Computer Programming (a)

E1123

Fall 2022-2023

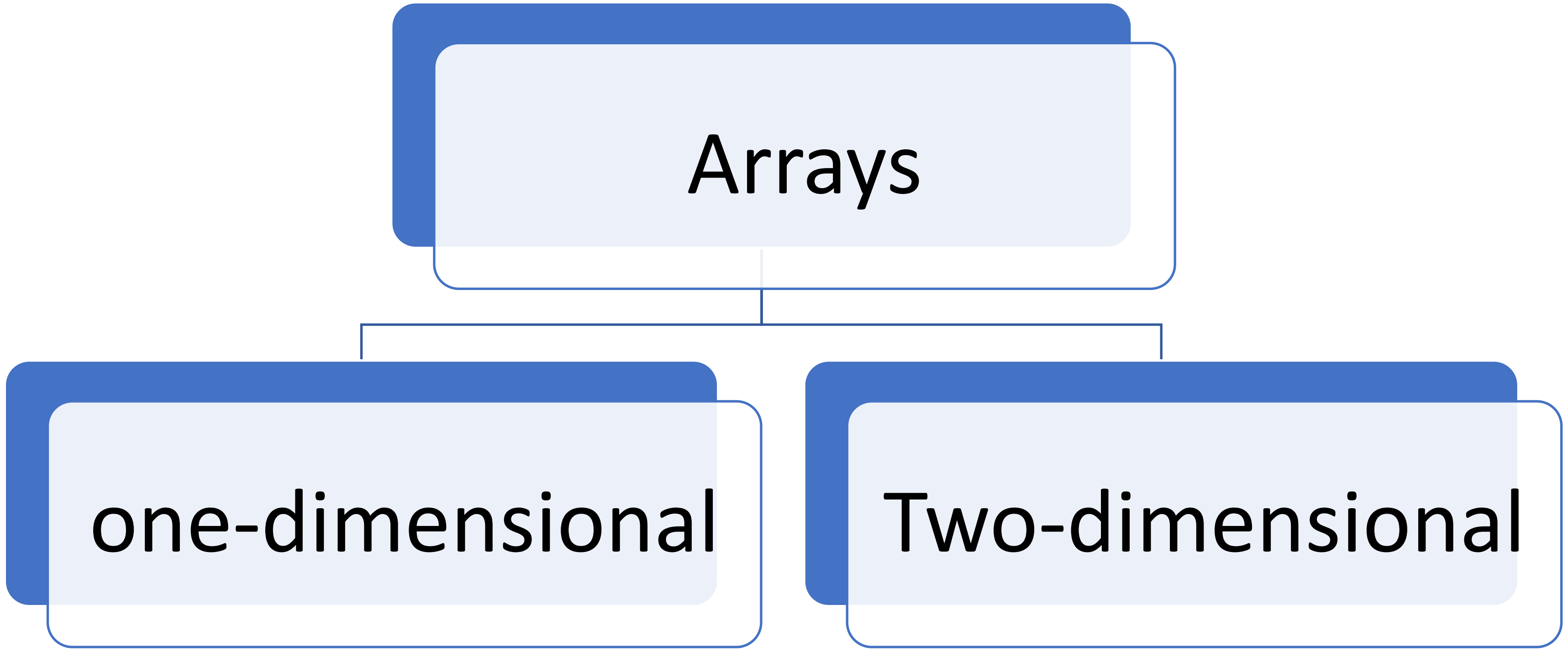
Lecture 12

Multidimensional Arrays

INSTRUCTOR

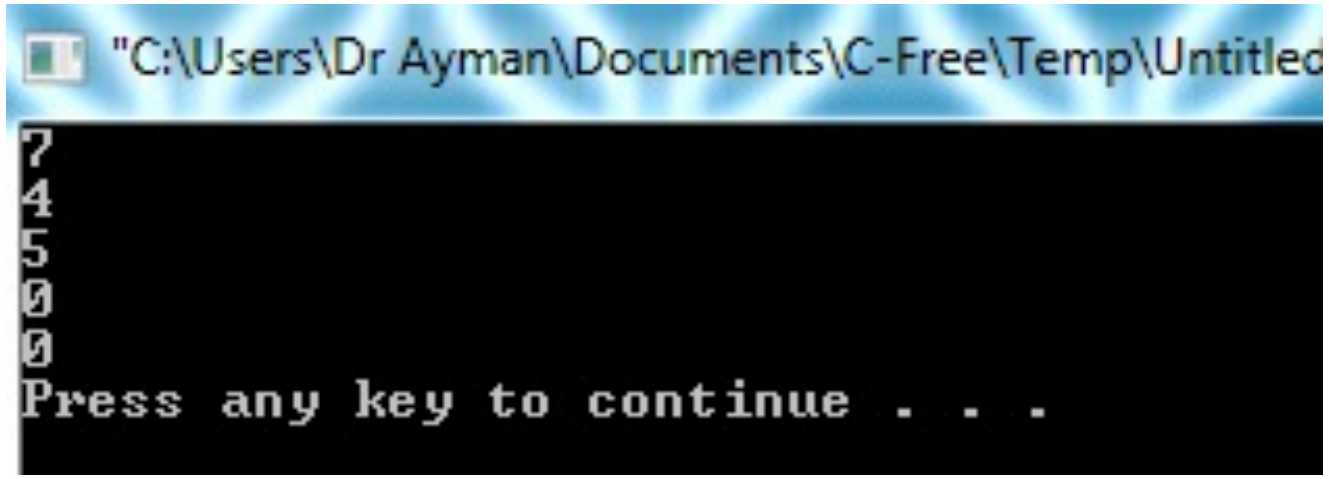
DR / AYMAN SOLIMAN

➢ Contents


Arrays
one-dimensional
Two-dimensional

➢ Introduction

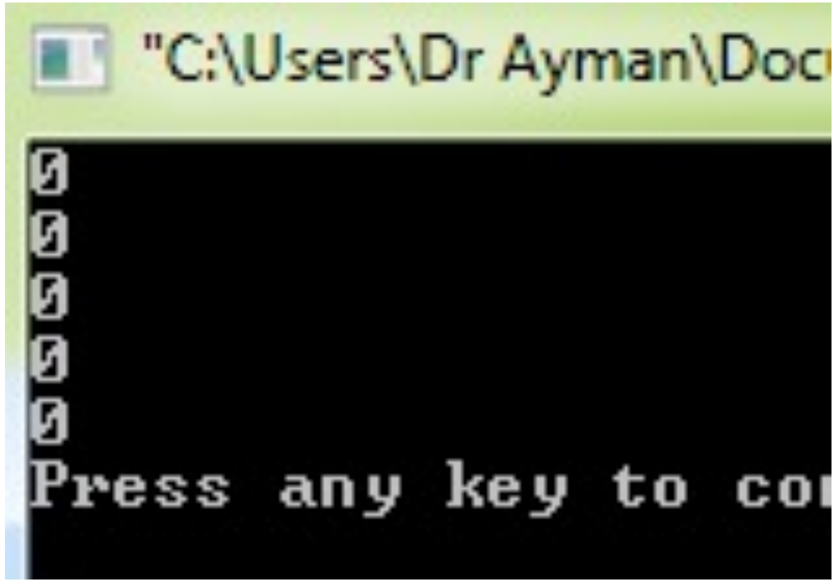
```
#include <iostream.h>
 int main()
{
   int array[5]={ 7, 4, 5 }; // only initialize first 3 elements
   cout << array[0] << '\n';
   cout << array[1] << '\n';
   cout << array[2] << '\n';
   cout << array[3] << '\n';
   cout << array[4] << '\n';
    return 0;
}
```



```
"C:\Users\Dr Ayman\Documents\C-Free\Temp\Untitled
7
4
5
0
0
Press any key to continue . . .
```

➢ Example 1

```
#include <iostream.h>
 int main()
{
   int array[5]={}; // only initialize all elements to zero
   cout << array[0] << '\n';
   cout << array[1] << '\n';
   cout << array[2] << '\n';
   cout << array[3] << '\n';
   cout << array[4] << '\n';
    return 0;
}
```


"C:\Users\Dr Ayman\Doc

```
0
0
0
0
0
Press any key to co
```

Multidimensional Arrays

➢ Multidimensional Arrays

❑ Arrays could be more than one dimension.

```
int array[3][5]; // declaration of a 3*5 element array
```

```
[0][0]  [0][1]  [0][2]  [0][3]  [0][4]  // row 0
[1][0]  [1][1]  [1][2]  [1][3]  [1][4]  // row 1
[2][0]  [2][1]  [2][2]  [2][3]  [2][4]  // row 2
```

❑ To access the elements of a two-dimensional array, simply use two subscripts:

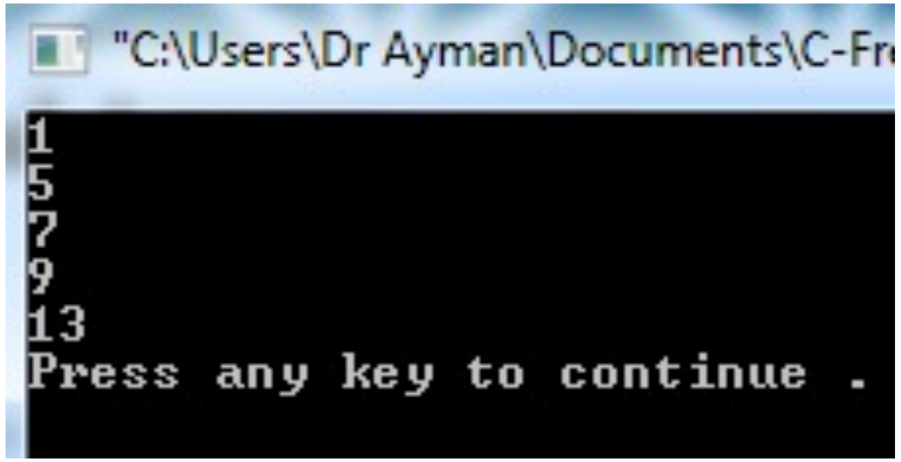
```
array[0][0] = 3;
array[0][1] = 30;
array[2][3] = 7;
```

➢ Multidimensional Arrays (cont.)

❑ Example 2

```
#include <iostream.h>
int main()
{ int array[3][5] =
{{ 1, 2, 3, 4, 5 },          // row 0
{ 6, 7, 8, 9, 10 },          // row 1
{ 11, 12, 13, 14, 15 }    // row 2
};
return 0;}
```

```
cout << array[0][0] << '\n';
cout << array[0][4] << '\n';
cout << array[1][1]<< '\n';
cout << array[1][3] << '\n';
cout << array[2][2] << '\n';
```



"C:\Users\Dr Ayman\Documents\C-Fr

```
1
5
7
9
13
Press any key to continue .
```

➢ Multidimensional Arrays (cont.)

❑ Initializing two-dimensional arrays

//Acceptable

```
int array[3][5] =
{
{ 1, 2, 3, 4, 5 },           // row 0
{ 6, 7, 8, 9, 10 },          // row 1
{ 11, 12, 13, 14, 15 }       // row 2
};
```

//Acceptable

```
int array[3][5] =
{
{ 1, 2 },                  // row 0 = 1, 2, 0, 0, 0
{ 6, 7, 8 },               // row 1 = 6, 7, 8, 0, 0
{ 11, 12, 13, 14 }  // row 2 = 11, 12, 13, 14, 0
};
```

```
int array[3][5] = { }; //Acceptable
```

➢ Multidimensional Arrays (cont.)

- Initializing two-dimensional arrays

```
int array[ ][5] =
{
{ 1, 2, 3, 4, 5 },       // row 0
{ 6, 7, 8, 9, 10 },      // row 1
{ 11, 12, 13, 14, 15 } // row 2
};
```

//Acceptable

//Compiler Error

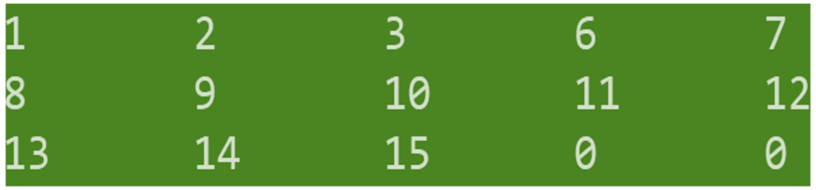
```
int array[ ][ ] =
{
{ 1, 2, 3, 4, 5 },       // row 0
{ 6, 7, 8, 9, 10 },      // row 1
{ 11, 12, 13, 14, 15 } // row 2
};
```

//Compiler Error

```
int array[3][ ] = {{ 1, 2, 3, 4, 5 },       // row 0
                   { 6, 7, 8, 9, 10 },      // row 1
                   { 11, 12, 13, 14, 15 }  // row 2

};
```

➢ Example 3

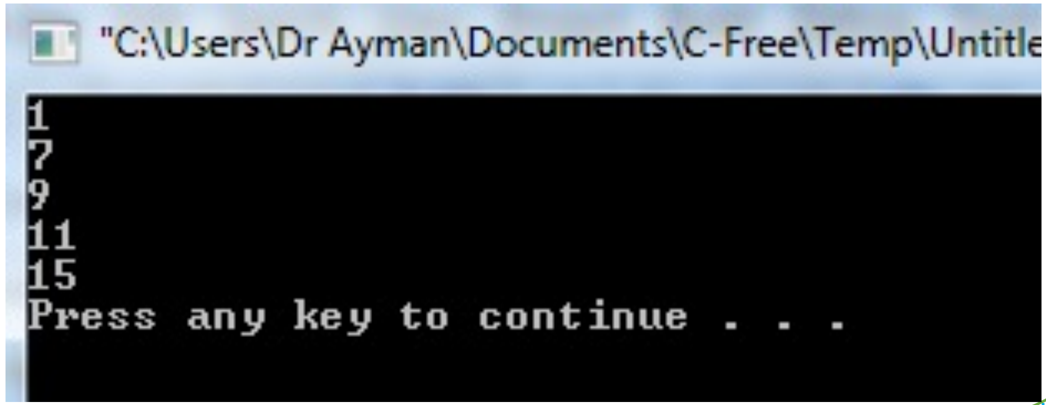
```
#include <iostream.h>
int main()
{
int array[3][5] =
{
1, 2, 3,
6, 7, 8, 9, 10 ,
11, 12, 13, 14, 15
};
```



```
1       2       3       6       7
8       9       10      11      12
13      14      15      0       0
```

//Acceptable

```
cout << array[0][0] << '\n';
cout << array[0][4] << '\n';
cout << array[1][1]<< '\n';
cout << array[1][3] << '\n';
cout << array[2][2] << '\n';
return 0;}
```



"C:\Users\Dr Ayman\Documents\C-Free\Temp\Untitle

```
1
7
9
11
15
Press any key to continue . . .
```

➢ Declaration and assignment of a 3D array

```
int array[2][2][3];

array[0][0][0] = 2;

array[0][0][1] = 3; ……etc
```

➢ Initialize all elements to 0

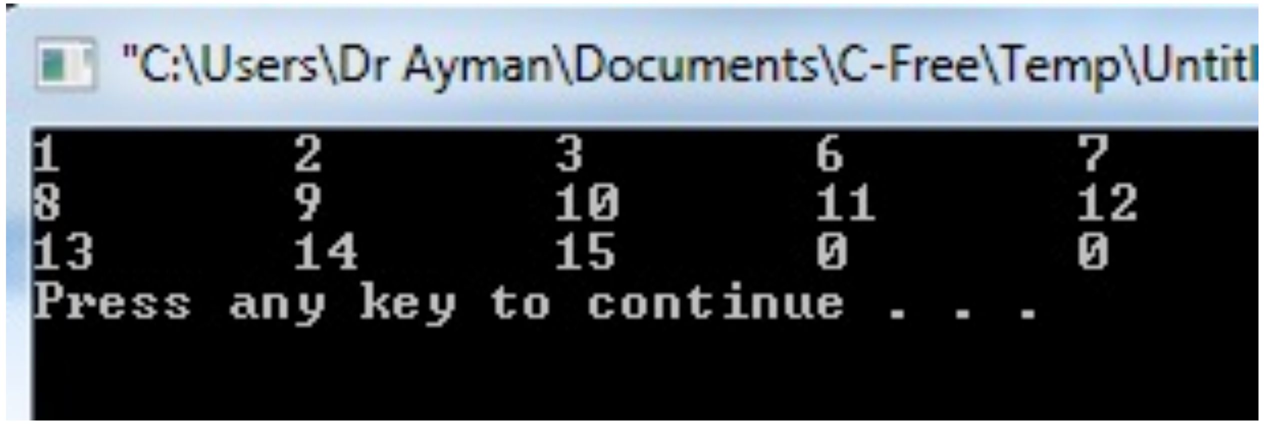
```
int array[2][3][7] ={};
```

➢ Declaration and assignment of a 4D array

```
int array[2][2][3][7];
```

➢ Printing all elements of array

```
#include <iostream.h>
int main()
{
int array[3][5] =
{1, 2, 3, 6, 7, 8, 9, 10 , 11, 12, 13, 14, 15};
    for ( int x=0;x<3;x++)
   {for (int y=0; y<5;y++)
    cout<<array[x][y]<<'\t';
    cout<<endl;}
    return 0;}
```

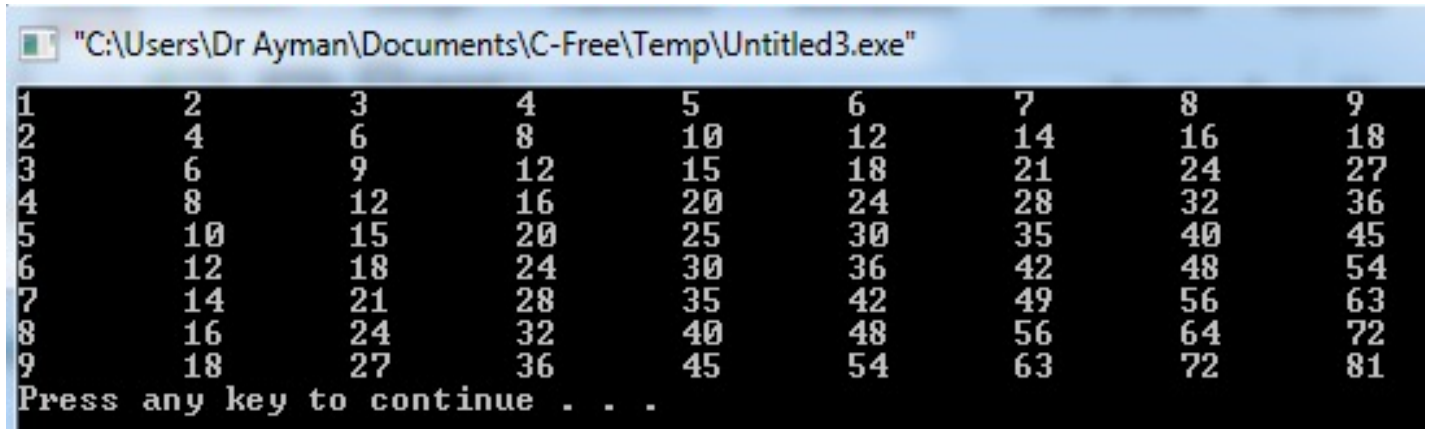


"C:\Users\Dr Ayman\Documents\C-Free\Temp\Untitl

```
1       2       3       6       7
8       9       10      11      12
13      14      15      0       0
Press any key to continue . . .
```

➢ Example 4 (multiplication table)

```
#include <iostream.h>
int main()
{  int multiplication[9][9]={};
   for (int x=0;x<9;x++)
   for (int y=0; y<9;y++)
   multiplication[x][y]=(x+1)*(y+1);
   // print the table:-
    for (int x=0;x<9;x++)
   {for (int y=0; y<9;y++)
   cout<<multiplication[x][y]<<'\t';
   cout<<endl;}
   return 0;}
```

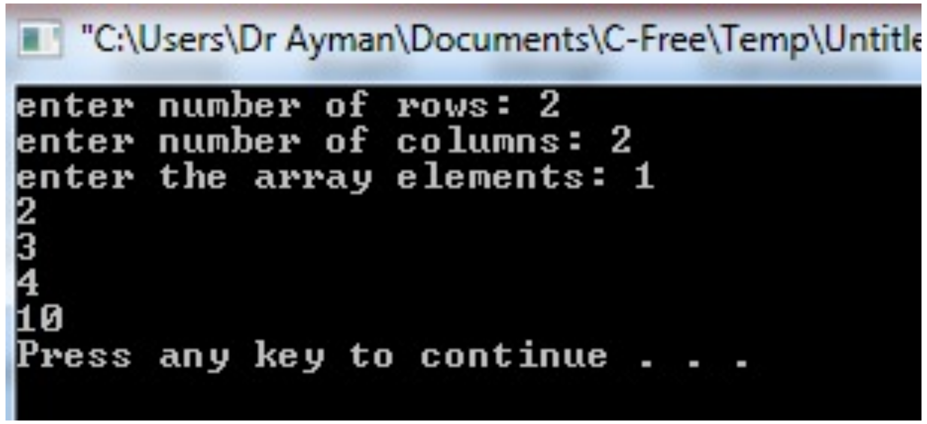


"C:\Users\Dr Ayman\Documents\C-Free\Temp\Untitled3.exe"

```
1       2       3       4       5       6       7       8       9
2       4       6       8       10      12      14      16      18
3       6       9       12      15      18      21      24      27
4       8       12      16      20      24      28      32      36
5       10      15      20      25      30      35      40      45
6       12      18      24      30      36      42      48      54
7       14      21      28      35      42      49      56      63
8       16      24      32      40      48      56      64      72
9       18      27      36      45      54      63      72      81
Press any key to continue . . .
```

➢ Example 5 Find the summation of all array elements

```
#include <iostream.h>
int main()
{int i,j,sum=0;
cout<<"enter number of rows: ";
cin>>i;
cout<<"enter number of columns: ";
cin>>j;
cout<<"enter the array elements: ";
int array[i][j];
    for (int x=0;x<i;x++)
    for (int y=0; y<j;y++)
    cin>>array[x][y];
// find the summation of all elements
    for (int x=0;x<i;x++)
    for (int y=0; y<j;y++)
    sum=sum+array[x][y];
    // print the summation
    cout<<sum<<endl;
    return 0;}
```


"C:\Users\Dr Ayman\Documents\C-Free\Temp\Untitle

```
enter number of rows: 2
enter number of columns: 2
enter the array elements: 1
2
3
4
10
Press any key to continue . . .
```

➢ Passing Array to a Function in C++ Programming

- ❑ In C++, we can pass arrays as an argument to a function. and, also we can return arrays from a function.

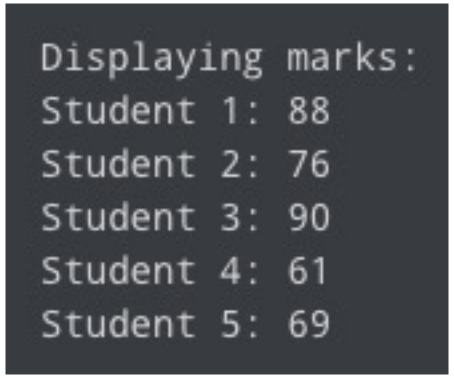
- ❑ The syntax for passing an array to a function is:

```
returnType functionName(dataType arrayName[arraySize])
{

   // code

}
```

➢ Example (C++ Program to display marks of 5 students)

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

// declare function to display marks
// take a 1d array as parameter
void display(int m[5]) {
    cout << "Displaying marks: " << endl;

    // display array elements
    for (int i = 0; i < 5; ++i) {
        cout << "Student " << i + 1 << ": " << m[i] << endl;
    }
}
```

```
int main() {

    // declare and initialize an array
    int marks[5] = {88, 76, 90, 61, 69};

    // call display function
    // pass array as argument
    display(marks);

    return 0;
}
```

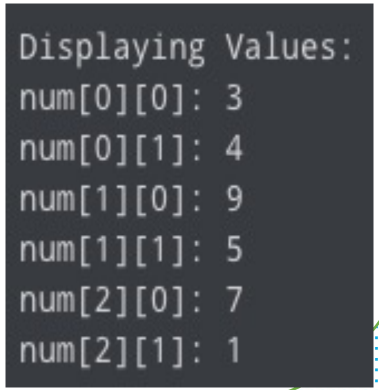


```
Displaying marks:
Student 1: 88
Student 2: 76
Student 3: 90
Student 4: 61
Student 5: 69
```

➢ Example (two-dimensional array by passing it to a function)

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

// define a function
// pass a 2d array as a parameter
void display(int n[3][2]) {
    cout << "Displaying Values: " << endl;
    for (int i = 0; i < 3; ++i) {
        for (int j = 0; j < 2; ++j) {
            cout << "num[" << i << "][" << j << "]: " << n[i][j] << endl;
        }
    }
}
```

```
int main() {

    // initialize 2d array
    int num[3][2] = {
        {3, 4},
        {9, 5},
        {7, 1}
    };

    // call the function
    // pass a 2d array as an argument
    display(num);

    return 0;
}
```



```
Displaying Values:
num[0][0]: 3
num[0][1]: 4
num[1][0]: 9
num[1][1]: 5
num[2][0]: 7
num[2][1]: 1
```

Quiz

➢ Quiz

Write a C++ program to use 2D arrays to perform matrix addition and print the result.

```
#include<iostream>
using namespace std;
main()
{
    int  m1[5][5], m2[5][5], m3[5][5];
    int  i, j, r, c;
    cout<<"Enter the no.of rows of the matrices to be added(max 5):";
    cin>>r;
    cout<<"Enter the no.of columns of the matrices to be added(max 5):";
    cin>>c;
    cout<<"\n1st Matrix Input:\n";
   for(i=0;i<r;i++)
   {for(j=0;j<c;j++)
              {cout<<"\nmatrix1["<<i<<"]["<<j<<"]=  ";
               cin>>m1[i][j];}}
cout<<"\n2nd Matrix Input:\n";
   for(i=0;i<r;i++){
              for(j=0;j<c;j++){
                        cout<<"\nmatrix2["<<i<<"]["<<j<<"]=  ";
                        cin>>m2[i][j];}}
   cout<<"\nAdding Matrices...\n";
   for(i=0;i<r;i++)
   {for(j=0;j<c;j++)
              {m3[i][j]=m1[i][j]+m2[i][j];}}
    cout<<"\nThe resultant Matrix is:\n";
   for(i=0;i<r;i++)
   {for(j=0;j<c;j++)
              {cout<<"\t"<<m3[i][j];}
              cout<<endl;}}
```

C++
Projects
</>
HTML
C++

➢ Project 1 - Car Rental System

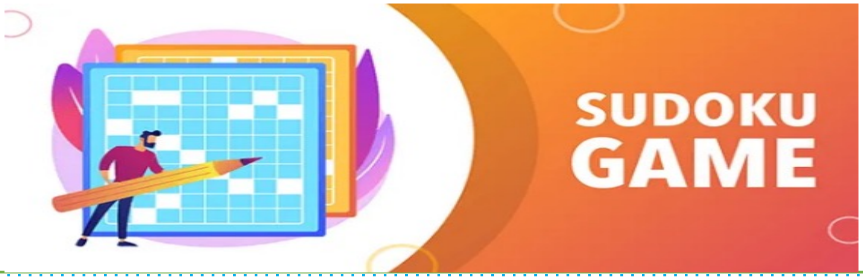
- ➢ This is a trendy project and very useful for learning about keyboard events, date-time functions, and implementing a C++ login system. The program has separate menus for admin and other users. There are also methods to calculate fare based on time and distance, including displaying car details, availability, etc.

➢ Project 2 - Bookshop inventory system

- ➢ This is a simple project where the system maintains the inventory of books in a bookshop. If a customer purchases a book, the book's count will decrease; if a book is added, the same is updated. You can modify the code to add a book ID and make the search based on book ID or make the search using just one parameter giving multiple results, and so on.

➢ Project 3 - Sudoku Game

- ➢ We all know about the popular Sudoku game, wherein we need to arrange numbers from 1-9 such that they appear only once in a row and column of a 9x9 grid. The program uses the concept of backtracking. In this program, we have hard-coded the initial values, but you can also get the same input from the user (though that will be cumbersome for this program). The main thing to understand is the backtracking to find rows and columns that are not assigned any values (are zero). Have a look at the program, execute, and see the results:



➢ Some projects

1. Attendance Management System.
2. Bus Reservation System.
3. College Registration System.
4. Doctor appointment System.
5. Hotel Management System.
6. Staff Management System.
7. Student Database Management System.

Thank you