if(count==0){
                val = arr[i];
                count = 1;
            }
            else{
                count--;
            }
        }
    }

    return val;
}
```

T.C : O(N)

S.C : O(1)

Q) **majority element-2**

↳ Given an **Integer array** of size **n**, find all **elements** that **appear** more than $\lfloor n/3 \rfloor$ times.

(not necessarily majority element is present.)

Ex: 0 1 2 3 4 5 6 7 8 9 10
2 3 2 4 2 3 5 3 3 2 4 → {2, 3}

Ex: 0 1 2 3 4 5 6 7 8 9 10
2 3 4 5 6 2 4 6 5 4 2 → { }

// Boyer moore voting idea

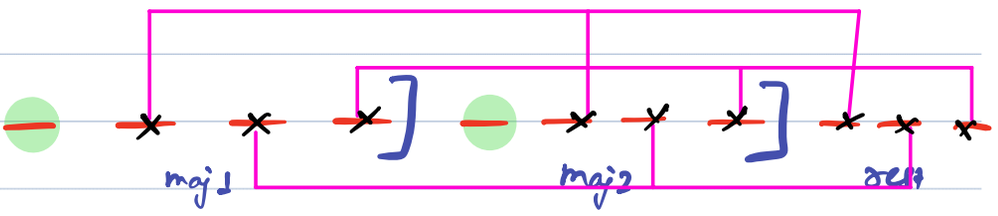
Obs 1: How many **majority elements** can be **present in one array?** → **Atmax 2**

total length = n

$$> \frac{n}{3} + > \frac{n}{3} + > \frac{n}{3}$$

$$= > n \quad \text{XX}$$

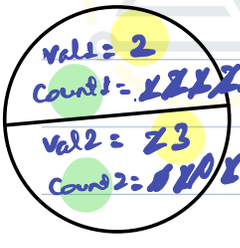
	0	1	2	3	4	5	6	7	8	9	10
Ex:	2	3	2	4	2	3	5	3	3	2	4



→ Cancelling distinct 3 elements at once.
↳ create distinct triplets.

	0	1	2	3	4	5	6	7	8	9	10
Ex:	2	3	2	4	2	3	5	3	3	2	4

v_i



{2, 3, 4} {2, 3, 4}
{2, 3, 5}

```
ArrayList<Integer> ls = new ArrayList<>();
List<Integer> ls = new ArrayList<>();
```

//Pseudo code

```
List<Integer> majorityElement(int[] nums){
    List<Integer> ans = new ArrayList<>();

    int val1 = nums[0];
    int count1 = 1;
    int val2 = -10;
    int count2 = 0;

T.c: O(N)
S.c: O(1)
    for(int i=1; i<nums.length; i++){
        if(nums[i]==val1){
            count1++;
        }
        else if(nums[i]==val2){
            count2++;
        }
        else if(count1==0){
            val1 = nums[i];
            count1 = 1;
        }
        else if(count2==0){
            val2 = nums[i];
            count2 = 1;
        }
        else{
            count1--;
            count2--;
        }
    }
```

```
int c1 = 0;
for (int i=0; i<n; i++) {
    if (nums[i] == val1) { c1++; }
}
if (c1 > n/3) { ans.add(val1); }

int c2 = 0;
for (int i=0; i<n; i++) {
    if (nums[i] == val2) { c2++; }
}
if (c2 > n/3) { ans.add(val2); }

return ans;
}
```


Tracing

```
int val1 = nums[0];
int count1 = 1;
int val2 = -10;
int count2 = 0;

for (int i = 1; i < nums.length; i++) {
    if (nums[i] == val1) {
        count1++;
    }
    else if (nums[i] == val2) {
        count2++;
    }
    else if (count1 == 0) {
        val1 = nums[i];
        count1 = 1;
    }
    else if (count2 == 0) {
        val2 = nums[i];
        count2 = 1;
    }
    else {
        count1--;
        count2--;
    }
}
```

	0	1	2	3	4	5	6	7 (i)	8	9	10
arr:	2	3	2	4	2	3	5	3	3	2	4

val1 = 2

count1 = ~~0~~ ~~1~~ ~~2~~ 1

val2 = ~~-10~~ 3

count2 = ~~0~~ ~~1~~ ~~0~~ 0

{2, 3, 4}

{2, 3, 5}

Q) Next greater element -3

↳ Given a character array with digits of a number at indexes. Find the smallest integer which has exactly the same digit and the value is greater than number in ch[].

Ex: 1 2 → 21

Ex: 9 7 5 3 2 → {-1}

//idea1 → int n

↳ iterate from n+1 till you get the answer & check for every integer if occ are same or not.

//idea 2

ch[]: 1 4 8 4 9 7 5 3 2

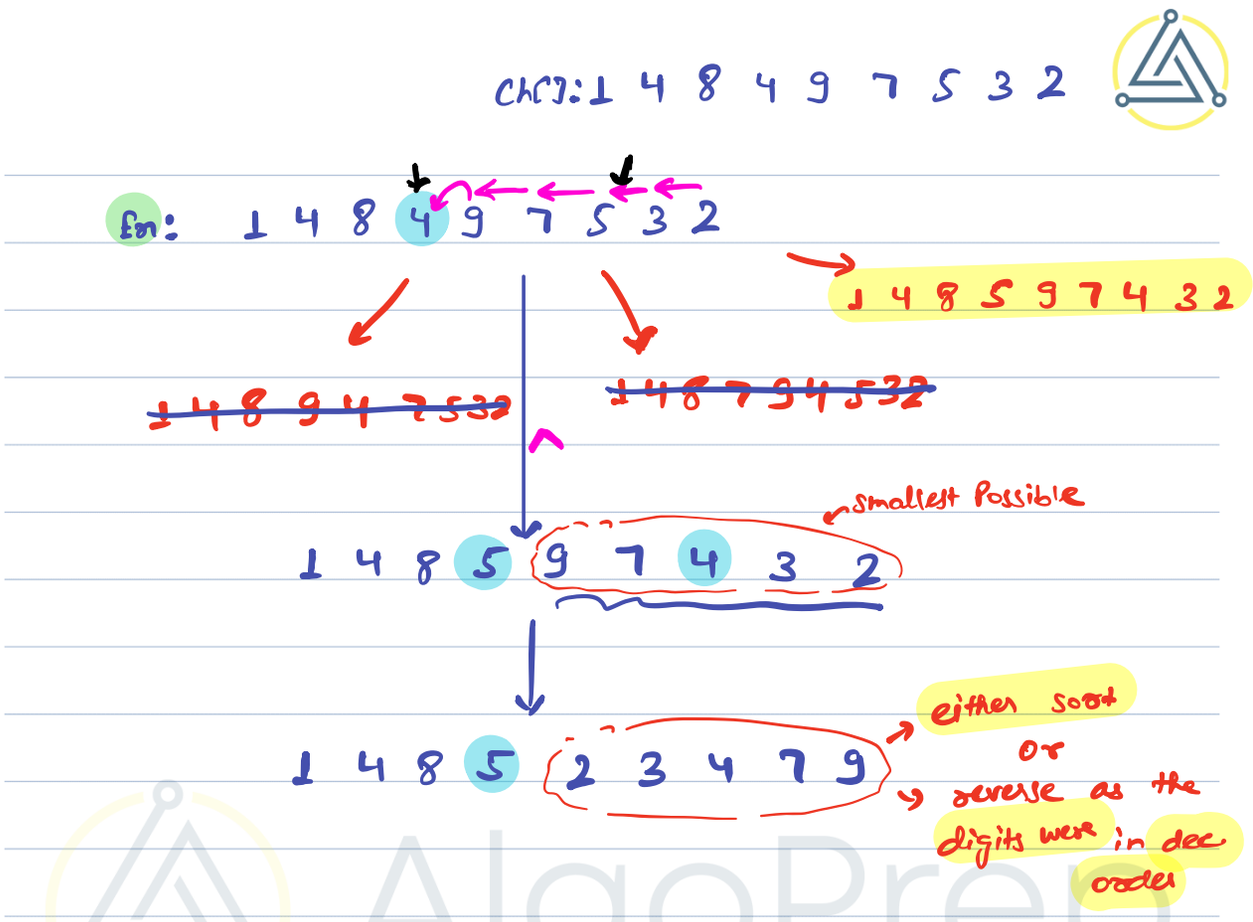
Ex: 1 4 8 4 9 7 5 3 2

2 4 7 1 1 3

2 4 7 2 1 3 > 2 4 7 1 9 2

min(00) max(90)

→ As we want to do minimum increment according to Problem Statement. we target Smaller Place value first using Swapping.


arr[]: 1 4 8 4 9 7 5 3 2
Ex: 1 4 8 4 9 7 5 3 2
1 4 8 5 9 7 4 3 2
1 4 8 9 4 7 5 3 2
1 4 8 7 9 4 5 3 2
smallest possible
1 4 8 5 9 7 4 3 2
1 4 8 5 2 3 4 7 9
either sort
or
reverse as the
digits were in dec
order

//Pseudo code

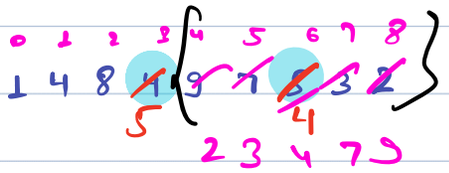
```
public int nextgreaterElement (int n){
    Char [] number = (n+ "").toCharArray();
    int idx=-1;
    for(int i= number.length-2; i>=0; i--){
        if (number[i] < number[i+1]){
            idx = i;
            break;
        }
    }

    if (idx == -1){
        return -1;
    }

    char val = number[idx];
    int Smallest idx = idx+1;

    for (int i= idx+1; i< number.length; i++){
        if ( number[i]>val && number[i]<=
                                number[Smallest idx])
            Smallestidx = i;
        }
    }

    Char temp= number[idx];
    number[idx] = number[smallestidx];
    number[smallestidx] = temp;
```

T.C: O(n) ↙ O(n) O(nlogn)

S.C: O(1)

```
Arrays.sort(numbers, idx+1, numbers.length);

long ans = Long.parseLong(new String(numbers));

    if (ans > Integer.MAX_VALUE){
        return -1;
    }
    else {
        return (int) ans;
    }
}
```

idx = ~~-1~~ 3 val = 4



Smallestidx = ~~4~~ 4 ~~5~~ 6

1 4 8 5 2 3 4 7 9

```
int idx=-1;
for(int i= number.length-2; i>=0; i--){
    if (number[i] < number[i+1]){
        idx = i;
        break;
    }
}

if (idx == -1){
    return -1;
}
char val = number[idx];
int Smallestidx = idx+1;

for (int i= idx+1; i< number.length; i++){
    if (number[i]>val && number[i]<=
                    number[Smallestidx])
        Smallestidx = i;
    }
}
```

```
char temp= number[idx];
number[idx] = number[Smallestidx];
number[Smallestidx] = temp;

Arrays.Sort (number, idx+1, number.length);
```

Max Chunks to make sorted 1

Java Code:

```java
class Solution {
    public int maxChunksToSorted(int[] arr) {
        int maxTillNow = Integer.MIN_VALUE;
        int ans =0;
        for(int i=0;i<arr.length;i++){
            maxTillNow = Math.max(maxTillNow,arr[i]);

            if(i == maxTillNow){
                ans++;
            }
        }

        return ans;
    }
}
```

C++ Code:

```
class Solution {
public:
    int maxChunksToSorted(vector<int>& arr) {
        int maxTillNow = INT_MIN;
        int ans = 0;
        for (int i = 0; i < arr.size(); i++) {
            maxTillNow = max(maxTillNow, arr[i]);

            if (i == maxTillNow) {
                ans++;
            }
        }
```

```
        return ans;
    }
};
```

Python Code:

```
class Solution:
    def maxChunksToSorted(self, arr: List[int]) -> int:
        maxTillNow = float('-inf')
        ans = 0
        for i in range(len(arr)):
            maxTillNow = max(maxTillNow, arr[i])

            if i == maxTillNow:
                ans += 1

        return ans
```

Max Chunks to make sorted 2

Java Code:

Java

```java
class Solution {
    public int maxChunksToSorted(int[] arr) {
        int n = arr.length;
        int[]prefixmax = new int[n];
        int[]suffixmin = new int[n];

        prefixmax[0] = arr[0];
        suffixmin[n-1] = arr[n-1];

        for(int i = 1;i<n;i++){
            prefixmax[i] = Math.max(prefixmax[i-1],arr[i]);
        }
```

```
        for(int i =n-2;i>=0;i--){
            suffixmin[i] = Math.min(suffixmin[i+1],arr[i]);
        }

        int ans = 0;

        for(int i =0;i<n-1;i++){
            if(prefixmax[i] <= suffixmin[i+1]){
                ans++;
            }
        }

        ans++;
        return ans;
    }
}
```

C++ Code:

```
class Solution {
public:
    int maxChunksToSorted(vector<int>& arr) {
        int n = arr.size();
        vector<int> prefixmax(n);
        vector<int> suffixmin(n);

        prefixmax[0] = arr[0];
        suffixmin[n - 1] = arr[n - 1];

        for (int i = 1; i < n; i++) {
            prefixmax[i] = max(prefixmax[i - 1], arr[i]);
        }

        for (int i = n - 2; i >= 0; i--) {
            suffixmin[i] = min(suffixmin[i + 1], arr[i]);
        }

        int ans = 0;

        for (int i = 0; i < n - 1; i++) {
```

```
            if (prefixmax[i] <= suffixmin[i + 1]) {
                ans++;
            }
        }

        ans++;
        return ans;
    }
};
```

Python Code:

```
class Solution:
    def maxChunksToSorted(self, arr: List[int]) -> int:
        n = len(arr)
        prefix_max = [0] * n
        suffix_min = [0] * n

        prefix_max[0] = arr[0]
        suffix_min[n - 1] = arr[n - 1]

        for i in range(1, n):
            prefix_max[i] = max(prefix_max[i - 1], arr[i])

        for i in range(n - 2, -1, -1):
            suffix_min[i] = min(suffix_min[i + 1], arr[i])

        ans = 0

        for i in range(n - 1):
            if prefix_max[i] <= suffix_min[i + 1]:
                ans += 1

        ans += 1
        return ans
```

Print Boundary

Java Code:

```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();

        int[][] mat = new int[n][n];
        for(int i=0;i<n;i++){
            for(int j=0;j<n;j++){
                mat[i][j] = scn.nextInt();
            }
        }

        int i=0;
        int j=0;

        for(int k = 0;k<n-1;k++){
            System.out.print(mat[i][j]+" ");
            j++;
        }

        for(int k = 0;k<n-1;k++){
            System.out.print(mat[i][j]+" ");
            i++;
        }

        for(int k = 0;k<n-1;k++){
            System.out.print(mat[i][j]+" ");
            j--;
        }

        for(int k = 0;k<n-1;k++){
            System.out.print(mat[i][j]+" ");
            i--;
        }
    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;

int main() {
    int n;
    cin >> n;

    int mat[n][n];
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            cin >> mat[i][j];
        }
    }

    int i = 0;
    int j = 0;

    for (int k = 0; k < n - 1; k++) {
        cout << mat[i][j] << " ";
        j++;
    }

    for (int k = 0; k < n - 1; k++) {
        cout << mat[i][j] << " ";
        i++;
    }

    for (int k = 0; k < n - 1; k++) {
        cout << mat[i][j] << " ";
        j--;
    }

    for (int k = 0; k < n - 1; k++) {
        cout << mat[i][j] << " ";
        i--;
    }

    return 0;
}
```

Python Code:

```
n = int(input())

mat = []
for i in range(n):
    row = list(map(int, input().split()))
    mat.append(row)

i, j = 0, 0

for k in range(n - 1):
    print(mat[i][j], end=" ")
    j += 1

for k in range(n - 1):
    print(mat[i][j], end=" ")
    i += 1

for k in range(n - 1):
    print(mat[i][j], end=" ")
    j -= 1

for k in range(n - 1):
    print(mat[i][j], end=" ")
    i -= 1
```

Print Spiral in Square Matrix

Java Code:

```
import java.io.*;
import java.util.*;
```

```
public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();

        int[][] mat = new int[n][n];
        for(int i=0;i<n;i++){
            for(int j=0;j<n;j++){
                mat[i][j] = scn.nextInt();
            }
        }

        int i=0;
        int j=0;
        int steps = n-1;
     while(steps >= 1){
        for(int k = 0;k<steps;k++){
            System.out.print(mat[i][j]+" ");
            j++;
        }

        for(int k = 0;k<steps;k++){
            System.out.print(mat[i][j]+" ");
            i++;
        }

        for(int k = 0;k<steps;k++){
            System.out.print(mat[i][j]+" ");
            j--;
        }

        for(int k = 0;k<steps;k++){
            System.out.print(mat[i][j]+" ");
            i--;
        }
         i++;
         j++;
         steps = steps - 2;
     }

        if(steps == 0){
            System.out.print(mat[i][j]+" ");
        }
    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;

int main() {
    int n;
    cin >> n;

    int mat[n][n];
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            cin >> mat[i][j];
        }
    }

    int i = 0, j = 0;
    int steps = n - 1;

    while (steps >= 1) {
        for (int k = 0; k < steps; k++) {
            cout << mat[i][j] << " ";
            j++;
        }

        for (int k = 0; k < steps; k++) {
            cout << mat[i][j] << " ";
            i++;
        }

        for (int k = 0; k < steps; k++) {
            cout << mat[i][j] << " ";
            j--;
        }

        for (int k = 0; k < steps; k++) {
            cout << mat[i][j] << " ";
            i--;
        }

        i++;
        j++;
        steps = steps - 2;
    }
```

```
    if (steps == 0) {
        cout << mat[i][j] << " ";
    }

    return 0;
}
```

Python Code:

```
n = int(input())
mat = []

for _ in range(n):
    row = list(map(int, input().split()))
    mat.append(row)

i, j = 0, 0
steps = n - 1

while steps >= 1:
    for k in range(steps):
        print(mat[i][j], end=" ")
        j += 1

    for k in range(steps):
        print(mat[i][j], end=" ")
        i += 1

    for k in range(steps):
        print(mat[i][j], end=" ")
        j -= 1

    for k in range(steps):
        print(mat[i][j], end=" ")
        i -= 1

    i += 1
    j += 1
    steps -= 2

if steps == 0:
    print(mat[i][j], end=" ")
```

Spiral matrix

Java Code:

Java

```java
class Solution {
    public List<Integer> spiralOrder(int[][] matrix) {
        List<Integer> ans = new ArrayList<>();
        int row = matrix.length;
        int col = matrix[0].length;

        int i = 0;
        int j = 0;
        int rsteps = row - 1;
        int csteps = col - 1;
        int count = 0;

    while(rsteps >= 1 && csteps >= 1){
        for(int k = 0;k < csteps;k++){
            ans.add(matrix[i][j]);
            j++;
            count++;
        }

        for(int k = 0;k < rsteps;k++){
            ans.add(matrix[i][j]);
            i++;
            count++;
        }

        for(int k = csteps;k >= 1;k--){
            ans.add(matrix[i][j]);
            j--;
            count++;
        }

        for(int k = rsteps;k >= 1;k--){
            ans.add(matrix[i][j]);
```

```
            i--;
            count++;
        }

        rsteps = rsteps - 2;
        csteps = csteps - 2;
        i++;
        j++;
    }

    if(rsteps == 0){
        for(int k = 0;k < csteps + 1;k++){
            ans.add(matrix[i][j]);
            j++;
        }
    }else if(csteps == 0){
        for(int k = 0;k < rsteps + 1;k++){
            ans.add(matrix[i][j]);
            i++;
        }
    }
        return ans;
    }
}
```

C++ Code:

```
class Solution {
public:
    vector<int> spiralOrder(vector<vector<int>>& matrix) {
        vector<int> ans;
        int row = matrix.size();
        int col = matrix[0].size();

        int i = 0;
        int j = 0;
        int rsteps = row - 1;
        int csteps = col - 1;
        int count = 0;
```

```
        while (rsteps >= 1 && csteps >= 1) {
            for (int k = 0; k < csteps; k++) {
                ans.push_back(matrix[i][j]);
                j++;
                count++;
            }

            for (int k = 0; k < rsteps; k++) {
                ans.push_back(matrix[i][j]);
                i++;
                count++;
            }

            for (int k = csteps; k >= 1; k--) {
                ans.push_back(matrix[i][j]);
                j--;
                count++;
            }

            for (int k = rsteps; k >= 1; k--) {
                ans.push_back(matrix[i][j]);
                i--;
                count++;
            }

            rsteps -= 2;
            csteps -= 2;
            i++;
            j++;
        }

        if (rsteps == 0) {
            for (int k = 0; k < csteps + 1; k++) {
                ans.push_back(matrix[i][j]);
                j++;
            }
        } else if (csteps == 0) {
            for (int k = 0; k < rsteps + 1; k++) {
                ans.push_back(matrix[i][j]);
                i++;
            }
        }
        return ans;
    }
};
```

Python Code:

```
class Solution:
    def spiralOrder(self, matrix):
        ans = []
        row = len(matrix)
        col = len(matrix[0])

        i = 0
        j = 0
        rsteps = row - 1
        csteps = col - 1
        count = 0

        while rsteps >= 1 and csteps >= 1:
            for k in range(csteps):
                ans.append(matrix[i][j])
                j += 1
                count += 1

            for k in range(rsteps):
                ans.append(matrix[i][j])
                i += 1
                count += 1

            for k in range(csteps, 0, -1):
                ans.append(matrix[i][j])
                j -= 1
                count += 1

            for k in range(rsteps, 0, -1):
                ans.append(matrix[i][j])
                i -= 1
                count += 1

            rsteps -= 2
            csteps -= 2
            i += 1
            j += 1

        if rsteps == 0:
            for k in range(csteps + 1):
                ans.append(matrix[i][j])
                j += 1
        elif csteps == 0:
            for k in range(rsteps + 1):
                ans.append(matrix[i][j])
```

```
            i += 1

        return ans
```

Flip Bits

Java Code:

```
class Solution {

    public static int maxOnes(int a[], int n) {
        int count = 0;
        for(int i=0;i<n;i++){
            if(a[i] == 0){
                a[i] = 1;
            }else{
                a[i] = -1;
                count++;
            }
        }

        int csum = 0;
        int osum = Integer.MIN_VALUE;
        for(int i=0;i<n;i++){
            if(csum >= 0){
                csum += a[i];
            }else{
                csum = a[i];
            }

            osum = Math.max(csum,osum);
        }

        if(osum > 0){
            return osum + count;
        }else{
            return count;
        }


    }
}
```

C++ Code:

```
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;

class Solution {
public:
    static int maxOnes(vector<int>& a, int n) {
        int count = 0;
        for (int i = 0; i < n; i++) {
            if (a[i] == 0) {
                a[i] = 1;
            }
            else {
                a[i] = -1;
                count++;
            }
        }

        int csum = 0;
        int osum = INT_MIN;
        for (int i = 0; i < n; i++) {
            if (csum >= 0) {
                csum += a[i];
            }
            else {
                csum = a[i];
            }

            osum = max(csum, osum);
        }

        if (osum > 0) {
            return osum + count;
        }
        else {
            return count;
        }
    }
};

int main() {
```

```
    int n;
    cin >> n;
    vector<int> a(n);

    for (int i = 0; i < n; i++) {
        cin >> a[i];
    }

    int result = Solution::maxOnes(a, n);
    cout << result << endl;

    return 0;
}
```

Python Code:

```
class Solution:
    @staticmethod
    def maxOnes(a, n):
        count = 0
        for i in range(n):
            if a[i] == 0:
                a[i] = 1
            else:
                a[i] = -1
                count += 1

        csum = 0
        osum = float('-inf')
        for i in range(n):
            if csum >= 0:
                csum += a[i]
            else:
                csum = a[i]

            osum = max(csum, osum)

        if osum > 0:
            return osum + count
        else:
            return count

# Input
n = int(input())
a = list(map(int, input().split()))
```

```
# Call the function and print the result
result = Solution.maxOnes(a, n)
print(result)
```