26|04|21
Lec-04

Chp- 03 - Linked List

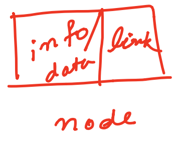
→ defn and mem. mgmt. ✓

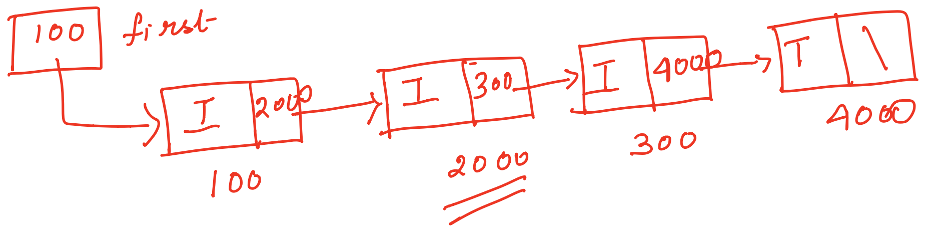
$$\begin{tikzcd} & \text{List ADT} \\ \text{ADT} \arrow[ur] \arrow[r] \arrow[dr] & \text{stack ADT} \\ & \text{Queue ADT} \end{tikzcd}$$

→ single L. List
- Traversing
- Counting
- searching
- Insertion (04 algos)
- deletion (03 algos)
- Reversal

→ Implementation is practically done using arrays or linked list.

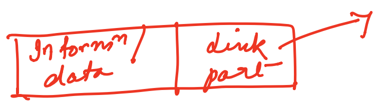
Linked List :



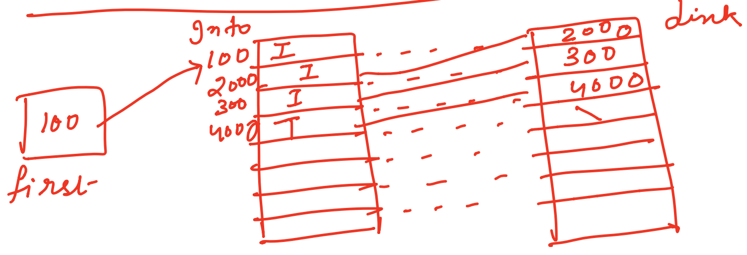


defn :
A linked list / or a one-way list is a linear collection of data elements called Nodes where the linear order is given by pointer.

Each node has 2 parts :



Representn. of L. List in memory :

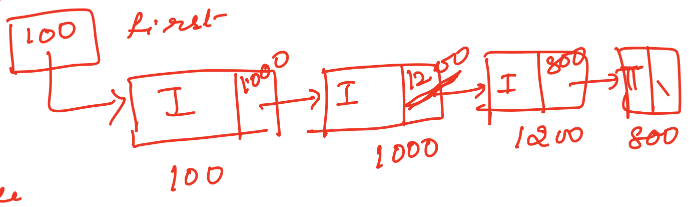


Info and link are two parallel arrays and first is a pointer which keeps the address of first node and the last node link part has a NULL value.

Single linked list :

1) Traversal :

```
1. set ptr = first
2. repeat steps 3 & 4 while
       ptr ≠ NULL
3.     write info (ptr)
4.     ptr = link (ptr)
5.     return.
```



ptr
~~100~~
~~1000~~
~~1200~~
~~800~~
NULL

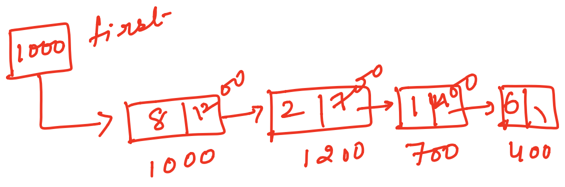
I
I
I
I

2) Count :

```
1. set ptr = first, Num = 0
2. repeat steps 3 & 4 while ptr ≠ NULL
3.     Num = Num + 1
4.     ptr = link (ptr)
5. return.
```

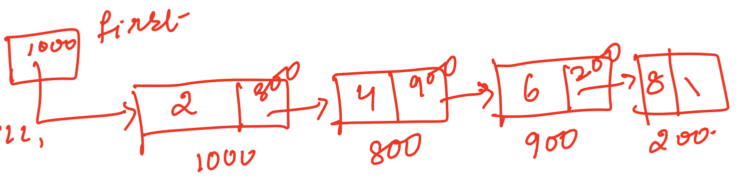
3) Search - unsort :

```
1) set ptr = first , doc = NULL
2) repeat step 03 while ptr ≠ NULL
3) if (info (ptr) = item ) then
       set doc = ptr
       return (doc)
   else ptr = link (ptr)
4) exit
```



doc	Item	ptr
~~NULL~~	01	~~1000~~
700		~~1200~~
		700

4) search - sort :



```
1.  set ptr = first, loc = NULL,
2.  repeat step 03 while ptr ≠ NULL
3.      if item > info (ptr) then
            ptr = link (ptr)
        ~~else~~ if item = info (ptr) then
            loc = ptr, return (loc)
        else loc = NULL, return (loc)
4.      return (doc)
5.  Exit
```

Item
08

ptr	loc
~~1000~~	~~NULL~~
~~800~~	200
~~900~~	
~~200~~	
NULL	

Item
10