Array Data Structure

An array is a collection of items stored at contiguous memory locations.

The idea is to store multiple items of the same type together.

This makes it easier to calculate the position of each element by simply adding an offset to a base value, i.e.,

the memory location of the first element of the array (generally denoted by the name of the array).



ADAVANTAGES AND DRAWBACKS OF USING THE ARRAY

The access time to an element by index is constant, regardless of the desired element. This is explained by the fact that the array elements are contiguous in memory. Thus, it is possible to calculate the memory address of the element to be accessed, from the base address of the array and the index of the element. Access is immediate, as it would be for a single variable.

This advantage is also one of the limits of such a structure. An array is represented in memory in the form of contiguous cells, insert and delete operations element are impossible, unless you create a new array, larger or smaller (depending on the operation). It is then necessary to copy all elements of the original array into the new one, and then release the memory space allocated to the old array. So that's a lot of operations and require some languages ​​providing such opportunities to implement their own array, not in the traditional form (adjacent cells), but using a linked list, or a combination of both structures to improve performance.