Algorithms and Data Structure

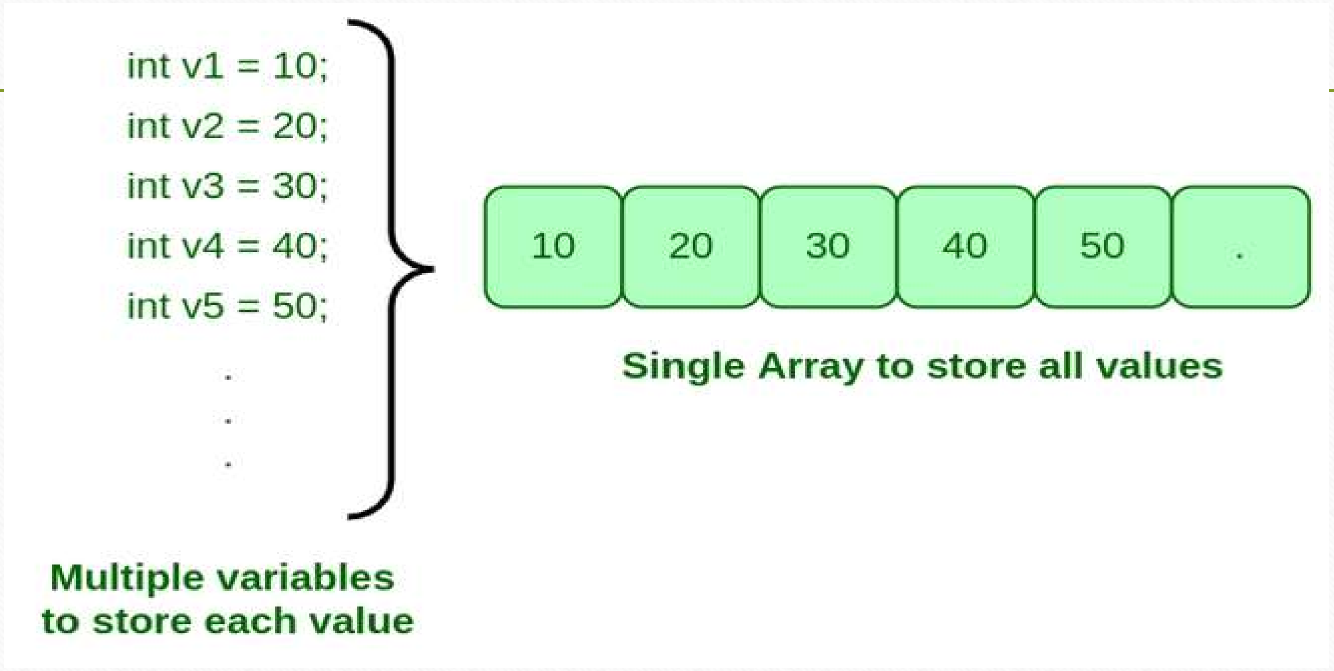
Introduction to Arrays

Mr Haniche Fayçal

Why Array Data Structures is needed?

- Assume there is a class of five students and if we have to keep records of their marks in examination then, we can do this by declaring five variables individual and keeping track of records but what if the number of students becomes very large, it would be challenging to manipulate and maintain the data.
- What it means is that, we can use normal variables (v1, v2, v3, ..) when we have a small number of objects. But if we want to store a large number of instances, it becomes difficult to manage them with normal variables. **The idea of an array is to represent many instances in one variable**..

Why Array Data Structures is needed?



What is an Array?

- An **array** is a collection of items of the same variable type that are stored at contiguous memory locations. It’s one of the most popular and simple data structures and is often used to implement other data structures.
- We can directly access an array element by using its index value.

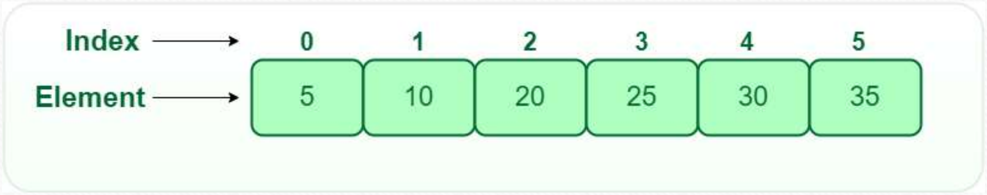
Basic terminologies of array

- **Array Index:** In an array, elements are identified by their indexes. Which represents the number of element starting from initial value for the first element
- **Array element:** Elements are items stored in an array and can be accessed by their index.
- **Array Length (Size):** The length of an array is determined by the number of elements it can contain.

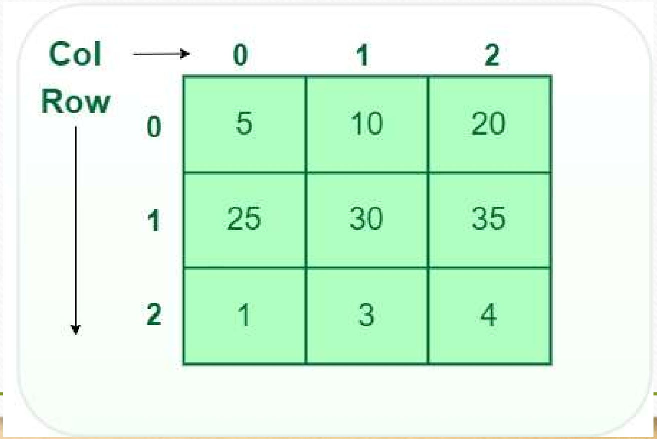
Types of arrays

There are majorly two types of arrays:

- **One-dimensional array (1-D arrays called Vectors)**: You can imagine a 1d array as a row, where elements are stored one after another.



- **Two-dimensional array (Matrix):** 2-D Multidimensional arrays can be considered as an array of arrays or as
  a matrix consisting of rows and columns.



Advantages of using Arrays:

- Arrays allow random access to elements. This makes accessing elements by position faster.
- Arrays have better cache locality which makes a pretty big difference in performance.
- Arrays represent multiple data items of the same type using a single name.
- Arrays store multiple data of similar types with the same name.
- Array data structures are used to implement the other data structures like linked lists, stacks, queues, trees, graphs, etc.

Disadvantages of Array

- As arrays have a fixed size, once the memory is allocated to them, it cannot be increased or decreased, making it impossible to store extra data if required. An array of fixed size is referred to as a static array.
- Allocating less memory than required to an array leads to loss of data. An array is homogeneous in nature so, a single array cannot store values of different data types.
- Arrays store data in contiguous memory locations, which makes deletion and insertion very difficult to implement. This problem is overcome by implementing linked lists, which allow elements to be accessed sequentially.

One-dimensional array

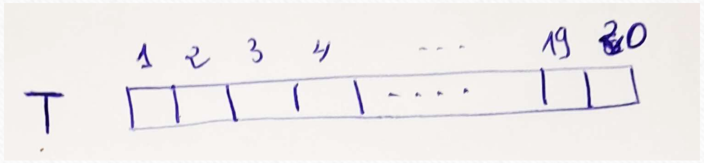
- Declaration

form1 : <Id_array> : array [size] <dataType> ;

example

moy : array [50] real ;

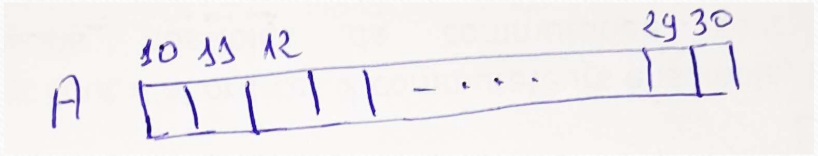
T : array [20] integer;



form2 : <id_array> : array[Vi . . Vf] <datatype>

example

A: array[10 . . 30] integer;



How to access array elements

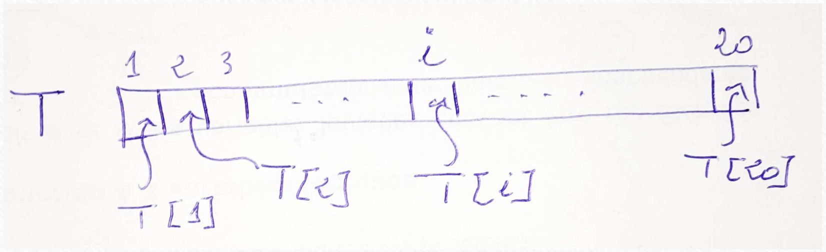
- Array elements are accessed using their index.

supposing the following declaration T : array [20] integer;

The first element is referenced as T[1]

The second is referenced as T[2]

The i th element is referenced as T[i]
and so one



The array elements are manipulated individually as simple variables

How to browse all array elements?

- For manipulating all array elements we need to browse it using a for loop

(we »use the loop counter as array index to access the elements one after another)

```
For  i := vi  to vf  do
 begin
    <Manipulation of  the i th   element>
 end;
```