**Assignment 3**

Date : 21-04-2022

**1. Explain linked list and its structure?**

A linked list is a sequence of data structures, which are connected together via links.

Linked List is a sequence of links which contains items. Each link contains a connection to another link. Linked list is the second most-used data structure after array. Following are the important terms to understand the concept of Linked List.

* **Link** − Each link of a linked list can store a data called an element.
* **Next** − Each link of a linked list contains a link to the next link called Next.
* **LinkedList** − A Linked List contains the connection link to the first link called First.

**2. List various operations performed on Linked list.**

* Traversal - access each element of the linked list.
* Insertion - adds a new element to the linked list.
* Deletion - removes the existing elements.
* Search - find a node in the linked list.
* Sort - sort the nodes of the linked list.

**3. Explain what does it mean by self-referential structure? Explain its need?**

The self-referential structure is a structure that points to the same type of structure. It contains one or more pointers that ultimately point to the same structure.

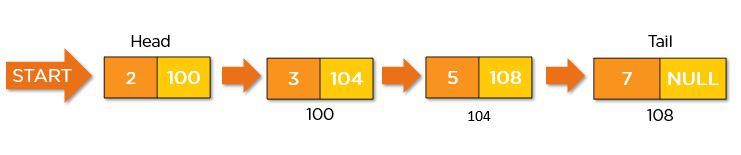
self referential structures are **very useful in applications that involve linked data structures, such as lists and trees**.

**4.Difference between Array and Linklist**

|  |  |  |
| --- | --- | --- |
| **Sr.No.** | **Array** | **Linklist** |
| 1 | An array is a grouping of data elements of equivalent data type. | A linked list is a group of entities called a node. The node includes two segments: data and address. |
| 2 | It stores the data elements in a contiguous memory zone. | It stores elements randomly, or we can say anywhere in the memory zone. |
| 3 | In the case of an array, memory size is fixed, and it is not possible to change it during the run time. | In the linked list, the placement of elements is allocated during the run time. |
| 4 | The elements are not dependent on each other. | The data elements are dependent on each other. |
| 5 | The memory is assigned at compile time. | The memory is assigned at run time. |
| 6 | It is easier and faster to access the element in an array. | In a linked list, the process of accessing elements takes more time. |
| 7 | In the case of an array, memory utilization is ineffective. | In the case of the linked list, memory utilization is effective. |
| 8 | When it comes to executing any operation like insertion, deletion, array takes more time. | When it comes to executing any operation like insertion, deletion, the linked list takes less time. |

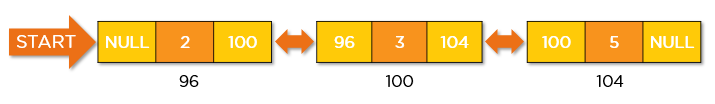
**5. List types of Linked List.**

1] [singly linked list](https://www.simplilearn.com/tutorials/data-structure-tutorial/singly-linked-list)



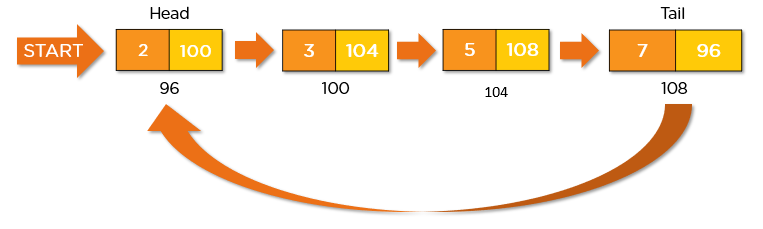
A [singly linked list](https://www.simplilearn.com/tutorials/data-structure-tutorial/singly-linked-list) is a unidirectional linked list. So, you can only traverse it in one direction, i.e., from head node to tail node.

2] [doubly linked list](https://www.simplilearn.com/tutorials/data-structure-tutorial/singly-linked-list)



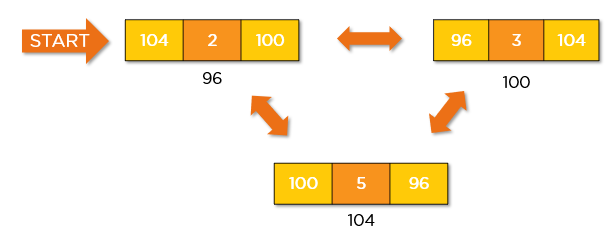
A [doubly linked list](https://www.simplilearn.com/tutorials/data-structure-tutorial/doubly-linked-list) is a bi-directional linked list. So, you can traverse it in both directions. Unlike singly linked lists, its nodes contain one extra pointer called the previous pointer. This pointer points to the previous node.

3]  [circular linked list](https://www.simplilearn.com/tutorials/data-structure-tutorial/singly-linked-list)



A [circular Linked list](https://www.simplilearn.com/tutorials/data-structure-tutorial/circular-linked-list) is a unidirectional linked list. So, you can traverse it in only one direction. But this type of linked list has its last node pointing to the head node. So while traversing, you need to be careful and stop traversing when you revisit the head node.

3]  [circular doubly linked list](https://www.simplilearn.com/tutorials/data-structure-tutorial/singly-linked-list)



A circular doubly linked list is a mixture of a doubly linked list and a circular linked list. Like the doubly linked list, it has an extra pointer called the previous pointer, and similar to the circular linked list, its last node points at the head node. This type of linked list is the bi-directional list. So, you can traverse it in both directions.

**6.List real time applications of Linked List.**

1] Image viewer – Previous and next images are linked, hence can be accessed by next and previous button.

2] Previous and next page in web browser

3] Music Player – Songs in music player are linked to previous and next song. you can play songs either from starting or ending of the list.

4] Redo and undo functionality. (doubly LL)

**7.Mention some drawbacks of the linked list.**

**1] Memory usage:** More memory is required in the linked list as compared to an array. Because in a linked list, a [pointer](https://www.geeksforgeeks.org/pointers-in-c-and-c-set-1-introduction-arithmetic-and-array/) is also required to store the address of the next element and it requires extra memory for itself.

**2] Traversal:** In a [Linked list traversal](https://www.geeksforgeeks.org/recursive-insertion-and-traversal-linked-list/) is more time-consuming as compared to an array. Direct access to an element is not possible in a linked list as in an array by index. For example, for accessing a node at position n, one has to traverse all the nodes before it.

**3] Reverse Traversing:** In a singly linked list reverse traversing is not possible, but in the case of a [doubly-linked list](https://www.geeksforgeeks.org/doubly-linked-list/), it can be possible as it contains a pointer to the previously connected nodes with each node. For performing this extra memory is required for the back pointer hence, there is a wastage of memory.

**4] Random Access:** Random access is not possible in a linked list due to its [dynamic memory allocation](https://www.geeksforgeeks.org/what-is-dynamic-memory-allocation/).

**8. List some advantages of Linked List**

**1] Dynamic data structure:** A linked list is a dynamic arrangement so it can grow and shrink at runtime by allocating and [deallocating memory](https://www.geeksforgeeks.org/how-to-deallocate-memory-without-using-free-in-c/). So there is no need to give the initial size of the linked list.

**2] No memory wastage:** In the Linked list, efficient memory utilization can be achieved since the size of the linked list increase or decrease at run time so there is no memory wastage and there is no need to pre-allocate the memory.

**3] Implementation:** Linear data structures like stack and queues are often easily implemented using a linked list.

**4] Insertion and Deletion Operations:** Insertion and deletion operations are quite easier in the linked list. There is no need to shift elements after the insertion or deletion of an element only the address present in the next pointer needs to be updated.

**9. Which data structures can we implement using linked list?** **A stack data structure &Queue**

**10. How can we implement stack and queue using linked list?** **With help of node**

11.How to write a program to create heterogeneous linked list?

12. Mention a package that is used for Linked List class in Java.

13.Mention some interfaces implemented by Linked List in Java.

14.How will you partition linked list

15. How to write a program to reverse linked list?

16. How can we reverse linked list using single pointer? Write a program

17. How to detect whether linked list is linear or circular