**Array Representation in Memory**

* **Contiguous Memory Allocation**: Arrays are stored in contiguous memory locations. This means that all elements of the array are stored sequentially in the memory, allowing direct access to any element using its index.
* **Indexing**: The index of an array starts from 0, and the element at any position can be accessed directly using its index. For example, the element at index i can be accessed in constant time, O(1)O(1)O(1), using the expression array[i].
* **Advantages**:
  + **Direct Access**: The primary advantage of arrays is that they allow direct access to elements, making retrieval operations very fast.
  + **Cache Locality**: Due to contiguous memory allocation, accessing elements sequentially can take advantage of CPU caching, leading to faster access times.
  + **Fixed Size**: Arrays have a fixed size, which can simplify memory management and reduce overhead compared to dynamic structures.

**Array Representation in Memory**

* **Contiguous Memory Allocation**: Arrays are stored in contiguous memory locations. This means that all elements of the array are stored sequentially in the memory, allowing direct access to any element using its index.
* **Indexing**: The index of an array starts from 0, and the element at any position can be accessed directly using its index. For example, the element at index i can be accessed in constant time, O(1)O(1)O(1), using the expression array[i].
* **Advantages**:
  + **Direct Access**: The primary advantage of arrays is that they allow direct access to elements, making retrieval operations very fast.
  + **Cache Locality**: Due to contiguous memory allocation, accessing elements sequentially can take advantage of CPU caching, leading to faster access times.
  + **Fixed Size**: Arrays have a fixed size, which can simplify memory management and reduce overhead compared to dynamic structures.