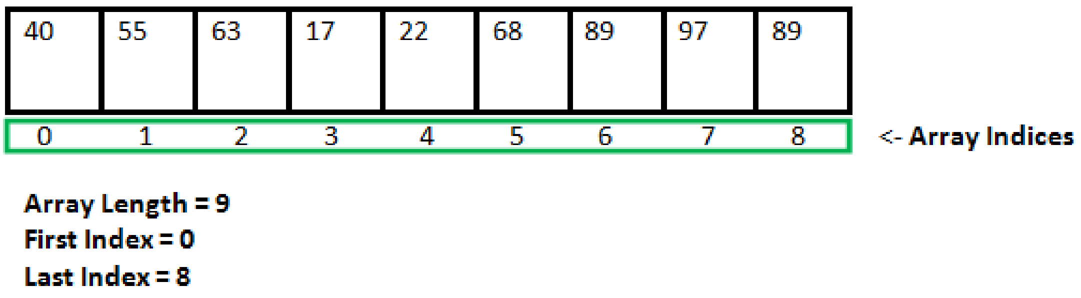
Linked List vs Array

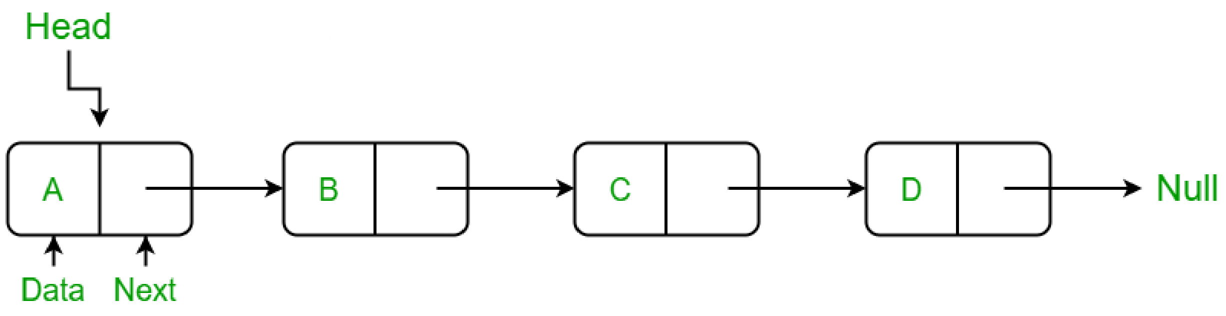
Last Updated : 17 Feb, 2025

Array: Arrays store elements in contiguous memory locations, resulting in easily calculable addresses for the elements stored and this allows faster access to an element at a specific index.



Data storage scheme of an array

Linked List: Linked lists are less rigid in their storage structure and elements are usually not stored in contiguous locations, hence they need to be stored with additional tags giving a reference to the next element.



Linked-List representation

Advantages of Linked List over arrays :

- **Efficient insertion and deletion**: Linked lists allow insertion and deletion in the middle in O(1) time, if we have a pointer to the target position, as only a few pointer changes are needed. In contrast, arrays require O(n) time for insertion or deletion in the middle due to element shifting.

- **Implementation of Queue and Deque** : Simple array implementation is not efficient at all. We must use circular array to efficiently implement which is complex. But with linked list, it is easy and straightforward. That is why most of the language libraries use Linked List internally to implement these data structures.
- **Space Efficient in Some Cases :** Linked List might turn out to be more space efficient compare to arrays in cases where we cannot guess the number of elements in advance. In case of arrays, the whole memory for items is allocated together. Even with dynamic sized arrays like vector in C++ or list in Python or ArrayList in Java. the internal working involves de-allocation of whole memory and allocation of a bigger chunk when insertions happen beyond the current capacity.
- **Circular List with Deletion/Addition :** Circular Linked Lists are useful to implement CPU round robin scheduling or similar requirements in the real world because of the quick deletion/insertion in a circular manner.

Advantages of Arrays over Linked List :

- **Random Access**. : We can access ith item in O(1) time (only some basic arithmetic required using base address). In case of linked lists, it is O(n) operation due to sequential access**.**
- **Cache Friendliness** : Array items (Or item references) are stored at contiguous locations which makes array cache friendly (Please refer Spatial locality of reference for more details)
- **Easy to use :** Arrays are relatively very easy to use and are available as core of programming languages
- **Less Overhead :** Unlike linked list, we do not have any extra references / pointers to be stored with every item.