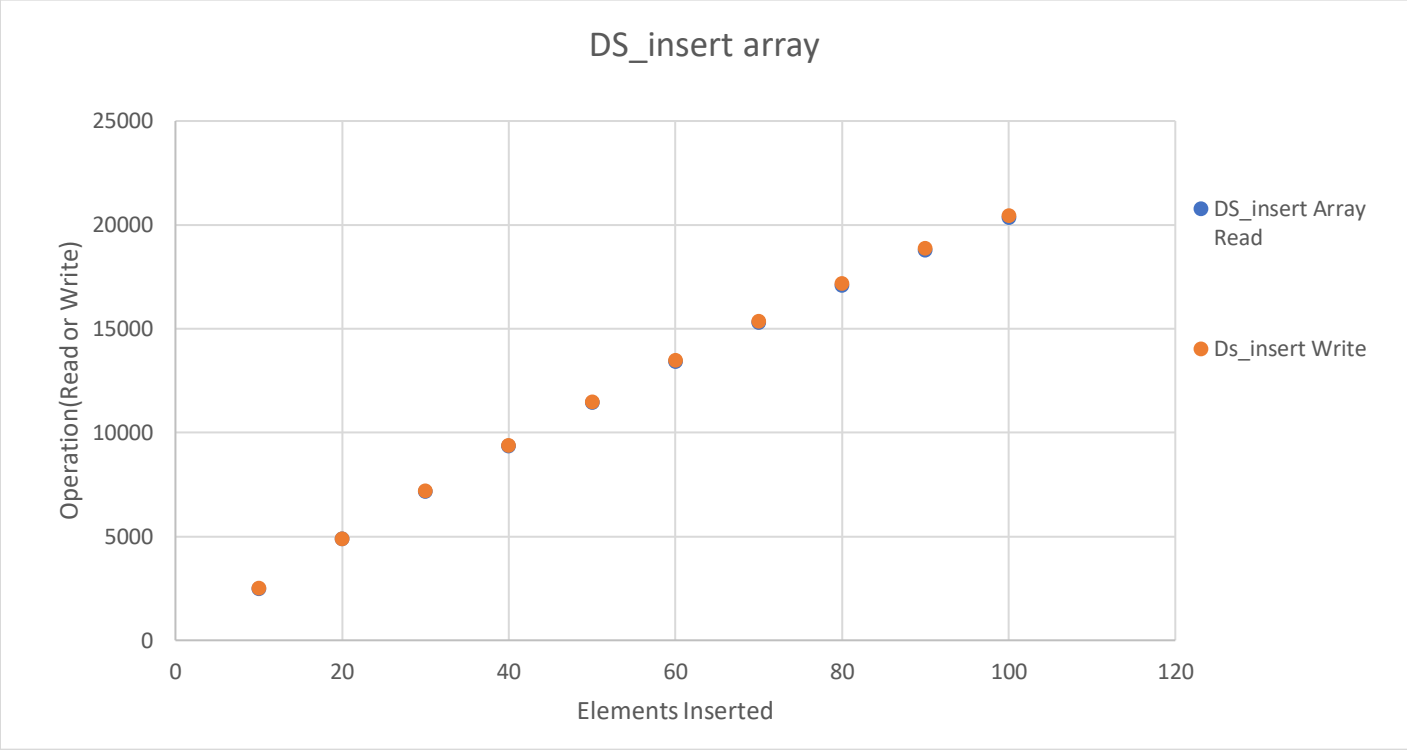
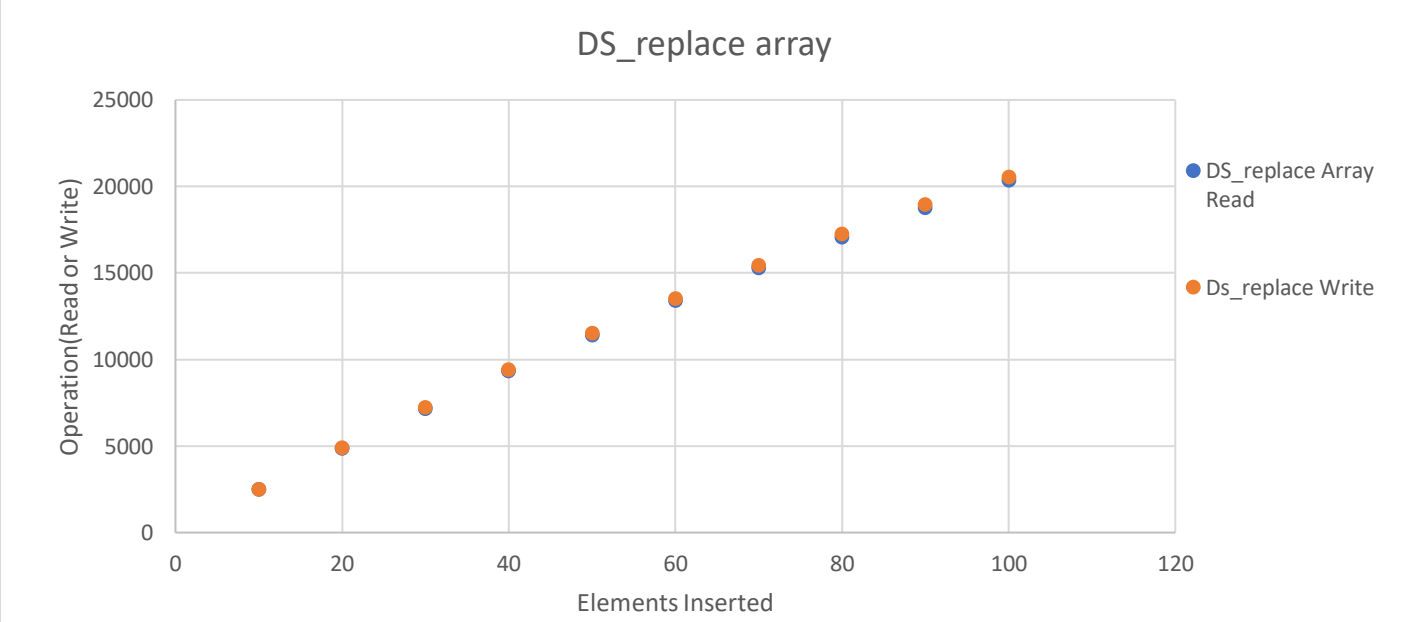
Results From Program:

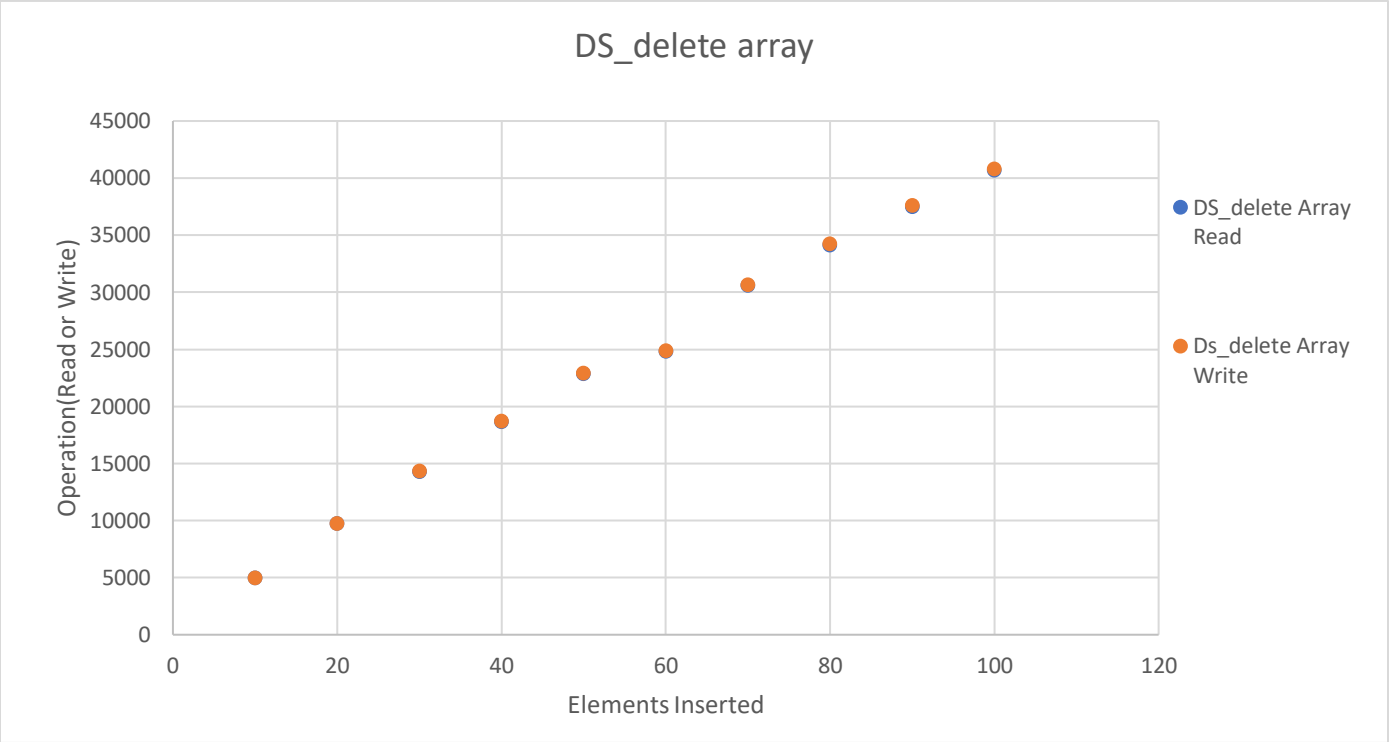
DS_insert Array:



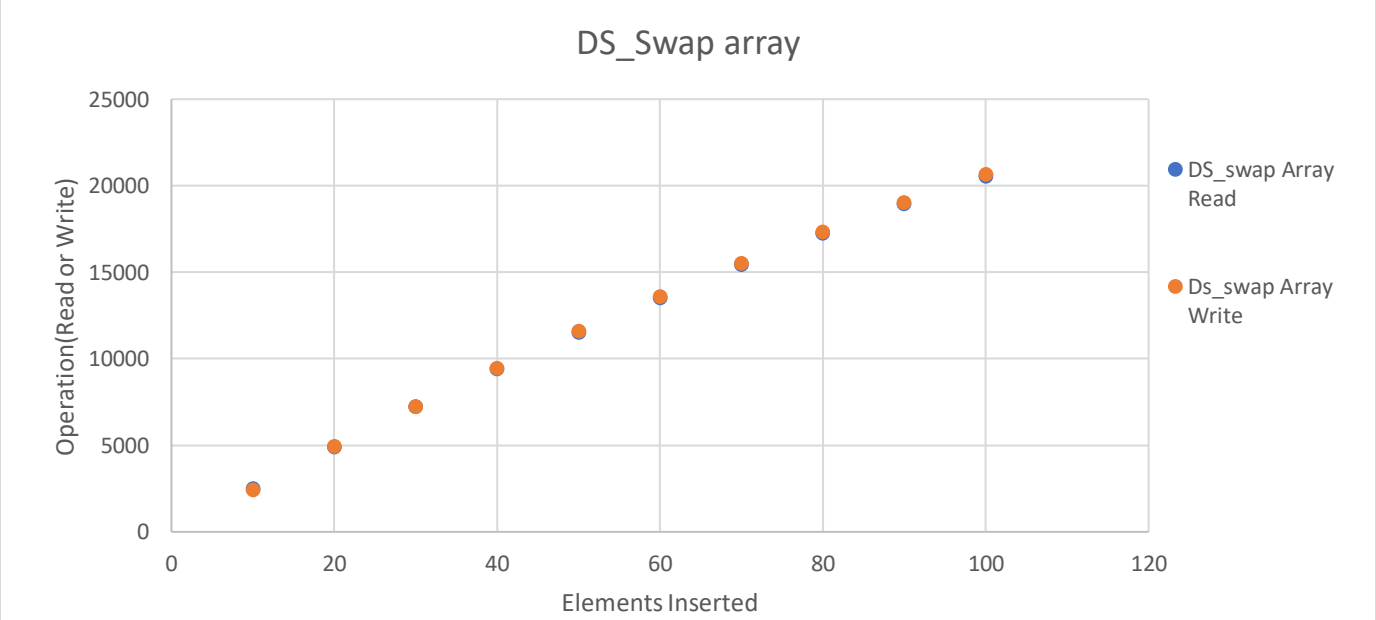
DS_replace Array:



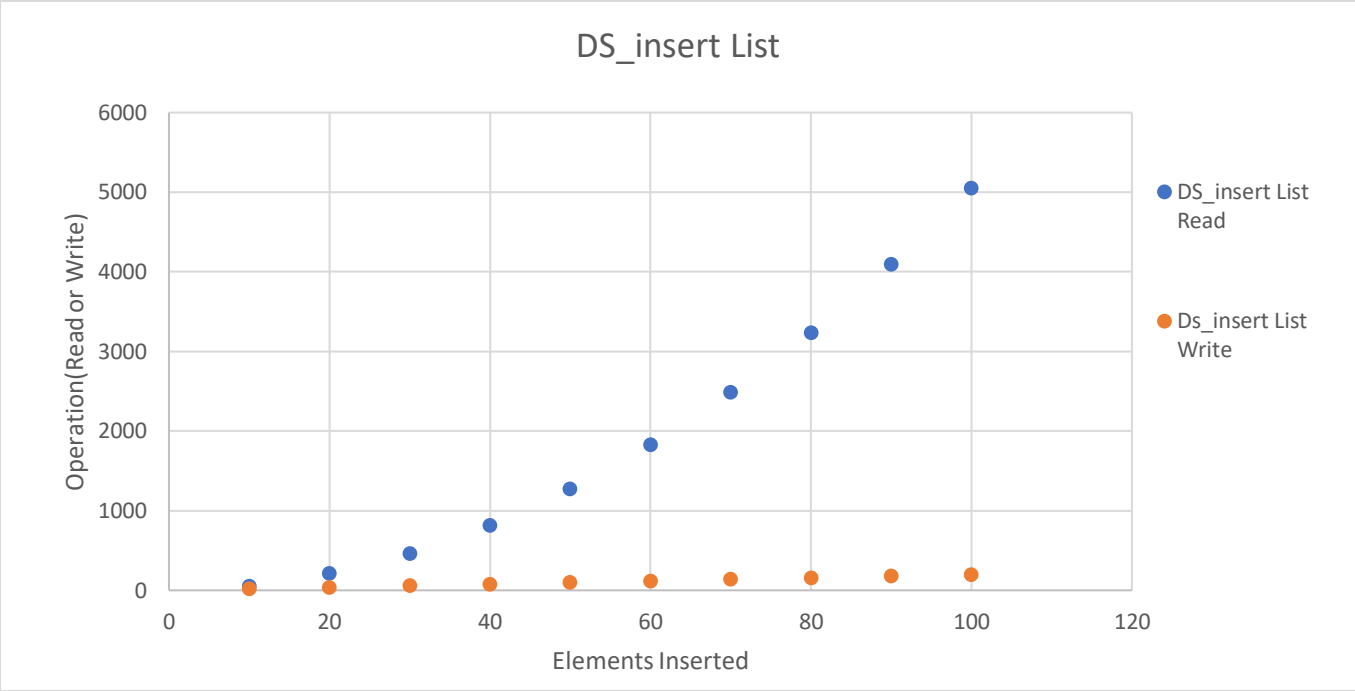
DS_delete Array:



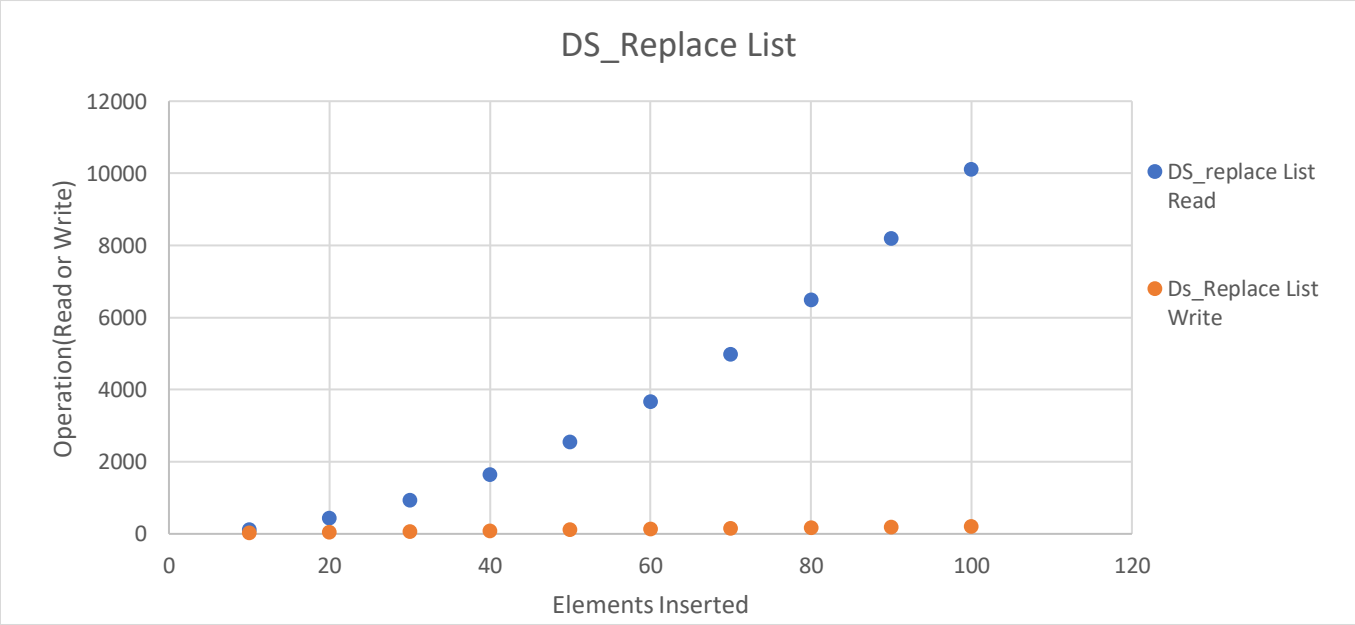
Ds_swap Array:



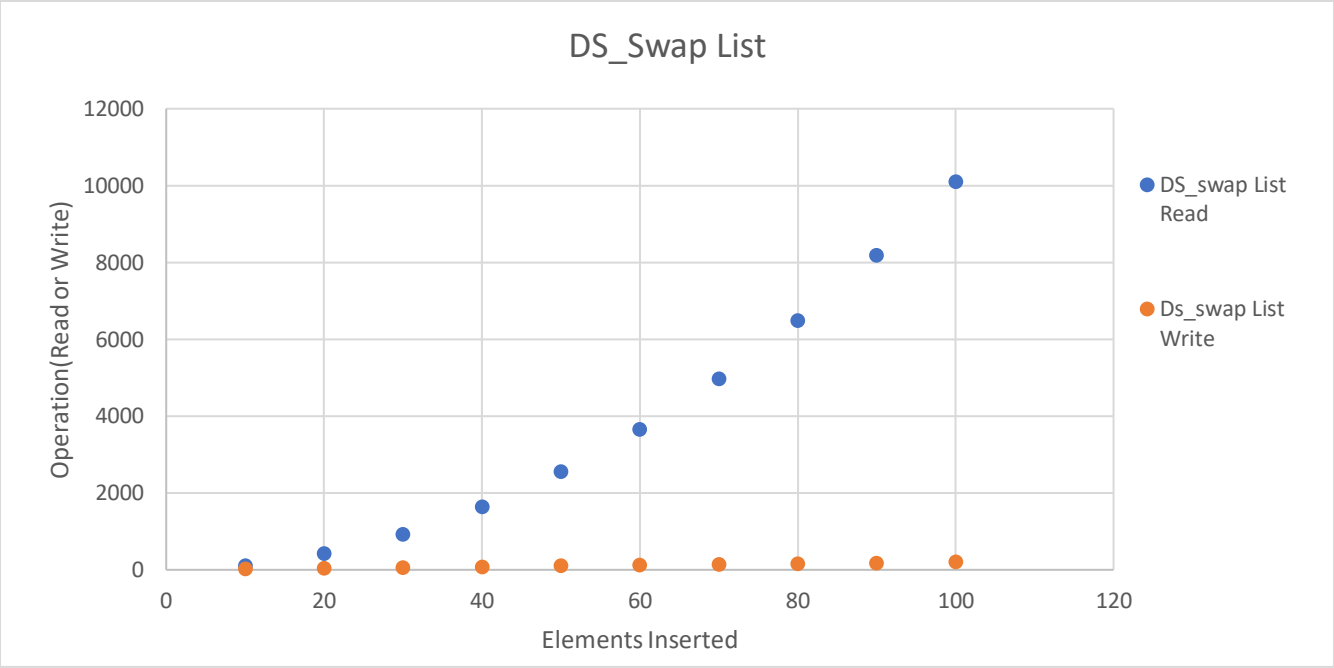
DS_insert List:



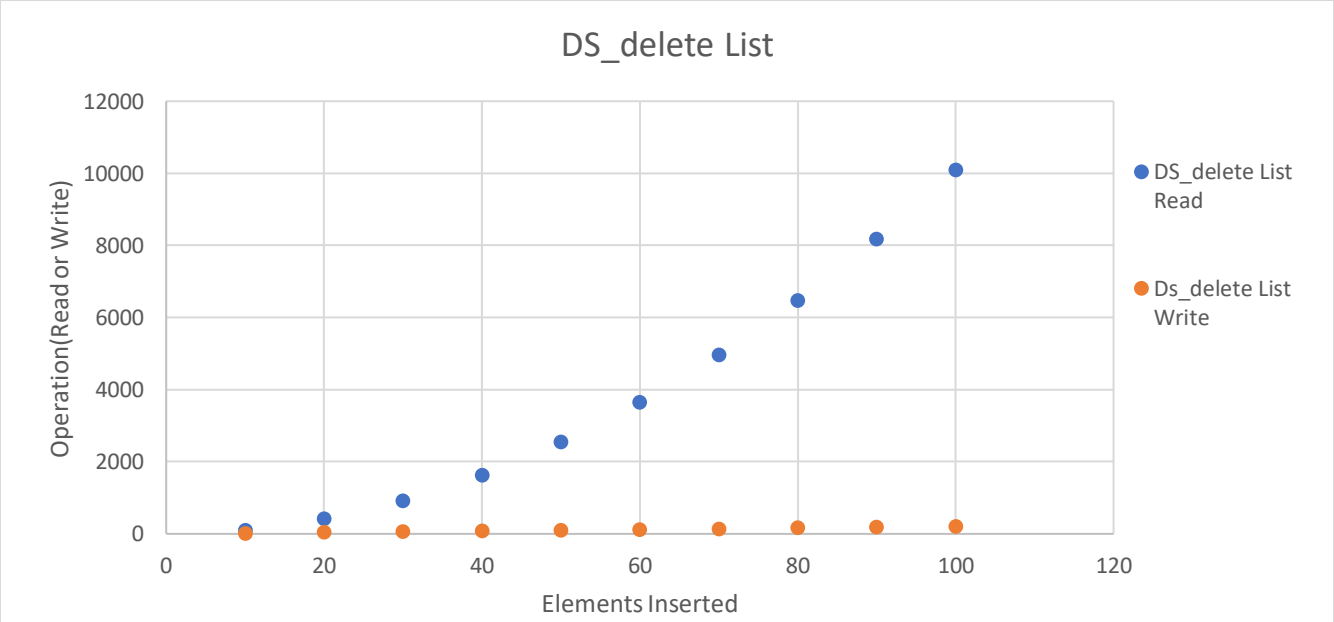
DS_Replace List:



DS_Swap List:



DS_delete List:



Result Analysis:

Both Linked List and Array are used to store similar type of linear data, but an array absorbs contiguous storage positions reserved at the point of initialization, i.e. at the moment of array creation, whereas space is delegated as and when information is applied to it, i.e. at runtime, for a linked list.

This is the fundamental difference between a linked list and an array, and the most important. We should address this in depth in the chapter below as well as explain certain variations.

Array is a datatype that is widely implemented in almost all modern programming languages as a default type and is used to store similar type of data.

But there are many use cases, such as the one where we don't realise the amount of data to be processed, which requires advanced data structures, so one such data structure is called a linked list.

The table below summarizes the differences between both data structures:

ARRAY	Linked List
Array is a collection of similar type of data elements.	Linked List is an organised set of the same form of elements that are linked by means of pointers.
Array supports random access, which means that elements can be directly accessed using their index, such as arr[0] for the first element, arr[6] for the seventh element, etc. Accessing elements in an array is therefore fast with O(1)'s constant time complexity.	Linked List supports Sequential Access, which ensures that we must sequentially traverse the entire linked list, up to that point, to access each component / node in a linked list. To access nth element of a linked list, time complexity is O(n).
In an array, elements are stored in contiguous memory location or consecutive manner in the memory.	In a linked list, new elements can be stored anywhere in the memory. Address of the memory location allocated to the new element is stored in the previous node of linked list, hence formatting a link between the two nodes/elements.
In array, Insertion and Deletion operation takes more time, as the memory locations are consecutive and fixed.	In case of linked list, a new element is stored at the first free and available memory location, with only a single overhead step of storing the address of memory location in the previous node of linked list. Insertion and Deletion operations are fast in linked list.
Memory is allocated as soon as the array is declared, at compile time. It's also known as Static Memory Allocation.	Memory is allocated at runtime, as and when a new node is added. It's also known as Dynamic Memory Allocation.
In array, each element is independent and can be accessed using it's index value.	In case of a linked list, each node/element points to the next, previous, or maybe both nodes.
Size of the array must be specified at time of array declaration.	Size of a Linked list is variable. It grows at runtime, as more nodes are added to it.