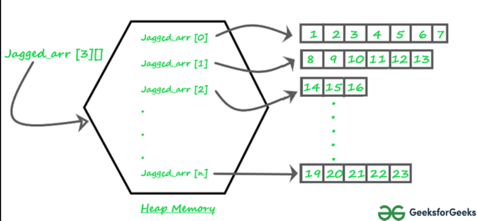
ARRAY

1. Array is a list of variables of a single type. If a data structure contains data of different type, then it cannot be called array.
2. Contiguous area of memory (whole at one place) consisting of equal size elements indexed by continuous integer.
3. Indexing starts from 0 in most of the languages while some languages (Julia, Fortran, R) have its initialization starting from 1 also we need to tell where to initialize in some.
4. Initialized to 0 by default in case of integer type.
5. 0 indexing means that array [10] element at 10 is not valid.
6. Constant time to read and access and have random access.
7. We usually face input buffer in the code while taking 2-D input hence use cin.sync();

in case of misbehaviour.

1. Row major 🡪 In 2-D array (like matrixes) while allocating we have our column number change more rapidly than row number and this is called row major. Moving across the rows and then down the next row, to successive memory locations
2. Column major 🡪



1. Jagged Arrays 🡪 An array of arrays such that member arrays can be of different sizes, i.e., we can create a 2-D array but with a variable number of columns in each row.
2. Dynamic arrays 🡪 is an array with resizable capacity.

It is known as ARRAY LIST in Java

VECTOR in C++

ARRAY in JavaScript

LIST in Python

Vs ll

Use array when we are sure about the size. This can be created in the heap or stack.

The linked list is always created in the heap i.e. collection of nodes is created always in heap and can be accessed using a pointer and that pointer is created inside the stack. So, a linked list is always created in the heap.

Vectors

While creating vectors exceed the memory then a new array is created and all the data is copied and the previous one is deleted.