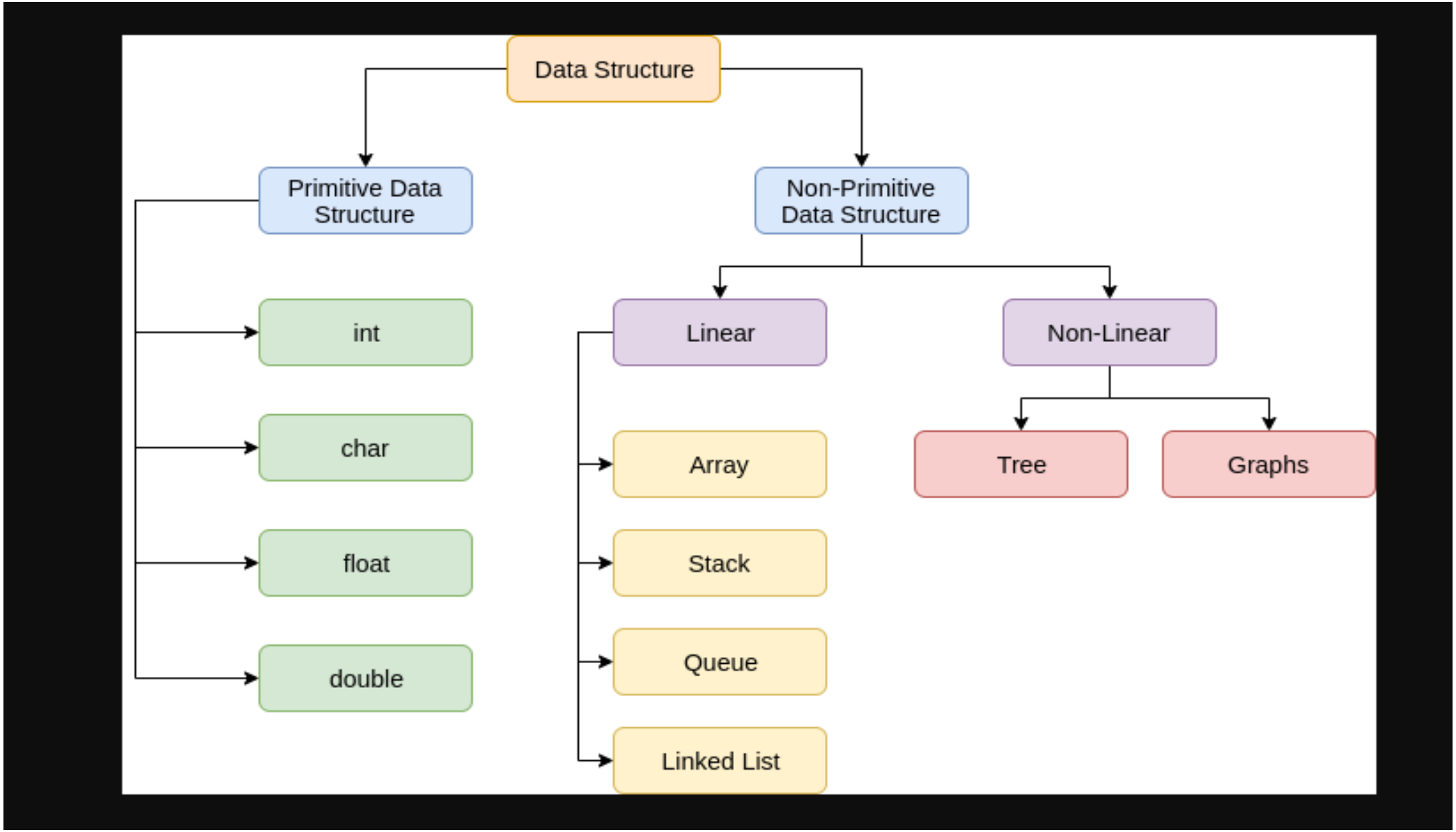
Unit 1

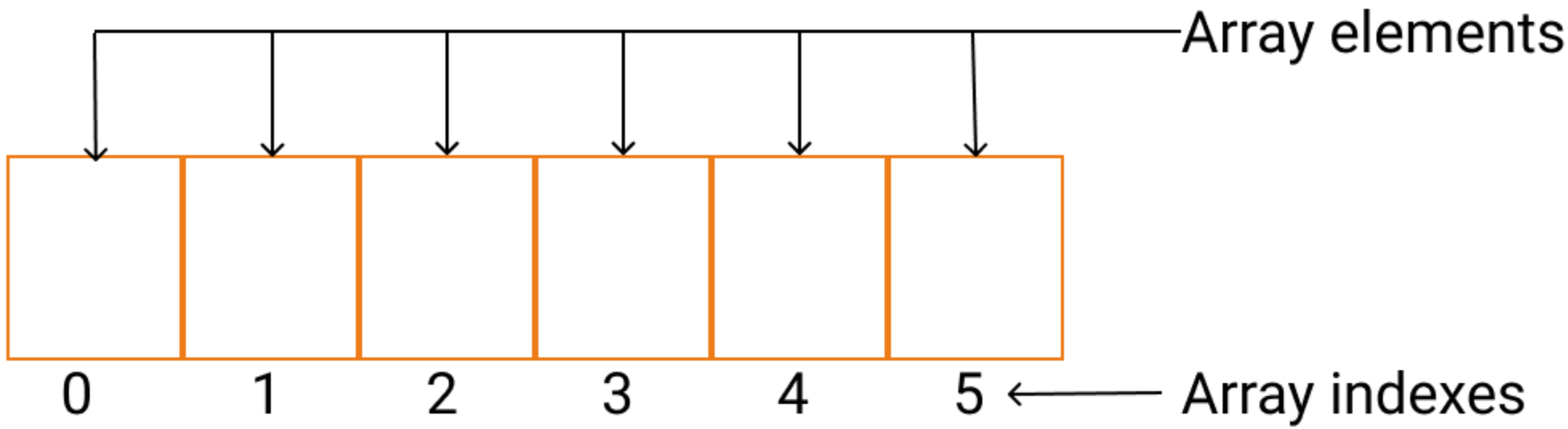
Basics of array

Array in Data Structure

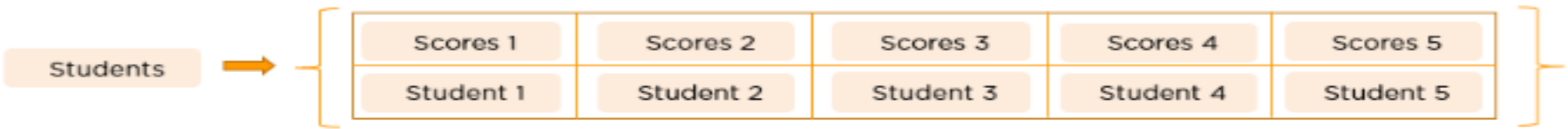
- Arrays are defined as the collection of similar types of data items stored at contiguous memory locations.
- It is one of the simplest data structures where each data element can be randomly accessed by using its index number.


Data Structure
Primitive Data Structure
Non-Primitive Data Structure
int
char
float
double
Linear
Non-Linear
Array
Stack
Queue
Linked List
Tree
Graphs

- In C programming, they are the derived data types that can store the primitive type of data such as int, char, double, float, etc.
- For example, if we want to store the marks of a student in 6 subjects, then we don't need to define a different variable for the marks in different subjects.
- Instead, we can define an array that can store the marks in each subject at the contiguous memory locations.


Array elements
0
1
2
3
4
5
Array indexes

Why Do You Need an Array in Data Structures?



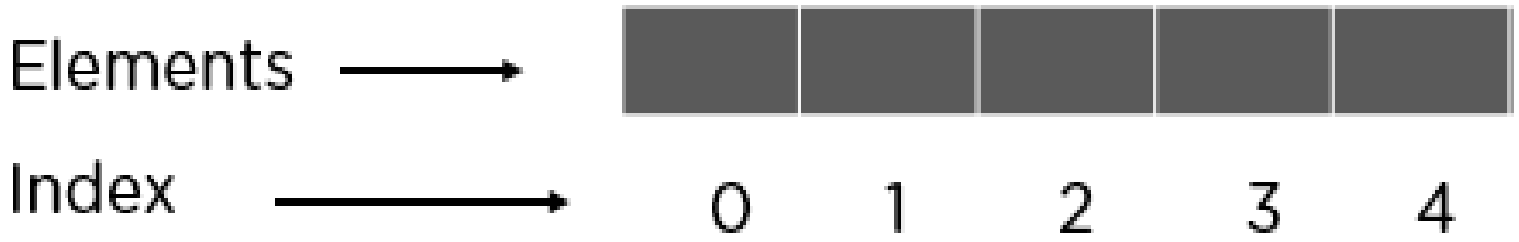
Let's suppose a class consists of ten students, and the class has to publish their results. If you had declared all ten variables individually, it would be challenging to manipulate and maintain the data.

If more students were to join, it would become more difficult to declare all the variables and keep track of it. To overcome this problem, arrays came into the picture.

What Are the Types of Arrays?

There are majorly two types of arrays, they are:

One-Dimensional Arrays:

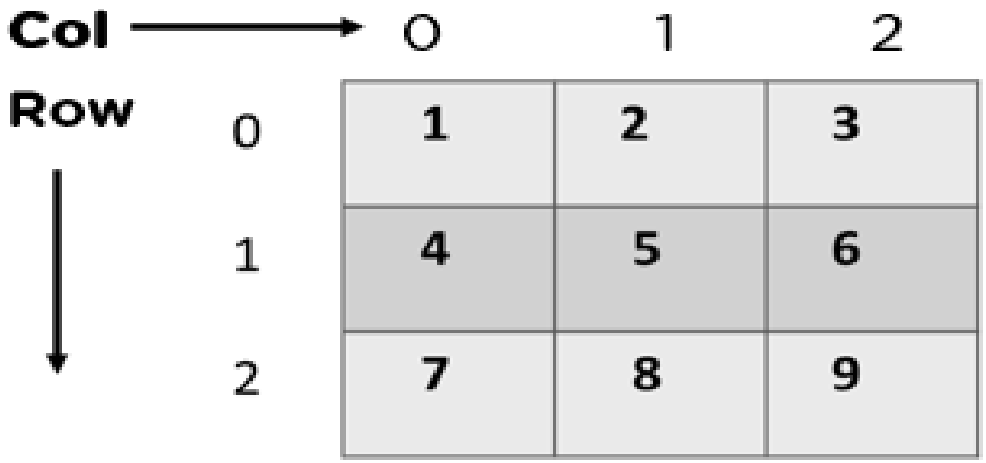


You can imagine a 1d array as a row, where elements are stored one after another.

Multi-Dimensional Arrays:

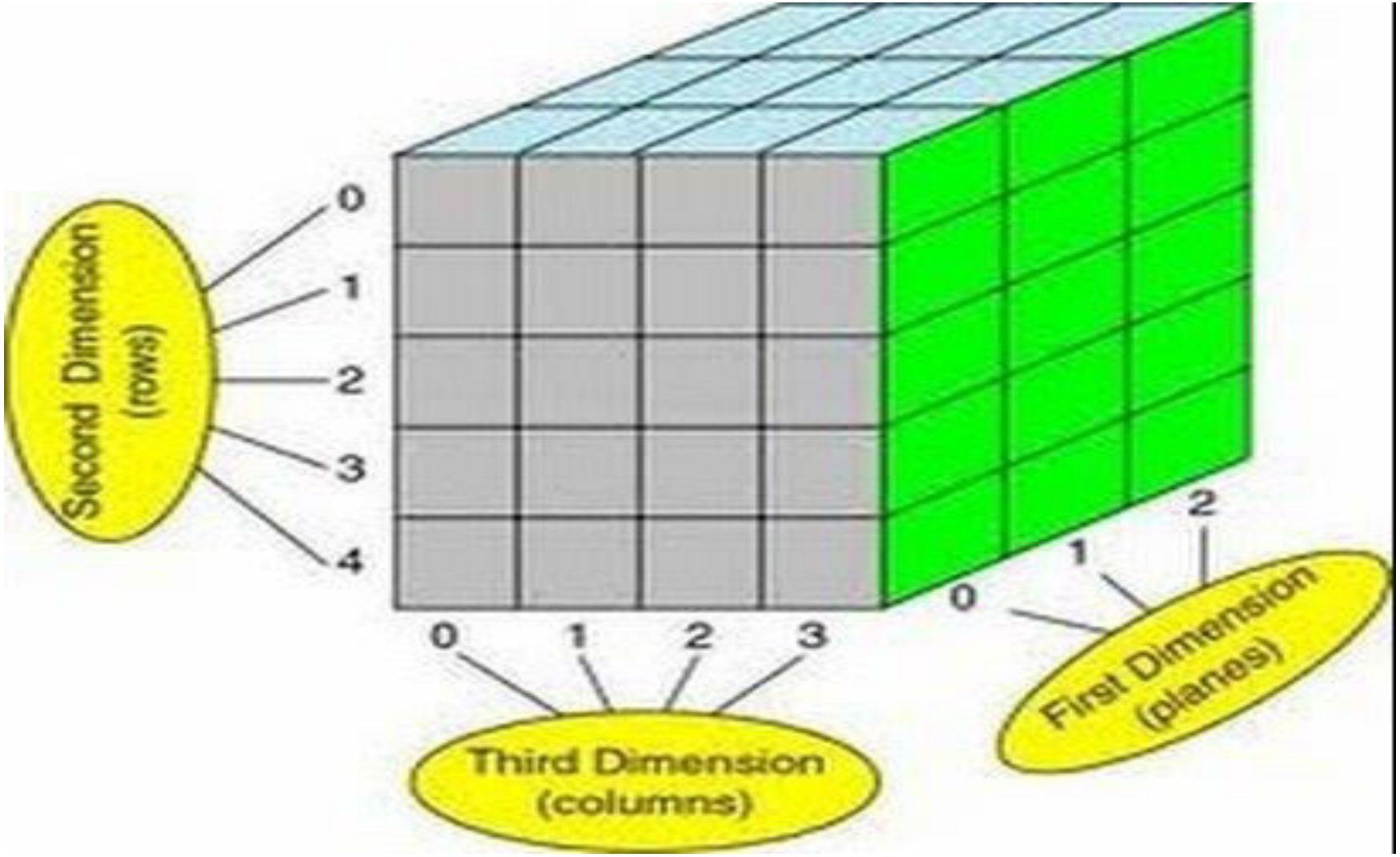
These multi-dimensional arrays are again of two types. They are:

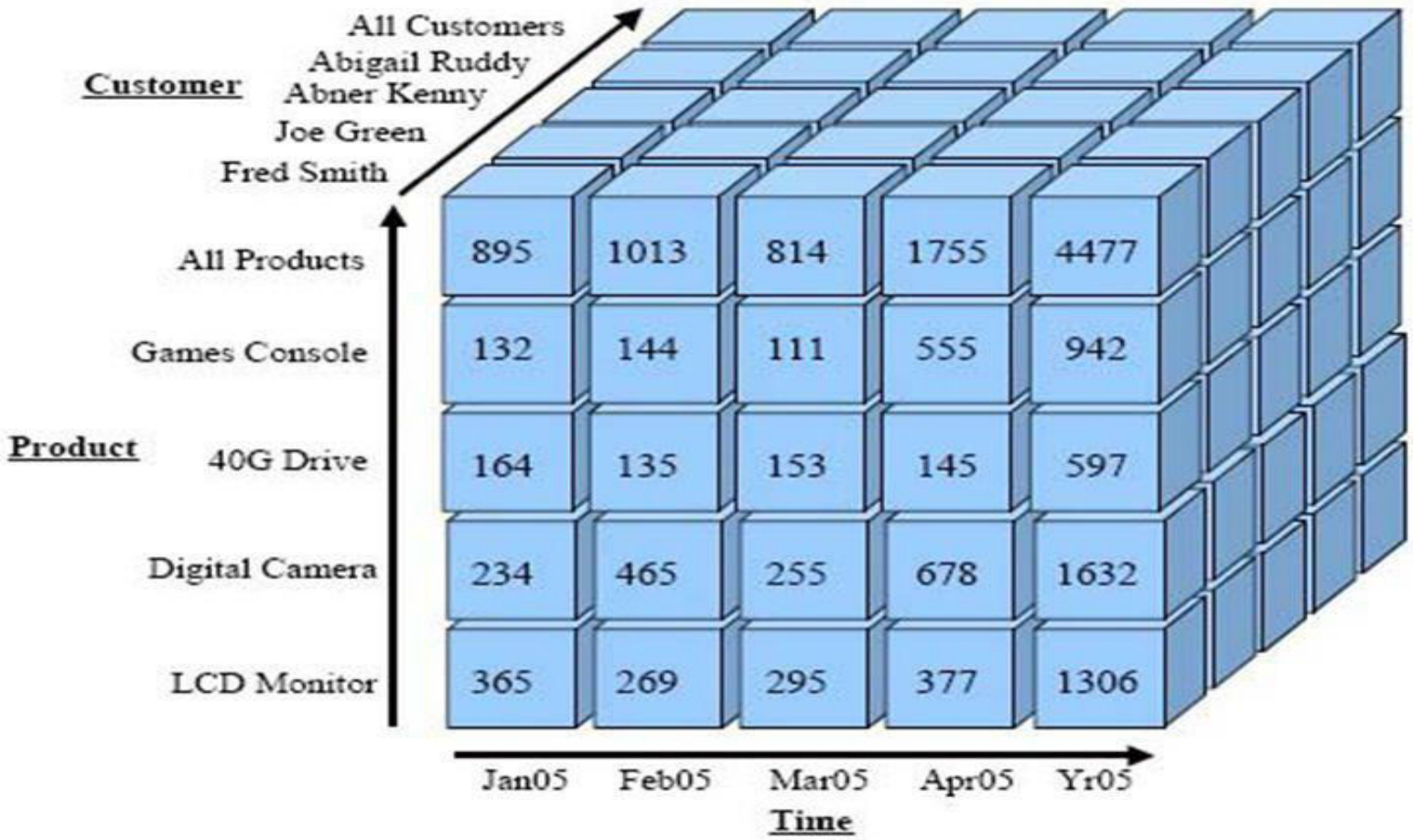
Two-Dimensional Arrays:



You can imagine it like a table where each cell contains elements.

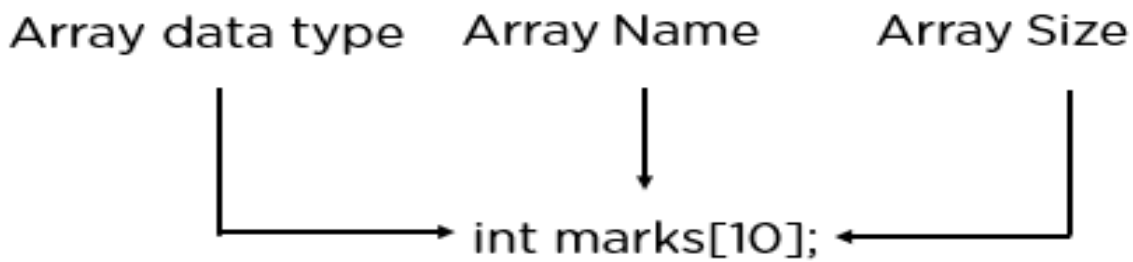
3 dimensional array





OLAP Data Hypercube (No. of Dimensions = 3)

How Do You Declare an Array?



Arrays are typically defined with square brackets with the size of the arrays as its argument.

Here is the syntax for arrays:

1D Arrays: int arr[n];

2D Arrays: int arr[m][n];

3D Arrays: int arr[m][n][o];

How Do You Initialize an Array?

You can initialize an array in four different ways:

- **Method 1:**

```
int a[6] = {2, 3, 5, 7, 11, 13};
```

- **Method 2:**

```
int arr[]= {2, 3, 5, 7, 11};
```

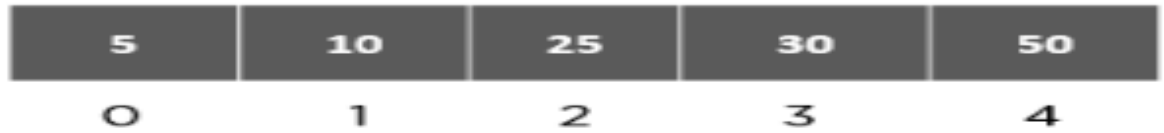
- **Method 3:**

```
int n;

scanf("%d",&n);

int arr[n];

for(int i=0;i<5;i++)

{

scanf("%d",&arr[i]);

}
```

- ## Method 4:

```
int arr[5];

arr[0]=1;

arr[1]=2;

arr[2]=3;

arr[3]=4;

arr[4]=5;
```

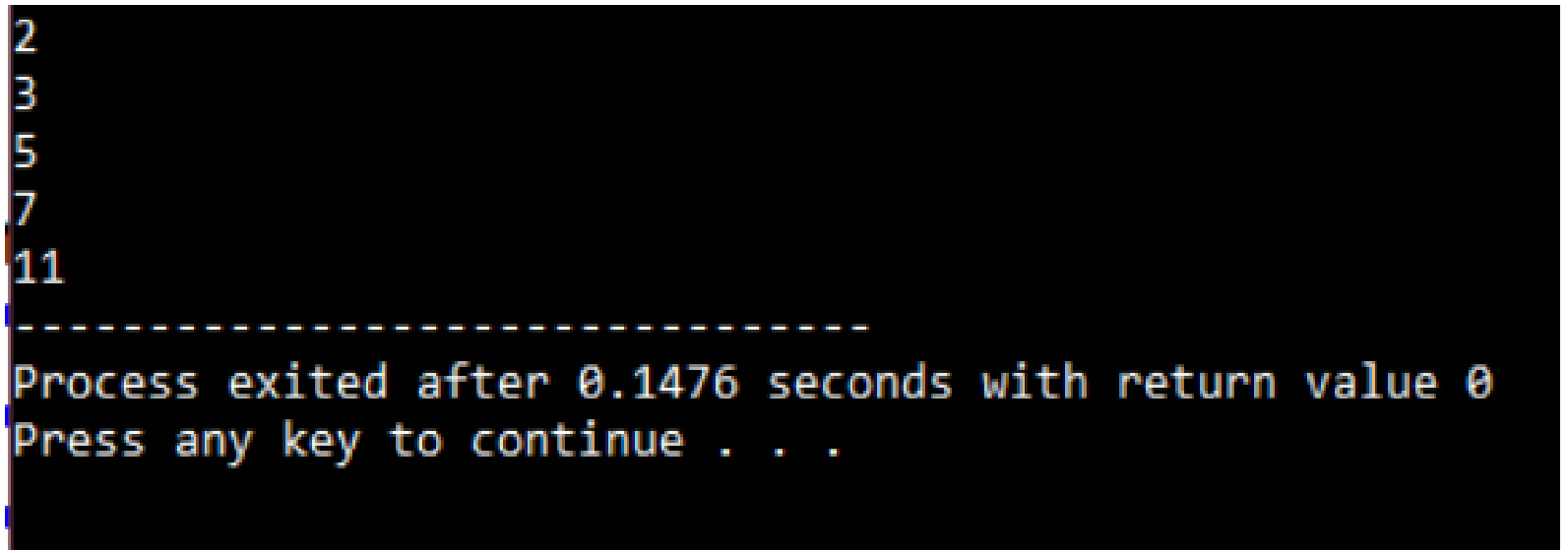
How Can You Access Elements of Arrays in Data Structures?



You can access elements with the help of the index at which you stored them. Let's discuss it with a code:

```
#include<stdio.h>

int main()

{

int a[5] = {2, 3, 5, 7, 11};

printf("%d\n",a[0]); // we are accessing

printf("%d\n",a[1]);

printf("%d\n",a[2]);

printf("%d\n",a[3]);

printf("%d",a[4]);

return 0;

}
```

output


2
3
5
7
11

Process exited after 0.1476 seconds with return value 0
Press any key to continue . . .