RED & WHITE®

Multimedia Education

Shaping "skills" for "scaling" higher...!!!

WELCOME,

PROGRAMMERS

Let’s see **2D Array** in detail with some examples...

2D ARRAY

A **2D array** (two-dimensional array) is a data structure that represents a **table** or a **matrix** with **rows and columns**.

It is essentially an **array of 1D arrays**, where each element of the main array is a 1D array itself.

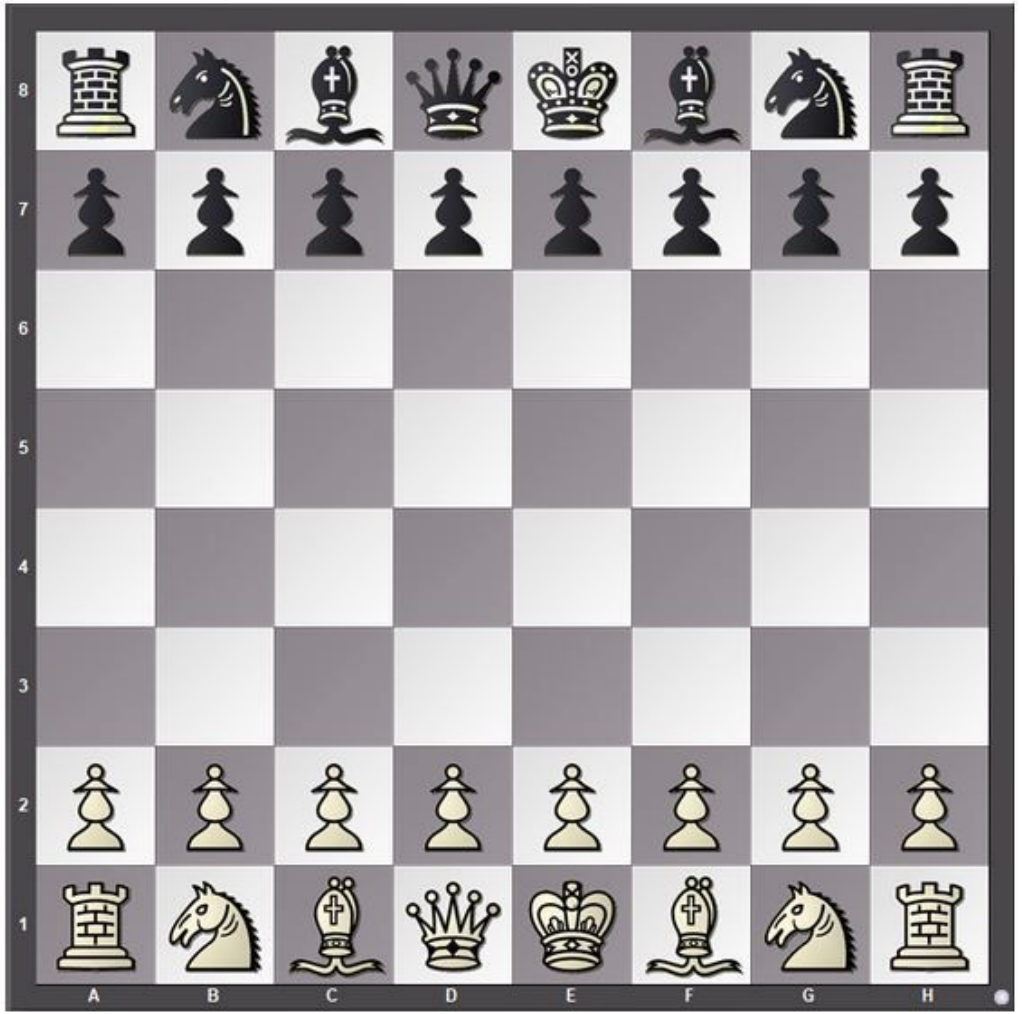
This structure allows you to organize data in a two-dimensional grid.

2D ARRAY

EXAMPLES

01

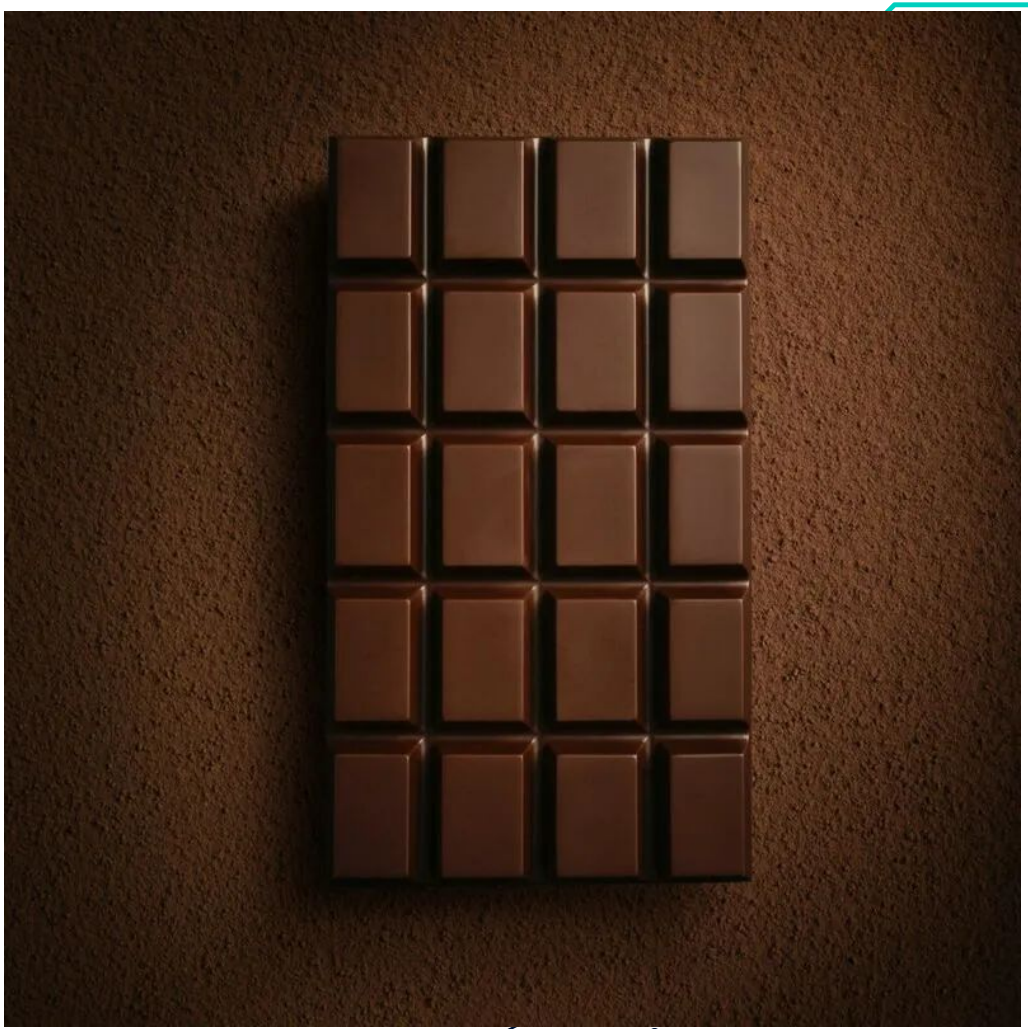
CHESS BOARD



RED & WHITE®
Multimedia Education
Shaping "skills" for "scaling" higher...!!!

02

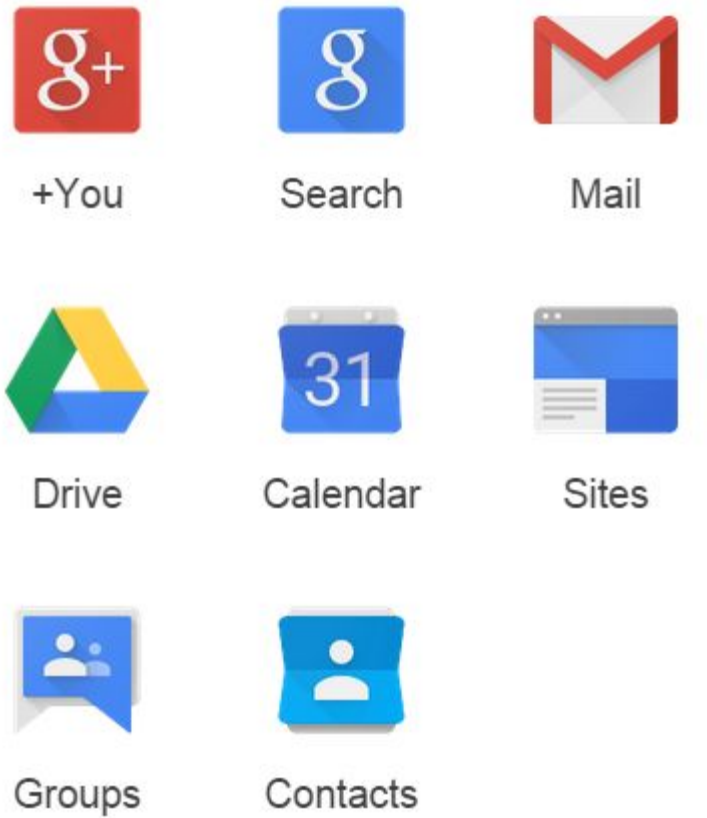
CHOCOLATE GRID



RED & WHITE®
Multimedia Education
Shaping "skills" for "scaling" higher...!!!

03

APPS GRID



RED & WHITE®
Multimedia Education
Shaping "skills" for "scaling" higher...!!!

Let’s see **syntax** of **2D Array** in detail with some examples...

Syntax of 2D Array

```
datatype array_name[row_size][column_size];
```

ARRAY OPERATIONS

There are many operations can be perform on an array. But, here are the **most common operations** of Array:

Insertion	Iteration	Modification / Updation
1	2	3

Let's see **each operations** in detail...

01 Insertion Operation

Insertion Operation

	Elements			index (i)
int a[3][3] =	{{6,	9,	4},	0
	{5,	8,	3},	1
	{7,	4,	2}};	2
index (j)	0	1	2	

Predefined Array

Insertion Operation

	Elements			index (i)
int a[3][3];	0	0	0	0
	0	0	0	1
	0	0	0	2
index (j)	0	1	2	

Empty Array

Insertion Operation

```
int a[3][3];

a[0][0] = 6;
a[0][1] = 9;
a[0][2] = 4;
```

	Elements			index (i)
int a[3][3]; // Empty Array	6	9	4	0
	0	0	0	1
	0	0	0	2
index (j)	0	1	2	

Index-wise static insertion

Insertion Operation

```
a[1][0] = 5;
a[1][1] = 8;
a[1][2] = 3;
```

	Elements			index (i)
int a[3][3]; // Empty Array	6	9	4	0
	5	8	3	1
	0	0	0	2
index (j)	0	1	2	

Index-wise static insertion

Insertion Operation

```
a[2][0] = 7;
a[2][1] = 4;
a[2][2] = 2;
```

	Elements			index (i)
int a[3][3]; // Empty Array	6	9	4	0
	5	8	3	1
	7	4	2	2
index (j)	0	1	2	

Index-wise static insertion

Insertion Operation

	Elements			index (i)
int a[3][3]; // Empty Array	0	0	0	0
	0	0	0	1
	0	0	0	2
index (j)	0	1	2	

Empty Array

Insertion Operation

```
int a[3][3];

scanf("%d", &a[0][0]); // 6
scanf("%d", &a[0][1]); // 9
scanf("%d", &a[0][2]); // 4
```

	Elements			index (i)
int a[3][3]; // Empty Array	6	9	4	0
	0	0	0	1
	0	0	0	2
index (j)	0	1	2	

Index-wise dynamic insertion

Insertion Operation

```
scanf("%d", &a[1][0]); // 5
scanf("%d", &a[1][1]); // 8
scanf("%d", &a[1][2]); // 3
```

	Elements			index (i)
int a[3][3]; // Empty Array	6	9	4	0
	5	8	3	1
	0	0	0	2
index (j)	0	1	2	

Index-wise dynamic insertion

Insertion Operation

```
scanf("%d", &a[2][0]); // 7
scanf("%d", &a[2][1]); // 4
scanf("%d", &a[2][2]); // 2
```

	Elements			index (i)
int a[3][3]; // Empty Array	6	9	4	0
	5	8	3	1
	7	4	2	2
index (j)	0	1	2	

Index-wise dynamic insertion

02

Iteration Operation

Iteration Operation

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

printf("%d", a[0][0]); // 6
printf("%d", a[0][1]); // 9
printf("%d", a[0][2]); // 4
```

	Elements			index (i)
int a[3][3] =	{{6,	9,	4},	0
	{5,	8,	3},	1
	{7,	4,	2}};	2
index (j)	0	1	2	

Index-wise static accessing of elements

```
for(i=0; i<=2; i++)
{
    for(j=0; j<=2; j++)
    {
        printf("%d ", a[i][j]);
    }
    printf("\n");
}
```

	Elements			index (i)
int a[3][3] =	{{6,	9,	4},	0
	{5,	8,	3},	1
	{7,	4,	2}};	2
index (j)	0	1	2	

Index-wise dynamic accessing of elements

03

Modification/Updation Operation

Updation Operation

	Elements			index (i)
int a[3][3] =	{{6,	9,	4},	0
	{5,	8,	3},	1
	{7,	4,	2}};	2
index (j)	0	1	2	

Predefined Array

Updation Operation

```
a[1][1] = 6;
```

	Elements			index (i)
int a[3][3];	6	9	4	0
	5	6	3	1
	7	4	2	2
index (j)	0	1	2	

Index-wise static updation

Updation Operation

```
scanf("%d", &a[2][0]); // 9
```

	Elements			index (i)
int a[3][3];	6	9	4	0
	5	6	3	1
	9	4	2	2
index (j)	0	1	2	

Index-wise dynamic updation

LANGUAGE

Let’s start now...