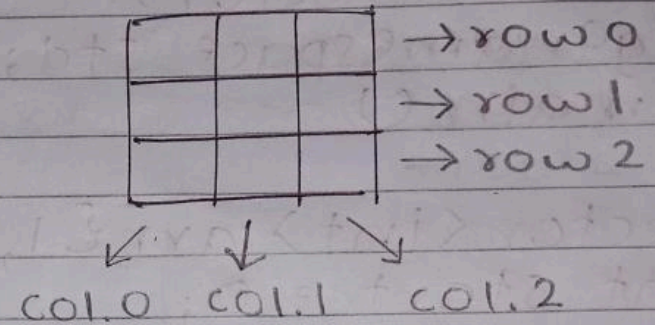
Arrays (class3)

12-02-2023

Date.....

* 2D Arrays

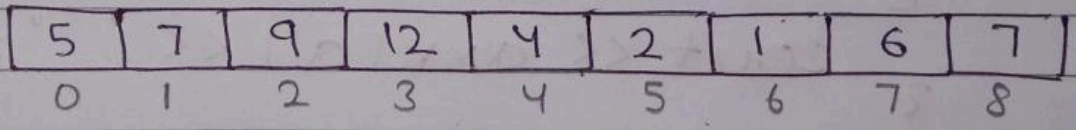



```
int arr[1000][1000];
```

- datatype
- name of array
- no. of rows
- no. of columns

	0	1	2
0	5	7	9
1	12	4	2
2	1	6	7

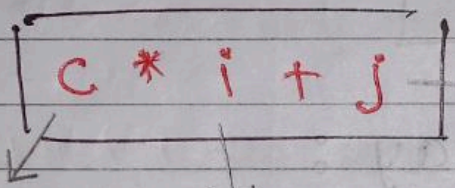
→ this is the visualization on 2D array but actually it is stored linearly in the memory.



Date.....

arr[1][2] means 2
arr[2][0] means 1
arr[0][2] means 9

How does arr[1][2] know that it should be placed on 5th index in the memory.
It is calculated by the formula, i.e.,



Total columns are 3, row no. is 1 & column no. is 2.
so,

=> 3 * 1 + 2
=> 3 + 2
=> 5 → this is the index no. in the linear array at which arr[1][2] is going to store.

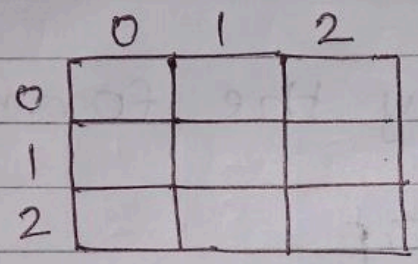
Date.....

* Declare 2D Array :

```
int arr[3][3];
```

no. of Rows

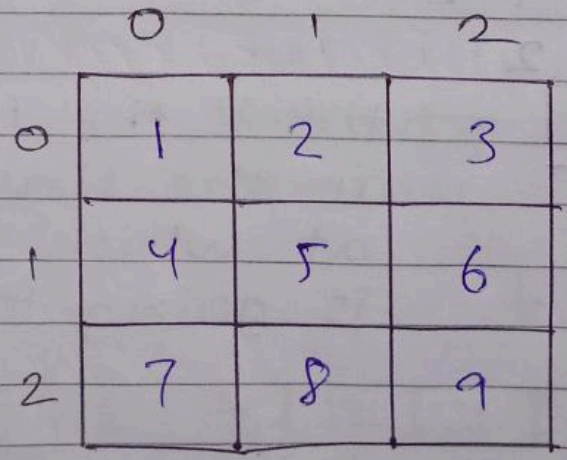
no. of columns



* Initialize 2D Array :

```
int arr[3][3] = {
                {1, 2, 3},
                {4, 5, 6},
                {7, 8, 9}
};
```

col.0 col.1 col.2



Date.....

* Accessing elements :-

```
cout << arr[1][2];   // 6
cout << arr[1][0];   // 4
cout << arr[0][2];   // 3
cout << arr[0][0];   // 1
cout << arr[2][1];   // 8
```

* Access 2D array row wise :-

```
//outer loop for holding one
  element at a time
for(int i = 0; i < 3; i++){
   cout << "Row no. " << i << ": ";
   //inner loop for printing
     every column corresponding
     to ith row.
   for(int j = 0; j < 3; j++){
       cout << arr[i][j] << " ";
   }
   cout << endl;
}
```

Date.....

Ques. Print row-wise sum in the matrix.

In this, we've given a matrix & we have to print the sum of each row.
for example :-

1	2	3	4	→ sum = 10
2	3	4	11	→ sum = 20
5	6	1	3	→ sum = 15
2	4	6	8	→ sum = 20
1	9	9	7	→ sum = 26

code :-

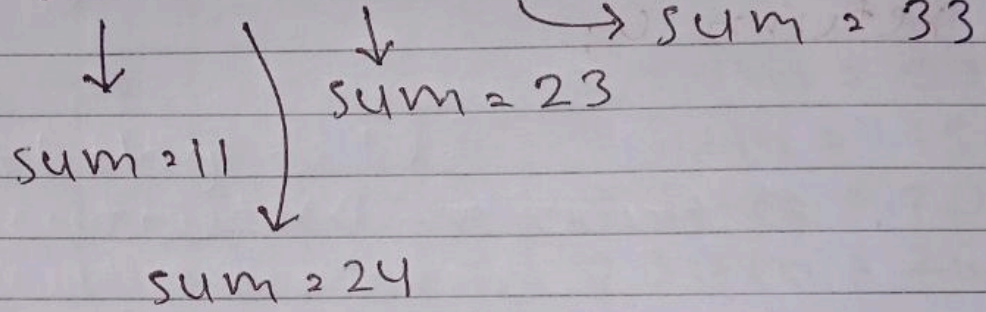
```
#include<iostream>
#include<vector>
using namespace std;
int main()
{
    int arr[5][4] = {
                        {1,2,3,4},
                        {2,3,4,11},
                        {5,6,1,3},
                        {2,4,6,8},
                        {1,9,9,7}
                    };
```

Date.....

```
for(int i = 0; i < 5; i++){
    int sum = 0;
    for(int j = 0; j < 4; j++){
        sum = sum + arr[i][j];
    }
    cout << "Row " << i << " sum
        is : " << sum;
    cout << endl;
}
return 0;
}
```

Que: Print column-wise sum in the matrix.

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



code :-

```
int main(){
    int arr[5][4] = {
                        {1,2,3,4},
                        {2,3,4,11},
                        {5,6,1,3},
                        {2,4,6,8},
                        {1,9,9,7}
                    };
int i,j;
for(i=0; i<4; i++){
    int sum = 0;
    for(j=0; j<5; j++){
        sum = sum + arr[j][i];
    }
    cout << sum;
}
}
```


Teacher's Sign .....

Date.....

Ques. Linearly search an element in 2D array.

code:-

```
#include <iostream>
using namespace std;
int main()
{
   int arr[5][4] = {
                    {10, 12, 13, 14},
                    {15, 16, 17, 18},
                    {19, 20, 21, 22},
                    {23, 24, 25, 26},
                    {27, 28, 29, 30}
                  };

   //print array
   cout << "current array: ";
   for(int i = 0; i < 5; i++)
   {
     for(int j = 0; j < 4; j++)
     {
        cout << arr[i][j] << " ";
     }
     cout << endl;
   }

   int n;
   cout << "\n Enter element: ";
   cin >> n;
```

Date.....

```
int flag = 0;
for(int i = 0; i < 5; i++){
    for(int j = 0; j < 4; j++){
        if(arr[i][j] == n){
            flag = 1;
            break;
        }
    }
}

if( flag == 1){
    cout << "In found.";
}
else {
    cout << "In Not Found.";
}

return 0;
}
```

Ques Maximum & Minimum element in an array.

code:-

```
int main()
{
    int arr[5][4] = {
                        {10, 12, 13, 14},
                        {15, 16, 17, 18},
                        {19, 80, 21, 22},
                        {23, 24, 95, 26},
                        {27, 28, 29, 30}
                    };

    int max = INT_MIN;
    for(int i = 0; i < 5; i++)
    {
        for(int j = 0; j < 4; j++)
        {
            if(arr[i][j] > max)
            {
                max = arr[i][j];
            }
        }
    }
}
```

Date.....

```
int min = INT_MAX;
for(int i = 0; i < 5; i++)
{
    for(int j = 0; j < 4; j++)
    {
        if(arr[i][j] < min)
        {
            min = arr[i][j];
        }
    }
}

cout << "minimum : " << min;
cout << "maximum : " << max;

return 0;
}
```

Date.....

Ques. Transpose a matrix.

Transpose means, Row becomes column and column becomes Row.

	0	1	2
0	10	12	13
1	15	16	17
2	19	80	21

	0	1	2
0	10	15	19
1	12	16	80
2	13	17	21

(before transpose)

(after transpose)

code:-

```
#include<iostream>
using namespace std;
int main()
{
   int arr[3][3] = {
                    {10,12,13},
                    {15,16,17},
                    {19,80,21}
                   };

   int brr[3][3];
```

```
cout << "current array:";
for(int i=0; i<3; i++)
{
    for(int j=0; j<3; j++)
    {
        cout << arr[i][j] << " ";
    }
    cout << endl;
}

for(int i=0; i<3; i++)
{
    for(int j=0; j<3; j++)
    {
        brr[j][i] = arr[i][j];
    }
}

cout << "array after transpos";
for(int i=0; i<3; i++)
{
    for(int j=0; j<3; j++)
    {
        cout << brr[i][j] << " ";
    }
    cout << endl;
}

return 0;
}
```

Date.....

Ques Declaring 2D vector.

code:-

```
#include<iostream>
#include<vector>
using namespace std;
int main()
{
    vector<vector<int>> arr;

    vector<int> a{1,2,3};
    vector<int> b{4,5,6};
    vector<int> c{7,8,9};

    arr.push_back(a);
    arr.push_back(b);
    arr.push_back(c);

    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;
}
```

2nd way of declaring vector of vector :-

no. of rows in outer vector

```
vector<vector<int>> arr(rows,
        vector<int>(col, 0));
```

- from which type of values we want to initialize our outer vector
- no. of columns in inner vector
- the value with which we want to initialize our inner vector

For example,

```
vector<vector<int>> arr(7, vector
        <int>(6, 0));
```

this will create this type of matrix internally

	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	0	0	0	0	0	0
6	0	0	0	0	0	0