ARRAYS-LEVEL 3

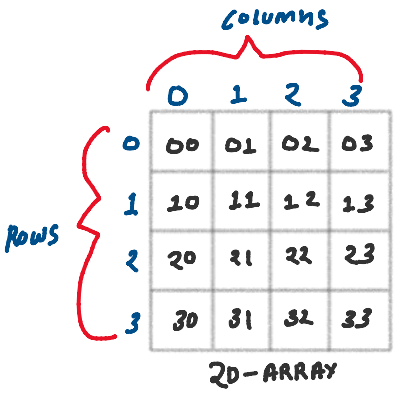
12 September 2023 09:41

CLASS 09 — 11/09/2023

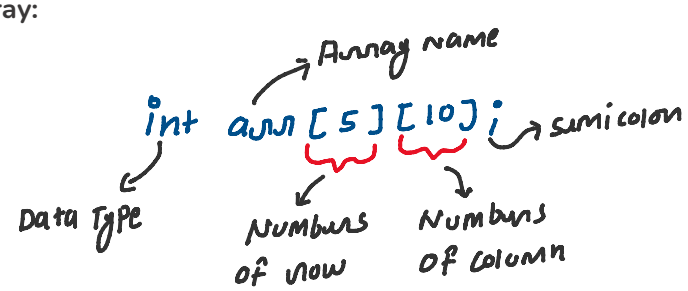
https://www.linkedin.com/in/manojoffcialmj/

ARRAY LEVEL-3

2D Array:



Creation of 2D Array:

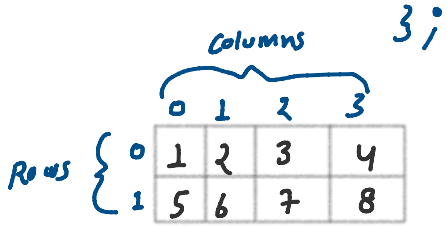


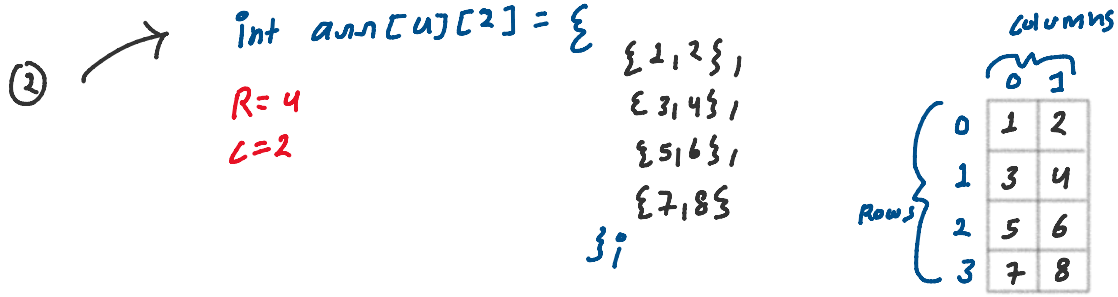
Initialization of 2D Array:

① int arr[2][4] = {

R=2
C=4

{1,2,3,4},
{5,6,7,8}

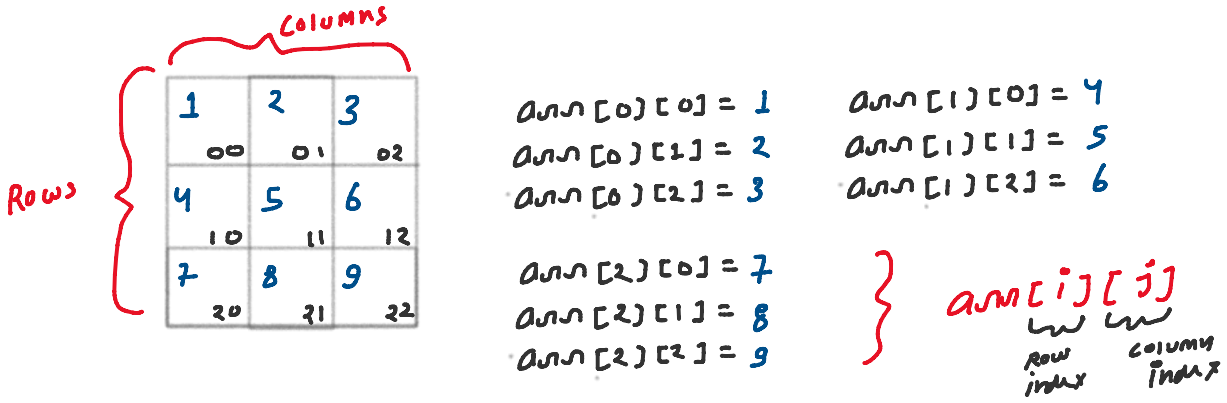




Note: Jab hum ek 2D array ko initialize karate hai ya fir use ek function me pass karate hai to hame uska column size hamesha dena likhna chaiye

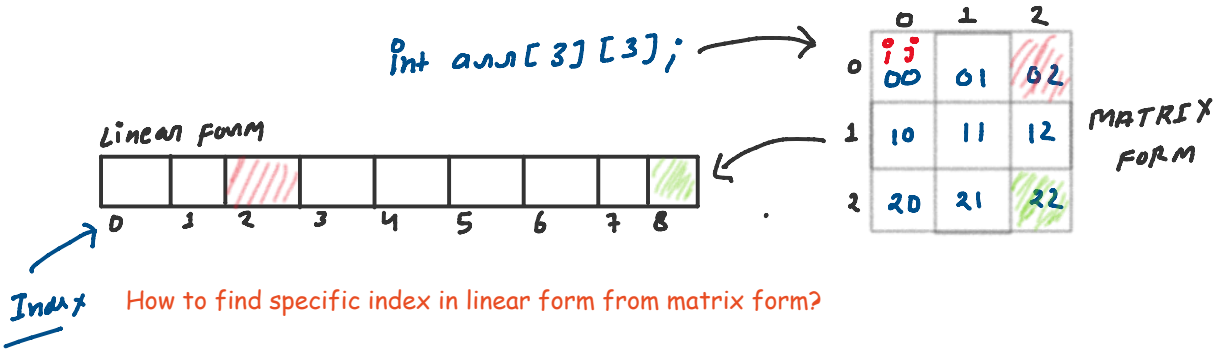
int arr[ ][4] ✓
int arr[2][ ] ✗ ERROR

Access 2D Array's element:



How 2D Array stored in memory:

It is stored in the form of linear data structure in the memory



How to find specific index in linear form from matrix form?

Index How to find specific index in linear form from matrix form?

Using this formula

↳ Number of columns * i + j

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

② arr[2][2] = 8
↳ 3 * 2 + 2 = 6 + 2 = 8

Print 2D Array row wise:

```cpp
// Print 2D-Array row wise
void printArray1(int arr[][4],int row,int col){
    for(int i=0;i<row;i++){
        for(int j=0;j<col;j++){
            cout<<arr[i][j]<<" ";
        }
        cout<<endl;
    }
}
```

@manojofficialmj

o/p

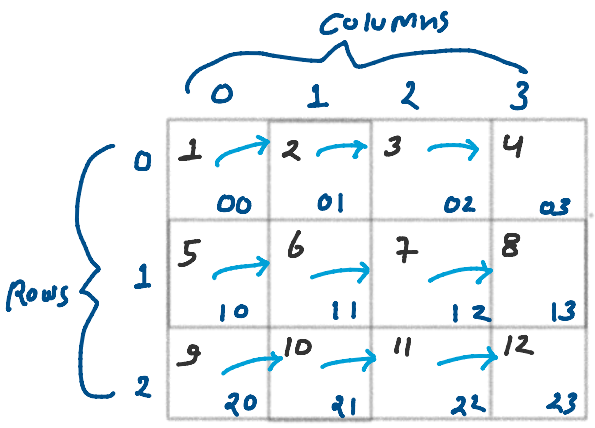
1 2 3 4
5 6 7 8
9 10 11 12

DRY RUN

i	j	arr[i][j]
0	0	1
	1	2
	2	3
	3	4
1	0	5
	1	6
	2	7
	3	8
2	0	9
	1	10
	2	11
	3	12
3		

↳ END

int arr[3][4];



Print 2D Array column wise:

```cpp
// Print 2D-Array column wise
void printArray2(int arr[][4],int row,int col){
    for(int i=0;i<col;i++){
        for(int j=0;j<row;j++){
            cout<<arr[j][i]<<" ";
        }
        cout<<endl;
    }
}
```

@manojofficialmj

o/p

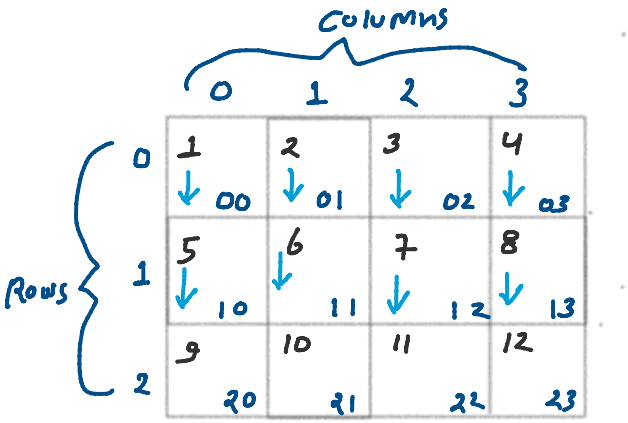
1 5 9
2 6 10
3 7 11
4 8 12

DRY RUN

i	j	arr[j][i]
0	0	1
	1	5
	2	9
1	0	2
	1	6
	2	10
2	0	3
	1	7
	2	11
3	0	4
	1	8
	2	12
4		

↳ END

int arr[3][4];

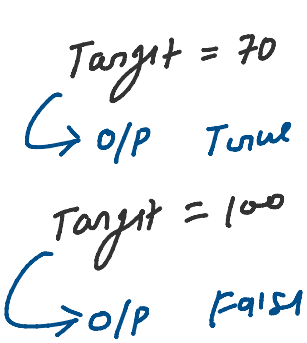


Taking input in 2D Array:

```cpp
int main(){
    // initialization of 2D array
    int arr[3][3];
    int row=3;
    int col=3;

    // Taking input from user
    for(int i=0;i<row;i++){
        for(int j=0;j<col;j++){
            cout<<"Row index = "<<i<<" Column index = "<<j<<endl;
            cin>>arr[i][j];
        }
        cout<<endl;
    }

    cout<<"Printing row wise"<<endl;
    printArray(arr,row,col);

    return 0;
}
/*
OUTPUT:
Row index = 0 Column index = 0
1
Row index = 0 Column index = 1
2
Row index = 0 Column index = 2
3
Row index = 1 Column index = 0
4
Row index = 1 Column index = 1
5
Row index = 1 Column index = 2
6
Row index = 2 Column index = 0
7
Row index = 2 Column index = 1
8
Row index = 2 Column index = 2
9
Printing row wise
1 2 3
4 5 6
7 8 9
*/
```

@manojofficialmj

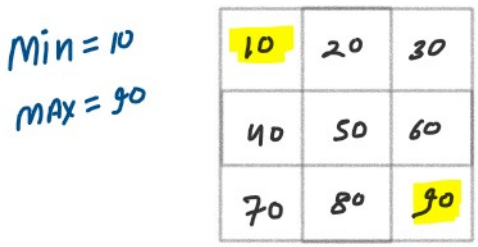
Linear search in 2D Array:



```cpp
//Program 03: Linear Search in 2D-Array
#include<iostream>
using namespace std;

// Linear Search in 2D-Array
bool findTarget(int arr[][3],int row,int col,int target){
    bool flag = false;
    for(int i=0;i<row;i++){
        for(int j=0;j<col;j++){
            if(arr[i][j]==target){
                flag = true;
            }
        }
    }
    return flag;
}

int main(){
    // initialization of 2D array
    int arr[3][3]={
        {10,20,30},
        {40,50,60},
        {70,80,90}
    };
    int row=3;
    int col=3;
    int target=100;

    bool flagAns=findTarget(arr,row,col,target);
    if(flagAns){
        cout<<"Target found";
    }else{
        cout<<"Target not found";
    }

    return 0;
}

/*
OUTPUT: when target=70
Target found

OUTPUT: when target=100
Target not found
*/
```

@manojofficialmj

Find min and max in 2D Array:




```cpp
// Find min
void findMin(int arr[][3],int row,int col){
    int min=INT_MAX;
    for(int i=0;i<row;i++){
        for(int j=0;j<col;j++){
            if(arr[i][j]<min){
                min=arr[i][j];
            }
        }
    }
    cout<<"Minimum is : "<<min<<endl; // 10
}
```

Min = 10

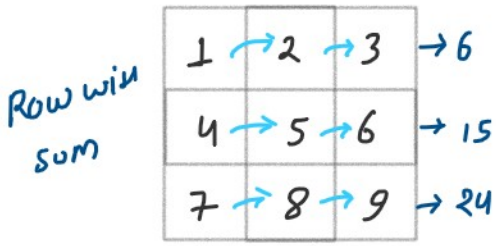
@manojofficialmj

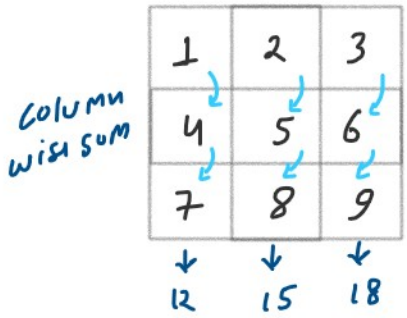
```cpp
// Find max
void findMax(int arr[][3],int row,int col){
    int max=INT_MIN;
    for(int i=0;i<row;i++){
        for(int j=0;j<col;j++){
            if(arr[i][j]>max){
                max=arr[i][j];
            }
        }
    }
    cout<<"Maximum is : "<<max<<endl; // 90
}
```

Max = 90

@manojofficialmj

Print row and col wise sum of 2D Array:






```cpp
// Print row wise sum
void printRowWiseSum(int arr[][3],int row,int col){

    for(int i=0;i<row;i++){
        int sum=0;
        for(int j=0;j<col;j++){
            sum+=arr[i][j];
        }
        cout<<sum<<endl;
    }
}
```

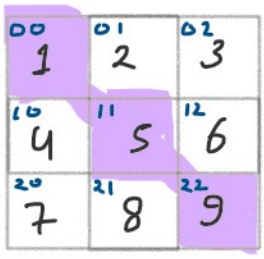
@manojofficialmj

```cpp
// Print column wise sum
void printCulumnWiseSum(int arr[][3],int row,int col){

    for(int i=0;i<col;i++){
        int sum=0;
        for(int j=0;j<row;j++){
            sum+=arr[j][i];
        }
        cout<<sum<<endl;
    }
}
```

@manojofficialmj

Print sum of diagonal of a matrix:

Row=3
Col=3



Principal Diagonal } 1+5+9 = 15

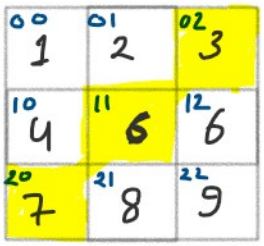
```cpp
// Sum of principal diagonal elements of a matrix
void principalDiagonalSum(int arr[][3],int row,int col){
    int sum=0;

    for(int i=0;i<row;i++){
        for(int j=0;j<col;j++){
            if(i==j){
                sum+=arr[i][j];
            }
        }
    }
    cout<<sum<<endl; // 15
}
```

@manojofficialmj

Logic: if(i==j)
↳ sum+=arr[i][j];

```cpp
// Sum of secondary diagonal elements of a matrix
void secondaryDiagonalSum(int arr[][3],int row,int col){
    int sum=0;

    for(int i=0;i<row;i++){
        for(int j=0;j<col;j++){
            if(i+j==(col-1)){
                sum+=arr[i][j];
            }
        }
    }
    cout<<sum<<endl; // 16
}
```

@manojofficialmj

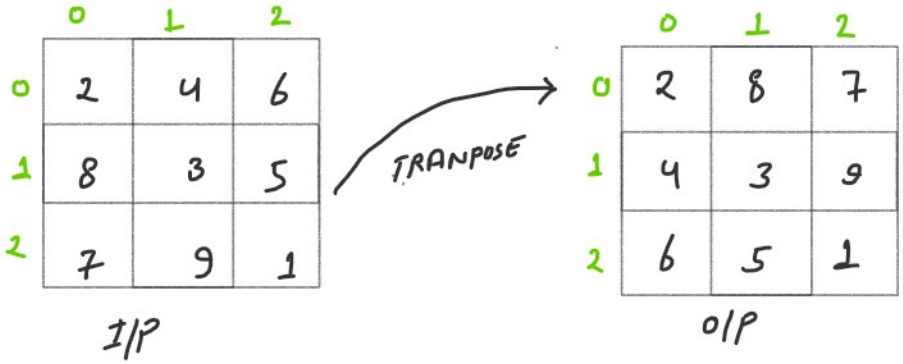


Secondary Diagonal { 3+6+7 = 16

Logic if (i+j = (col-1))

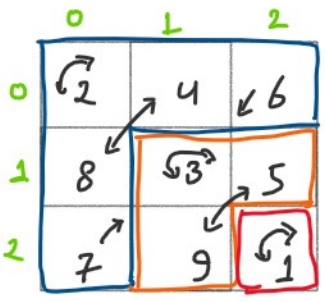
↳ sum += arr[i][j]

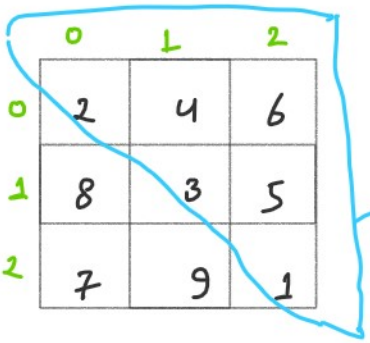
Transpose of a matrix:



① swaping (arr[i][j], arr[j][i])

WAY:1





ISS TARIKE SE TRAVERSE KARNA HAI

```cpp
for(int i=0; i<row; i++){
    for(j=i; j<col; j++){
        swap(arr[i], arr[j]);
    }
}
```

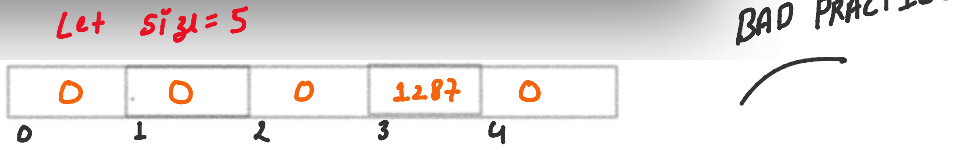
```cpp
// Way 01: Transpose of a matrix
void transposeMatrix1(int arr[][3],int row,int col){

    for(int i=0;i<row;i++){
        for(int j=i;j<col;j++){
            swap(arr[i][j],arr[j][i]);
        }
    }
}
```

@manojofficialmj

Vector STL C++:

```cpp
// Way 00: Static memory allocation with static array
int arr1[5]={1,2,3,4,5};
```


```cpp
// Way 01: Bad Practice with Dynamic memory allocation with user input
int size;
cin>>size;
int *arr2 = new int[size]; // Each element would be 0 or garbage value
```



Way 02:

Always Good Practice with Dynamic array (vector STL C++)

- *In vector, don't tell size of vector array.*
- *just keep inserting, we will manage the allocation*

Lesson 01: Declare 1D array

```cpp
// Lesson 01: Declare 1D array
vector<int> arr3;
```

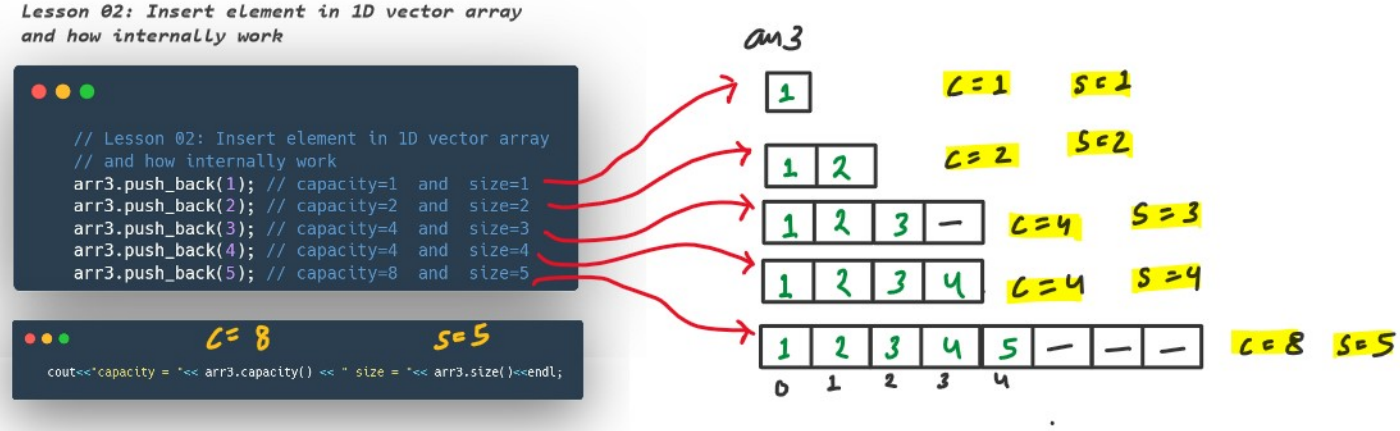
This is a Dynamic Array

Lesson 02: Insert element in 1D vector array and how internally work

```cpp
// Lesson 02: Insert element in 1D vector array
// and how internally work
arr3.push_back(1); // capacity=1  and  size=1
arr3.push_back(2); // capacity=2  and  size=2
arr3.push_back(3); // capacity=4  and  size=3
arr3.push_back(4); // capacity=4  and  size=4
arr3.push_back(5); // capacity=8  and  size=5
```

C= 8 S=5

```cpp
cout<<"capacity = "<< arr3.capacity() << " size = "<< arr3.size()<<endl;
```

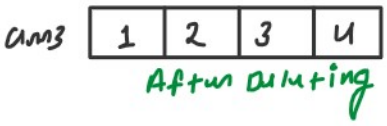


Lesson 03: Access 1D vector array's element

```cpp
// Lesson 03: Access 1D vector array's element
for(int i=0;i<arr3.size();i++){
    cout<<arr3[i]<<" "; // 1 2 3 4 5
}
```

Lesson 04: Delete element of 1D vector array

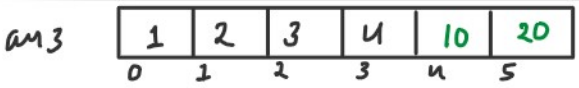
```cpp
// Lesson 04: Delete element of 1D vector array
arr3.pop_back(); // 1 2 3 4
```



Lesson 05: Taking input from user in 1D vector array

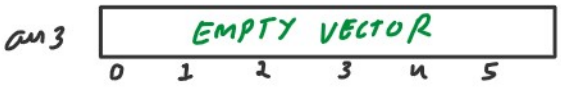
```cpp
// Lesson 05: Taking input from user in 1D vector array
int n;
cin>>n;
for(int i=0;i<n;i++){
    int data;
    cin>>data;
    arr3.push_back(data); // 10 , 20
}
```

Let n=2



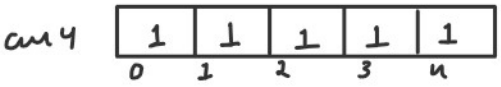
Lesson 06: We want to clear the 1D vector array

```cpp
// Lesson 06: We want to clear the 1D vector array
arr3.clear();
```



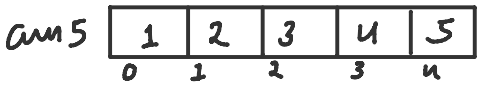
Lesson 07: We want to initialize 1D array vector with size and specific data

```cpp
// Lesson 07: We want to initialize 1D array vector with size and specific data
int size = 5;
int data = 1;
vector<int> arr4(size,data); // 1 1 1 1 1
```



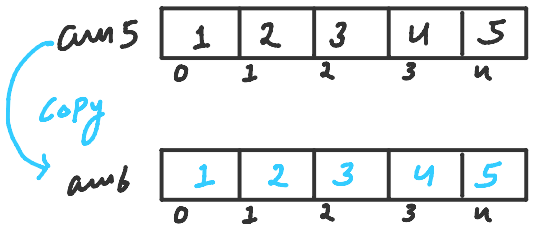
Lesson 08: We want to initialize 1D array vector with data only

```cpp
// Lesson 08: We want to initialize 1D array vector with data only
vector<int>arr5={1,2,3,4,5};
```



Lesson 09: We want to copy 1D array vector in other 1D array vector

```cpp
vector<int>arr5={1,2,3,4,5};

// Lesson 09: We want to copy 1D array vector in other 1D array vector
vector<int>arr6(arr5); // {1,2,3,4,5}
```



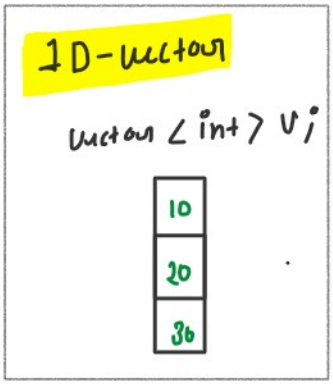
Lesson 10: We want to pass 1D vector array in a function to print it

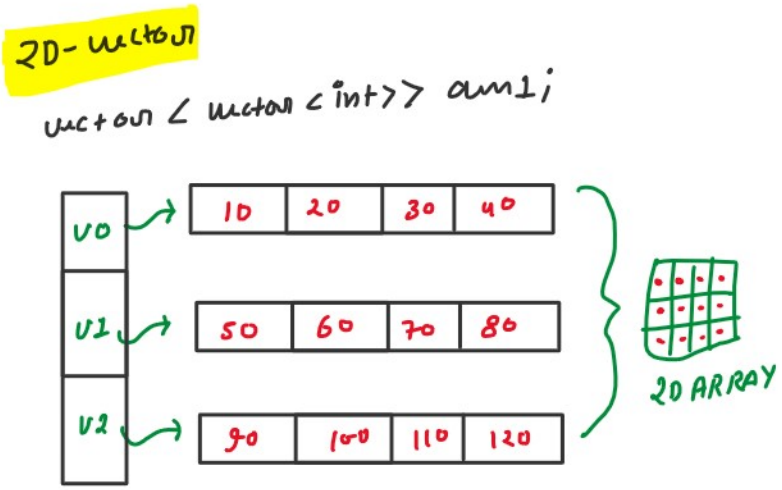
```cpp
#include<iostream>
#include<vector>
using namespace std;

// Function to print the 1D vector array
void printVector(vector<int> arr6){
    int size = arr6.size();
    for(int i=0;i<size;i++){
        cout<<arr6[i]<<" ";
    }
}


int main(){

    vector<int>arr6={1,2,3,4,5};

    // Lesson 10: We want to pass 1D vector array in a function to print it
    printVector(arr6);

    return 0;
}
```

2D VECTOR ARRAY

Lesson 01: Declare 2D array

```
// Lesson 01: Declare 2D array
vector<vector<int>> arr1;
```

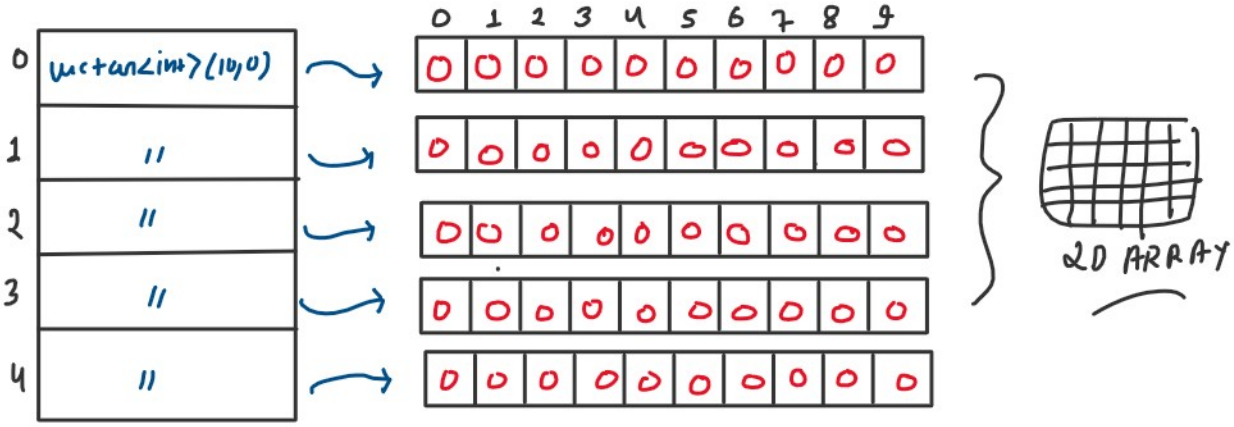




Lesson 02: Initialization with specific row size and column size that initiate with specific value

```
vector<vector<int>> arr2(5,vector<int>(10,0));
```

2D ARRAY — NAME — Row size — Row items initiate with vector of size 10 that is initialized with zero 0.



Lesson 03: Access elements of 2D vector array

```cpp
// Vector for 1D Array
#include<iostream>
#include<vector>
using namespace std;

int main(){

    vector<vector<int>> arr2(5,vector<int>(10,0));

    // Lesson 03: Access elements of 2D vector array
    int RowSize=arr2.size(); // 5
    int ColSize=arr2[0].size(); // 10

    for(int i=0;i<RowSize;i++){
        for(int j=0;j<ColSize;j++){
            cout<<arr2[i][j]<<" ";
        }
        cout<<endl;
    }

    return 0;
}

/*
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
*/
```

JAGGED ARRAY

```cpp
// Jagged array
#include <iostream>
#include<vector>
using namespace std;

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


  vector<int> vec1(10,0);
  vector<int> vec2(5,1);
  vector<int> vec3(7,0);
  vector<int> vec4(8,1);
  vector<int> vec5(20,0);

  arr.push_back(vec1);
  arr.push_back(vec2);
  arr.push_back(vec3);
  arr.push_back(vec4);
  arr.push_back(vec5);

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

  return 0;
}

/*
0 0 0 0 0 0 0 0 0 0
1 1 1 1 1
0 0 0 0 0 0 0
1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
*/
```

THANKS FOR Reading

Share to HIP each others