FREEMIND

Date

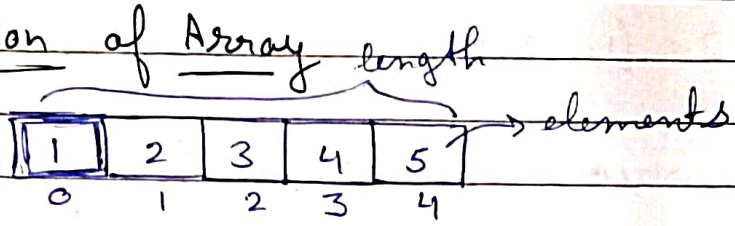
Page

Lecture 12 (Arrays 1)

What is array?

→ data structure which stores a collection of homogeneous items
→ contiguous memory.

Representation of Array



0-indexed

{index → location of every element.}

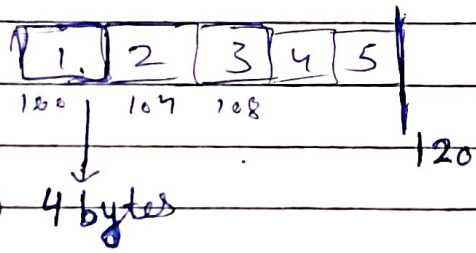
Syntax for array declaration

datatype arrayname [array size]

int array [5]

initialize it ⇒ int array [ ] = {1,2,3,4,5}

Memory allocations in Arrays → contiguous
→ consecutive blocks of memory



Arrays Types:-
- Single Dimenssional or one-Dimenssional Array
- Multidimenssional Array

Size operation in arrays:- int array [ ] = {1,2,3,4};

4×4=16

length → sizeof (array) → 16 / size of (array[0]) → 4

= 4

FREEMIND

Date

Page

Jab bhi array ko print karvao toh usse initialize zarur karna nhi toh jo bhi garbage value present hogi array me vo print hojaegi.

(A) → Printing array using loop. (Traverse)

```
int array[] = {1,2,3,4};

int size = sizeof(array)/sizeof(array[0]);

//for loop
for(int i = 0 ; i < size ; i++){
    cout << array[i] << endl;
}

//for each loop
for(int ele : array){
    cout << ele << endl;          //traverse every element
}                                   of the array

//While loop
int idx = 0
while(idx < size){
    cout << array[idx] << endl;
    idx++
}
```

(A) Taking input for array :-

```
int n;  //no. of elements in array
cin >> n;
char array[n];   (Any datatype)

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

FREEMIND

Date

Page

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

for (int &element : array) {    // for each loop
    cin >> element;
}
for (int i=0; i<n; i++) {
    cout << array[i] << " ";
}
```

Q⇒ Calculate the sum of all elements in the given array.

Sol⇒
```
int main() {
    int arr[] = {1, 2, 3, 4};
    int size = sizeof(arr) / sizeof(arr[0]);
    int sum = 0;
    for (int i=0; i<size; i++) {
        sum += arr[i];
    }
    cout << sum << endl;
    return 0;
}
```

Q⇒ Find the max. value out of all the elements in the array.

```
int arr[] = {3, 7, 1, 8, 9, 11}
int max = arr[0]
for (int i=0; i<5; i++) {
    if (arr[i] > max) {
        max = arr[i]
    }
}
cout << max << endl;
```

FREEMIND

Date

Page

⭐ Linear Search:

Q- Search if a given element is present in the array or not. If it is not present then return -1 else return index.

Sol:-

```
int arr[] = {3, 9, 18, 11, 7};
int item = 11;
int ans = -1;
for (int i = 0; i <= 5; i++) {
    if (arr[i] == item) {
        ans = i;
        break;
    }
}
cout << ans << endl;
return 0;
```

Lecture - 13 (Arrays-2)

⭐ Vectors in C++

→ dynamic arrays → resize when insert/delete elements

→ contigous memory

⟹ Basic operations in vectors:-

1) Declaration

```
# include <vector>

vector <datatype> Vector_name;
```

or) `Vector <datatype> Vector_name (size);`

eg: `Vector <int> V;`

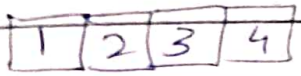
FREEMIND

Date

Page

2. Size :-

V.size() ⟶ length

eg:  ⟶ V.size() = 4

3) Resize :-

V.resize(new size);

eg: V.resize(8);

V.size() = 8

4) Capacity :-

V.capacity()

Capacity ≥ size

Note :- Capacity hamesha 2 ki powers me increase karti hai (Jab ek-2 element add karrahe hain)

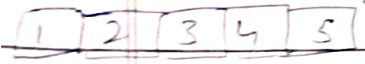
→ But jab resize karvate hai to compiler k acc. increase karti hai.

Vector<int> V size = 0, capacity = 0

 size = 1, capacity = 1

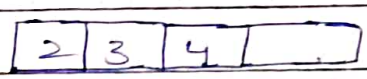
 size = 2, capacity = 2

 size = 3, capacity = 4

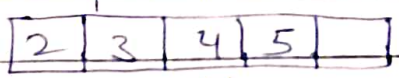
 size = 5, capacity = 8

5) Add elements :-

→ V.pushback(element)



V.pushback(5)



→ V.insert(position, element)

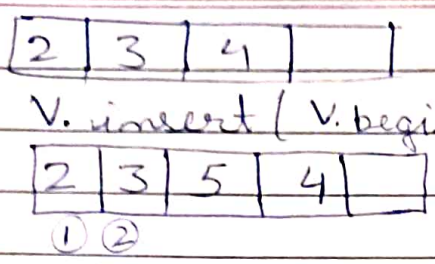
position ↓

relative to position of first element. ⟶ V.begin()

V.end()

Date

Page



6) Delete Elements

→ V.pop_back() {delete element from end}

add ki jgah Del.

→ V.erase(position)

↓

relative to start / end position of vector

Similar to previous eg:

⭐ Looping in Vector:- → for loop.
→ for each
→ while.

```
int main(){
    Vector<int> V;
    //for loop
    for(int i=0; i<5; i++){
        int element;
        cin>> element;
        V.push_back(element);
        //(cin>> V[i]; (but size must be defined)
    }
    for(int i=0; i<5; i++){
        cout<< V[i]<<" ";
    }
    cout<<endl;
//for each loop
    for(int ele:V){
        cout<<ele<<" ";
    } cout<<endl;
```

FREEMIND

Date

Page

```
// while loop
int idx = 0;
    while (idx < V.size()) {
        cout << V[idx++] << " ";
    }
return 0;
}
```

Q1⇒ Find the last occurance of an element x in a given array.

46.

Sol⇒

```
int main() {
    Vector<int> V[6];
    for (int i=0; i<6; i++) {
        cin >> V[i];
    }
    cout << "Enter x:";
    int x;
    cin >> x;
    int occurance = -1;
    for (int i=0; i<V.size(); i++) {
        if (V[i] == x) {
            occurance = i;
        }
    }
    cout << occurance << endl;
    return 0;
}
```

```
for (int i = V.size()-1; i>=0; i--)
{
    if (V[i] == x) {
        occurance = i;
        break;
    }
}
```

FREEMIND

Date

Page

Q⇒ Count the no. of occurances of a particular element x.

Sol⇒

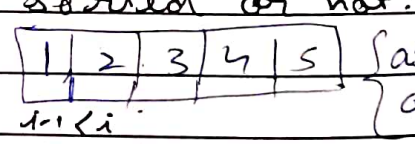
```
int main(){
    vector<int> v[6];
    input -> vector;
    input -> x ; int occurance = 0;
    for(int i=0; i<v.size(); i++){
        if (v[i] == x){
            occurance++;
        }
    }
    cout<<occurance<<endl;
    return 0;
}
```

Q⇒ Count the no. of elements strictly greater than value x

```
int count = 0
for (int i=0; i<v.size(); i++){
    if (v[i] > x){
        count++;
    }
}
cout<<count<<endl;
```

Q⇒ Chuck if the given array is sorted or not.

Logic: a[i] > a[i-1]



Sol⇒

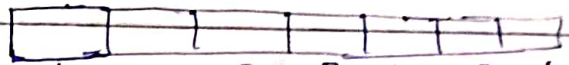
```
int array = {1,2,3,4,5,6};
bool sortedflag = true;
for (int i = 0; i<6; i++){
    if (array[i] <= array[i-1]){
        sortedflag = false;
    }
} cout<<sortedflag<<endl;
```

FREEMIND

Date

Page

Q⇒ Find the difference b/w the sum of elements at even indices to the sum of elements at odd indices.



(even sum - odd sum)

sum + even - odd
idx idx

Sol:
```
int main() {
  int arr[] = {1,2,3,4,5,6};
  int sum = 0;
  for (int i=0; i<6; i++) {
    if (arr[i]%2 == 0) {
      sum += arr[i]
    }
    else {
      sum -= arr[i]
    }
  } cout << sum << endl;
  return 0;
}
```

Lecture 14 — Arrays - 3

(Practice Session)

Q⇒ Find the total no. of pairs in the array whose sum is equal to the given value of x.

Sol→
```
int array = {3,4,6,7,1};
int targetsum = 7;
int size = 5; pairs = 0;
for (int i=0; i<size; i++) {
  for (int j=i+1; j<size; j++) {
    if (array[i] + array[j] = targetsum) {
      pairs++;
    }
  }
}
cout << pairs << endl;
```

FREEMIND

Date

Page

Q ⇒ Count the no. of triplets whose sum is equal to the given value X.

Sol:-

```
int arr[] = {3,1,2,4,0,6};
int targetsum = 6;
int size = 6;
for (int i=0; i<size; i++){
    for (int j=i+1; j<size; j++){
        for (int k=j+1; k<size; k++){
        if (arr[i] + arr[j] + arr[k] = targetsum){
            triplets++;
        }
        }
    }
}
Cout<<triplets<<endl;
```

Pattern : Array Manipulation

Q ⇒ Find the unique no. in a given array where all the elements are being repeated twice with one value being Unique.

Sol:-

```
int arr[] = {2,3,1,3,2,4,1}
int size = 7
for (int i=0; i<size; i++){
    for (int j=i+1; j<size; j++){
        if (arr[i] = arr[j]){
            arr[i] = arr[j] = -1
        }
    }
}
```

FREEMIND

Date

Page

```
for (int i=0; i<size; i++) {
    if (arr[i]>0) {
        cout << arr[i] << endl;
    }
}
return 0;
}
```

Q> Find Second largest element in the given array

-> Same as finding max element two times.
-> 1> Find max element and manipulate it by given value (-1)
2> Again Find max element. (second largest element of array)

{Hint: Make a funct. to find, max, Element}
(i) (index of) then a[i] = -1

```
#include<iostream>
Using namespace std;
int largestelement (int array[], int size) {
    int max = INT_MIN; (INT_MIN) //min. int. in C++
    for (int i=0; i<size; i++) {
        if (array[i] > max) {
            max = array[i]; int maxindex = i;
        }
    }
    return maxindex;
}
```

again on Next page

Q> If Duplicates of largest elements are also present in array:-
the all of these duplicates are also manipulated by applying loop. as shown on next page

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

int largestElementIndex(int arr[], int size){
    int max = INT_MIN;
    int maxindex = -1;
    for(int i=0; i<size; i++){
        if(arr[i] > max){
            max = arr[i];
            maxindex = i;
        }
    }
    return maxindex;
}
int main(){
    int arr[] = {2,3,5,7,6,1,7};
    int indexoflargest = largestElementIndex(arr,7);
    cout<<arr[index of largest]<<endl;

    //arr[indexoflargest] = -1;
    int indexofsecondlargest = largestelementindex(arr,7);

    cout<<arr[indexofsecondlargest]<<endl;

    return 0;
}
```

Ques → int largestElement = arr[indexoflargest];
```
for(int i=0; i<7; i++){
    if(arr[i] = largestElement){
        arr[i] = -1;
    }
}
```

Another Method For Same Question

Date
Page

→ // 2, 3, 5, 7, 6, 1, 7
// max = ?
// if (a[i] > max) max = a[i]

// Second_max = ?
// if (a[i] > second_max && a[i] != max) second_max = a[i]

Q:- Rotate the given array 'a' by k steps, where k is non-negative.
Note:- k can be greater than n as well where n is the size of array 'a'.

for k=2 → [1, 2, 3, 4, 5] → [4, 5, 1, 2, 3]
(n-k) (k) (last k elements assigning)

Sol:- int main() {
int array[] = {1, 2, 3, 4, 5};
int n = 5
int k = 7
// k can be greater than n.
k = k % n;
int ansarray[5];
int j = 0;
// inserting last k elements in k array
for(int i = n-k; i < n; i++) {
ansarray[j++] = array[i];
}
// inserting at n-k elements in ans array
for (int i = 0; i <= k; i++) {
ansarray[j++] = array[i];
}

```
for (int i=0; i<n; i++){
    cout<<ansarray[i]<<" ";
}
cout<<endl;
return 0;
}
```

OR By using reverse(V.begin(), V.end());

```
#include<iostream>
#include<vector>
using namespace std;
int main(){
    vector<int> V;
    V.push_back(1);
    V.push_back(2);
    V.push_back(3);
    V.push_back(4);
    V.push_back(5);
    int k=2;
    k=k%V.size();

    reverse(V.begin(),V.end());          // 5 4 3 2 1
    reverse(V.begin(),V.begin()+k);      // 4 5 3 2 1
    reverse(V.begin()+k,V.end());        // 4 5 1 2 3

    for(int a:V){
        cout<<a<<" ";
    }cout<<endl;
```

Q⇒ Given Q queries, check if the given no. is present in the array or not.

Note: Value of all the elements in the array is less than 10 to the power 5.

FREEMIND

Date

Steps

1) Vector input
2) freq.[V[i]] ++
3) q(query) input
4) freq.[query] output

```
int main (){
    int n;
    cin>>n;
    Vector<int> V(n);
    for (int i=0; i<n; i++){
        cin>>V[i];
    }
    const int N= 1e5 +10;  //10^5
    Vector <int> freq(N,0);
    for (int i=0; i<n; i++){
        freq[V[i]]++;
    }
    cout<<"Enter queries:";
    int q;
    cin>>q;

    while (q--){
        int queryelement;
        cin>>queryelement;
        cout<<freq[queryelement]<<endl;
    }
    return 0;
}
```

(A) Summary :-

1) Target sum
2) Array Manipulation
3) Rotate by k steps
4) Queries → frequency array

Date

Page

Lecture - 15 (Problem session-2)

Qt⇒ Sort an array consisting of only 0s and 1s.

Ans

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

void sortzeroesandones(vector<int> &v){
    int zeroes_count = 0;
    for(int ele : v){
        if(ele == 0){
            zeroes_count++;
        }
    }
    for(int i=0; i<v.size(); i++){
        if(i < zeroes_count){
            v[i] = 0;
        }
        else{
            v[i] = 1
        }
    }
}
int main(){
    int n;
    cin>>n;
    vector<int> v;
    for(int i=0; i<n; i++){
        int ele; cin>>ele;
        v.push_back(ele);
    }
    sortzeroesandones(v);
```

Date

Page

```
for (int i=0; i<n; i++){
    cout<<v[i]<<" "
} cout<<endl;
}
return 0;
}
```

[or]

Pattern: Two Pointers

```
Void SortzeroesAndones (Vector<int>&V){
    int left_ptr=0;
    int right_ptr=V.size()-1;
    while (left_ptr<right_ptr){
        if (V[left_ptr]==1 && V[right_ptr]==0){
            V[left_ptr++]=0;
            V[right_ptr--]=1;
        }
        if (V[left_ptr]==0){
            left_ptr++
        }
        if (V[right_ptr]==1){
            right_ptr--
        }
    }
    return;
}
int main(){
    //Same as Previous Code
}
```

-3, -2, -1, 0, 1, 2, 3, 4
± -2 3 (4)
16, 9, 9

FREEMIND
Date
Page

Q2⇒ Given an array of integers 'a', move all the even integers at the beginning of the array followed by all the odd integers. The relative order of odd or even integers does not matter. Return any array that satisfies the conditions.

Ans: Full code similar to previous Q1 code

Except:

```
Void SortbyParity(Vector<int>&V){
    ~~
    ~~
    while( ~~ ){
        if (V[left_ptr]%2==1 && V[right_ptr]%2==0)
        swap(V[left_ptr], V[right_ptr]);
        left_ptr++ ; right_ptr-- ;
        }
    ~~~~
    Same
}
```

Q3⇒ Given an integer array 'a' sorted in non-decreasing order, return an array of the squares of each number sorted in non-decreasing order.

Ans:

```
Void SortedSquareArray(Vector<int>&V){
    vector<int> ans;
    int left_ptr=0
    int right_ptr = V.size()-1;
    while (left_ptr < right_ptr){
        if( abs(V[left_ptr]) < abs(V.[right_ptr])){
            ans.push_back(V[right_ptr] * V[right_ptr]);
```

```
            right_ptr--
        }
        else{
            ans.push_back(V[left_ptr] * V[left_ptr]);
            left_ptr++;
        }
    }

    for(int i=0; i<V.size(); i++){
        cout<<ans[i]<<" ";
    } cout<<endl;
}
int main(){
        Same as Q1
    }
```

LECTURE - 16 — Problems on arrays - 3

★ Pattern: Prefix Sums

1 2 3 4 5
1 3 6 10 15

Q ⇒ Given an integers array 'a', return the prefix sum/running sum in the same array without creating a new array

Ans:⇒
```
Void runningSum(Vector<int> &V){
    for(int i=1; i<v.size(); i++){
        V[i] = V[i-1] + V[i];
    }
    return;
}
int main(){
    Same as Q1 (Prev. lec.)
}
```

FREEMIND

Date

Page

3

7.

Q22 Check if we can partition the array into two subarrays with equal sum. More formally, check that the prefix sum of a part of the array is equal to the suffix sum of rest of the array.

① Total
② Step by Step prefix
③ Suffix = Total - prefix
④ When Suffix == Prefix
⑤ Print true

Sol:-

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

bool checkprefixsuffixsum(vector<int> &v){
    int total_sum = 0
    for(int i=0; i<v.size(); i++){
        total_sum += v[i];
    }
    int prefix_sum = 0;
    for(int i=0; i<v.size(); i++){
        prefix_sum += v[i];
        int suffix_sum = total_sum - prefix_sum;

        if(suffix_sum == prefix_sum){
            return true;
        }
    }
    return false;
}

int main{
    Same as Q1 (prev. lec.)

    Cout << checkprefixsuffixsum(v) << endl;
}
```

FREEMIND

Date

Page

Q⇒ Given an array of integers of size n. Answer q queries where you need to print the sum of values in a given range of indices from l to r (both included)

Note:⇒ The values of l and r in queries follow 1-based indexing

Ans:⇒

```
int main(){
    int n;
    cin>>n;
    vector<int> v(n+1,0);
    for(int i=1; i<=n; i++){
        cin>>v[i];
    }
    //Calculate prefix sum array
    for(int i=1; i<=n; i++){
        v[i]+=v[i-1];
    }
    int q;
    cin>>q;
    while(q--){
        int l,r;

        int ans=0;
        //ans = prefix_sumarray[r] - prefix_sumarray[l-1]
        ans = v[r] - v[l-1];
        cout<<ans<<endl;
    }
    return 0;
}
```

1) n (input)
2) Vector (input)
3) prefix sum of
   v[i] += v[i-1]
4) q (input)
5) while(q--)
   ans = v[r] - v[l-1]
6) cout<<ans<<endl

FREEMIND

Date

Page

Lecture-17 2D Arrays in C++

★ Multidimensional array :-

datatype arrayname [size 1][size 2] ——— [size n]

★ 2D Array :

datatype arrayname [row][column];

a[i][j] → column no.
→ row no.

a[n][m]
↳ elements ⇒ n*m

a[2][2] →

a[0][0]	a[0][1]	a[0][2]
a[1][0]	a[1][1]	a[1][2]
a[2][0]	a[2][1]	a[2][2]

★ Methods to initialize 2D Array :-

int array[2][3] = {1,2,3,4,5,6};

int array[2][3] = {{1,2,3},{4,5,6}};

⇒ 3D Array :

⇒ 3 x 2 x 4

Means ⇒ 3 (2D) arrays of 2x4

0 [[0][0][0] 001 002 003
010 011 012 013

1 [[1][0][0] 112 113

2 [[2][0][0] 213

FREEMIND

Date

Page

⋆ Taking 2D array as input

```
for(int i=0; i<row; i++){
    for(int j=0; j<col.; j++){
        ~~a[i][j]~~
        cin>>arr[i][j]
    }
}
```

Q⋙ Why Multidimensional Arrays ?

→ representing grids
→ faster access
→ predefined size

Q.⋙ Write a program to display multiplication of two matrices entered by the user.

$[r_1 \times c_1]$ $[r_2 \times c_2]$

$c_1 = r_2$

resultant matrix is $r_1 \times c_2$

logic

```
i → 0 to r1
    j → 0 to c2
        K → 1 to c1/r2
        C[i][j] += A[i][K] * B[K][j]
```

sol is

```
int main(){
    int r1, c1;
    cin>>r1>>c1
    int A[r1][c1]
    for(int i=0; i<r1; i++){
        for(int j=0; j<c1; j++){
            cin>>A[i][j];
        }
```

```
int r2, c2;
cin >> r2 >> c2;
int B[r2][c2]
for (int i=0; i<r2; i++){
    for (int j=0; j<c2; j++){
     cin >> B[i][j];
    }
}
if (c1!=r2){
  cout<<"Matrix Multiplication is not possible for
        this input"<<endl;
}
int C[r1][c2];

for( int i=0; i<r1; i++){
   for( int j=0; j<c2; j++){ C[i][j]=0;
      for (int k=0; k<r2; k++){
         C[i][j] += A[i][k] * B[k][j];
      }
   }
}
for (int i=0; i<r1; i++){
   for (int j=0; j<c2; j++){
      cout << C[i][j] << " ";
   } cout<<endl;
}
return 0;
}
```

Q:- Write a program to display transpose of matrix entered by user.

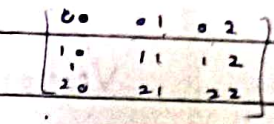
FREEMIND

Date

Page

Ans ⇒

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



[3×3]

Vector<Vector<int>>V(n,Vector<int>(m,0))

FREEMIND

Date

Page

Lecture-18 / Problems on 2D Arrays-II

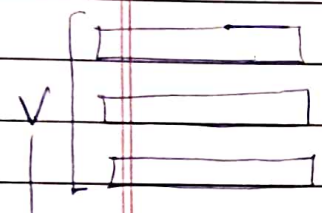
→ 2D Vectors

→ Pascal's Triangle Problems

(A) 2D Vectors → Vector of Vector Datatype

Vector<Vector<datatype>> Vector_name

⇒ initialization



Vector<int> V1 = {1,2,3};
Vector<int> V2 = {4,5};
Vector<int> V3 = {6,7};

Vector<Vector<int>> V = {V1, V2, V3};

{{1,2,3},{4,5},{6,7}}

Vector → n × m

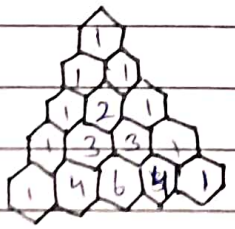
(B) 2D Vector

Vector<Vector<int>> V(n, Vector<int>(m));

Vector → 3 × 4 → 2D Vector → 0 values

Vector<Vector<int>> V(3, Vector<int>(4,0));

(C) Pascal's Triangle Problem



① ${}^0C_0$
② ${}^1C_0$ | ${}^1C_1$
③ ${}^2C_0$ | ${}^2C_1$ | ${}^2C_2$
④ ${}^3C_0$ | ${}^3C_1$ | ${}^3C_2$ | ${}^3C_3$
⑤ ${}^4C_0$ | ${}^4C_1$ | ${}^4C_2$ | ${}^4C_3$ | ${}^4C_4$

① Columns = row_number

② a[i][j] = ${}^iC_j$

$${}^nC_r = \frac{n!}{r!(n-r)!}$$

③ a[i][j] = a[i-1][j] + a[i-1][j-1]

n^{th} row → (j = 0) → ${}^nC_0 = 1$

→ (i = n) → ${}^nC_n = 1$

Date
Page

Q1 Give an integer n, return the first n rows of pascal's triangle.

Sol:-

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

vector<vector<int>> pascalTriangle(int n) {

    vector<vector<int>> pascal(n);

    for(int i=0; i<n; i++) {
        pascal[i].resize(i+1);
        for(int j=0; j<i+1; j++) {
            if(j==0 || j==i) {
                pascal[i][j] = 1
            }
            else {
                pascal[i][j] = pascal[i-1][j] + pascal[i-1][j-1];
            }
        }
    }
    return pascal;
}
int main() {
    int n; cin>>n;
    vector<vector<int>> ans;
    ans = pascalTriangle(n);
    for(int i=0; i<ans.size(); i++){
        for(int j=0; j<ans[i].size(); j++){
            cout<<ans[i][j]<<" ";
        } cout<<endl;
    } return 0;
}
```

FREEMIND

Date

Page

Lecture - 19 Advanced Problems on 2D-Arrays

Q→ Given a boleen 2D array, where each row is sorted. Find the row with the maximum no. of 1's.

Input: row = 3, col = 4

matrix[] = { {0,1,1,1}
{0,0,0,1}
{0,0,0,1} }

output ⇒ 0

	0	1	2	3
0	0	1	1	1
1	0	0	0	1
2	0	0	0	1

0th row; first occurance of 1 = 1

no. of ones = columns - index

= 4 - 1 = 3

Ans:

```
#include<iostream>
using namespace std;
int maximumOneRow(vector<vector<int>&V){

    int maxOnes = INT_MIN;
    int maxOnesRow = -1;
    int columns = V[0].size();
    for(int i=0; i<V.size(); i++){
        for(int j=0; j<V[i].size(); j++){
            if(V[i][j]==1){
                int numberOfOnes = columns - j;
                if (numberOfOnes > maxOnes){
                    maxOnes = numberOfOnes;
                    maxOneRow = i;
                } break;
            }
        }
    }
    return maxOnesRow;
}
```

(columns vale loop se pahle same k liye)

FREEMIND

Date

Page

```
int main(){
    int n, m;
    cin>>n>>m;
    vector<vector<int>> vec(n, vector<int>(m));
    for(int i=0; i<n; i++){
        for(int j=0; j<m; j++){
            cin>>vec[i][j];
        }
    }
    int res = maximumOnesRow(vec);
    cout<<res<<endl;

    return 0;
}
```

22174

→ To find: left most one (Same Question)

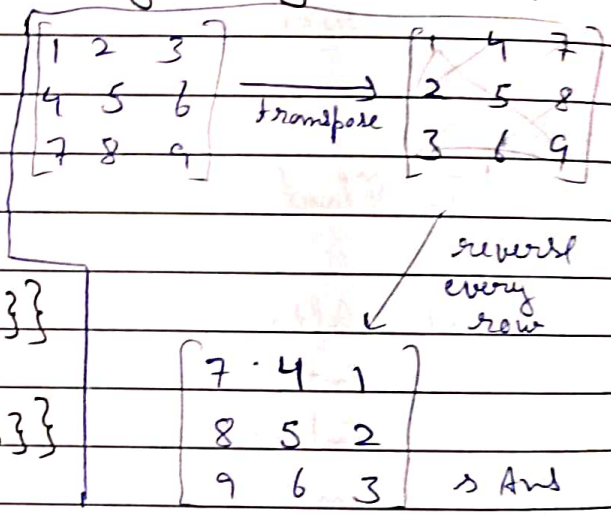
```
int leftmostoneRow(vector<vector<int>>&v){
    int leftmostone = -1;
    int maxonesRow = -1;
    int j = v[0].size()-1;

    //finding leftmost one in 0th Row.
    while(j>=0 && v[0][j]==1){
        leftmostone = j;
        maxonesRow = 0;
        j--;
    }
    //check in rest of the rows if we can find a one
                                    left to the leftmost one
    for(int i=1;
```

J phle hi kam ho chuka hai or ab Jo bhi check hoga Vo J se pichhe hoga)

```
for (int i=1; i<V.size(); i++) {
    while (j>=0 && V[i][j]==1) {
        leftmostone = j;
        j--;
        maxOnesRow = i;
    }
}
return maxOnesRow;
}
```

Q ⇒ Given a square matrix, turn it by 90 degrees in a clockwise direction without using any extra space.

1	2	3	7	4	1
4	5	6	⟹	8	5	2
7	8	9	9	6	3



Input:- {{1,2,3},{4,5,6},{7,8,9}}

Output:- {{7,4,1},{8,5,2},{9,6,3}}

Ans ⇒

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

void rotateArray(vector<vector<int>> &Vec) {

    int n = Vec.size();
    // transpose
    for (int i=0; i<n; i++) {
        for(int j=0; j<i; j++) {
            swap(Vec[i][j], Vec[j][i]);
        }
```

```
//reverse every row
for(int i=0; i<n; i++) {
    reverse(Vec[i].begin(), Vec[i].end());
}
return;
}
```

1) n input
2) Vector Form
3) Vector input
4) Funct. call
5) Vec (Print)

```
int main() {
    int n;
    cin>> n;
    Vector<Vector<int>> Vec(n, Vector<int>(n));

    for(int i=0; i<n; i++) {
        for(int j=0; j<n; j++) {
            cin>> Vec[i][j];
        }
    }
    rotatearray(Vec);
    for(int i=0; i<n; i++) {
        for(int j=0; j<n; j++) {
            cout<< Vec[i][j] << " ";
        } cout<< endl;
    }

    return 0;
}
```

FREEMIND

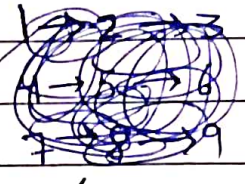
Date

Page

Lecture -20 Advanced Problems on 2D Arrays -3

⭐ Pattern: Spiral Matrix

Q→ Given an n×m matrix 'a', return all elements of the matrix in spiral order.



Sol.⇒

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

Void SpiralOrder (Vector<Vector<int>> &matrix){
 int left =0;
 int right = matrix[0].size()-1;
 int top =0;
 int bottom = matrix.size()-1;
 int direction=0;

 while (left<=right && top<=bottom){
    // left → right
    if (direction==0){
      for (int col = left; col<=right ; col++){
          cout << matrix[top][col]<<" ";
      }
      top++;
    }
    // top → bottom
    else if (direction==1){
      for (int row = top; row<=bottom; row++){
          cout << matrix[row][right]<<" ";
      }
      right--;
    }
```

```
// right ⟶ left
else if (direction == 2) {
    for (int col = right; col >= left; col--) {
        cout << matrix[bottom][col] << " ";
    }
    bottom--;
}
// bottom ⟶ top
else {
    for (int row = bottom; row >= top; row--) {
        cout << matrix[row][left] << " ";
    }
    left++;
}
direction = (direction + 1) % 4;
}
}
int main() {
    int n, m;
    cin >> n >> m;
    Vector<Vector<int>> matrix(n, vector<int>(m));

    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            cin >> matrix[i][j];
        }
    }
    SpiralOrder(matrix);
    return 0;
}
```

l
r
t
b

while (l <= r && t <= b)

① l → r } t++

② top to bottom
r--

③ r to l } b++

④ b to top
l++

direction = (dir + 1) % 4

FREEMIND

Date

Page

Q2> Given a positive integer n, generate an nxn matrix filled with elements from 1 to n^2 in spiral order

1 → 2 → 3
↑ 8 → 9 4 ↓
7 ← 6 ← 5 ↓

Sol>

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

Vector <Vector <int>> createSpiralMatrix (int n) {

Vector <Vector<int>> matrix (n, Vector<int>(n));

int left = 0;
int right = n-1;
int top = 0;
int bottom = n-1;
int direction=0;
int value =1;

while (left <= right && top <= bottom) {

    // left to right
    if (direction ==0) {
      for (int i = left; i <= right; i++) {
          matrix [top][i] = value ++;
      } top++;
    }
    // top to bottom
  else if (direction ==1) {
        for (int j = top; j <= bottom; j++) {
matrix [j] [right] = value ++;
    } right --
  }
```

```
// right to left
else if (direction == 2) {
    for (int i = right; i >= left; i--) {
        matrix [bottom] [i] = value++;
    } bottom--
}
// bottom to top
    else {
        for (int j = bottom; j >= top; j--) {
            matrix [left] [j] = value++;
        } left++

    direction = (direction + 1) % 4;

    return matrix;
}

int main () {
    int n;
    cin >> n;
    Vector < Vector < int > > matrix (n, Vector <int> (n));

    matrix = createSpiralMatrix (n)

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

    return 0;
}
```

FREEMIND

Date

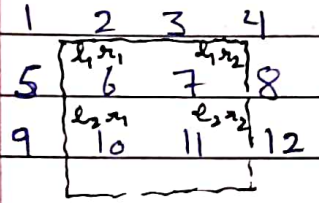
Page

Lecture-21 (Problems on 2D Arrays -5)

⭐ Pattern :- Prefix Sums in 2D Arrays

Q> Given a matrix 'a' of dimension n x m and 2 coordinates (l_1, r_1) and (l_2, r_2). Return the sum of the rectangle from (l_1, r_1) to (l_2, r_2).

⇒ Method 1: Brute Force



$l_1 \le l_2$

$r_1 \le r_2$

$i \longrightarrow l_1$ to l_2

$j \longrightarrow r_1$ to r_2

sum += array[i][j]

Sol:-

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

int rectangleSum(vector<vector<int>> &matrix, int l1, int l2,
                 int r1, int r2){
    int sum = 0;
    for (int i = l1; i <= l2; i++) {
        for (int j = r1; j <= r2; j++) {
            sum += matrix[i][j];
        }
    }
    return sum;
}
int main () {
    int n, m;
    cin >> n >> m;
```

```
Vector<Vector<int>> matrix(n, Vector<int>(m));

for (int i=0; i<n; i++) {
    for (int j=0; j<m; j++) {
        cin>>matrix[i][j];
    }
}
int l1, l2, r1, r2;
cin>>l1>>l2>>r1>>r2;

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

int sum = rectangleSum(matrix, l1, l2, r1, r2)
    cout<<"Sum: "<<endl;
    return 0;
}
```

Method 2:- Pre-calculating the horizontal sum for each row in the matrix.

```
int rectangleSum(Vector<Vector<int>>& matrix,
                 int l1, int l2, int r1, int r2){
    int sum=0
    for
    int sum = 0
    // prefix sum array Row-wise
    for (int i=0; i<matrix.size(); i++) {
        for (int j=0; j<matrix[0].size(); j++) {
    matrix[i][j] += matrix[i][j-1]
        }
    }
```

```
for (int i = l1; i <= l2; i++) {
    if (r1 != 0) {
        sum += matrix[i][r2] - matrix[i][r1-1];
    }
    else {
        sum += matrix[i][r2];
    }
}
return sum;

int main() {
    Same
}
```

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

Matrix (M-1)

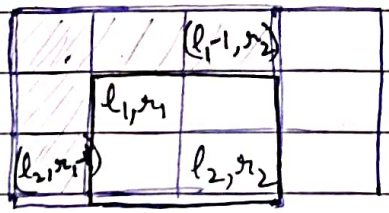
1	3	6	10
5	11	18	26
9	19	30	42

prefix Row wise (M-2)

1	3	6	10
6	14	24	36
15	33	54	78

both → Row column (M-3)

⭐ Method 3:- Prefix Sum over Columns and Rows both



Sum = arr[l2][r2]
- arr[l1-1][r2]
- arr[l2][r1-1]
+ arr[l1-1][r1-1]

{Kyuki 2 baar minus hora hai
Ek baar horizontal sum / ek baar vertical sum}

a[l][r] → rectangle sum from (0,0) to (l,r)

FREEMIND

Date

Page

Code → ~~Same~~ Almost Same

```
int rectangleSum( ---- )
    int sum = 0
    // prefix Sum array row wise
        ~~ Same ~~

    // prefix Sum array column wise
    for ( int i = 0 ; i < matrix.size() ; i++) {
        for( int j = 0 ; j < matrix[0].size() ; j++) {
            matrix [i][j] += matrix [i-1] [j] ;
        }
    }
// prefix Sum 2D array (print)

    for (int i = 0; i < matrix.size() ; i++) {
        for( int j = 0 ; i < matrix[0].size() ; j++) {
            cout << matrix [i] [j];
        } cout << endl ;
    }

    int top_sum = 0, left_sum = 0, topleft_sum = 0;
    int
    if (l1 != 0)  top_sum = matrix [l1-1] [r2] ;
    if (r1 != 0)  left_sum = matrix [l2] [r1-1] ;
    if (l1 != 0 && r1 != 0)  topleft_sum = matrix [l1-1] [r1-1];

    sum = matrix [l2] [r2] - top_sum - left_sum + topleft_sum
    return sum;
};
int main () {
    ~~ Same ~~
}
```