Get the why of every steps → concept clear + not remembering

Page No.
Date 24-06-25

Linked List

(complete implementation in 3 pages) → Just fns.

- Dynamically allocatted, different from arraylist (doubles when full after copying earlier)
- Dont have indices, they are pointed via ref. variables

- Basic Declaration:- (DONE)

```
public class LL {
    class Node {
        Node next ;
        int val ;

        public Node (int val , Node next) {
            this.val = val ;
            this.next = next ;
        }
    }
    [const^n of val only
}
    public LL () {
        this. size = 0
    }
```

this.val → helps it telling about which obj. calls for it.

const^n of both → reason :- insert index fn.

const^n of val only → reason → used in Insert first fn., insert last

public LL () → initialize size of LL (not Node) with 0. ☆

after this → ~~define~~ declare Node head, tail 4 int size as private

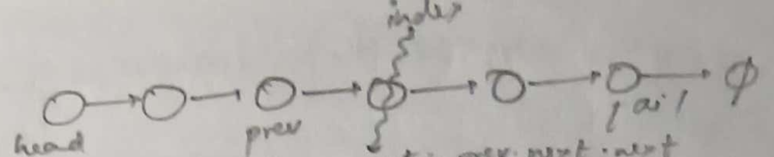
- Insert first , Insert last , Insert index
  Delete " Delete " Delete "
  Display

functions → easy 4 quick.

- Insert first → void return type → arg. value only ✓

```
✓① public void Insert first (int val) {
✓②     Node node = new Node (val); → create node ~~of~~ 4 value passed
```

Now think youre inserting at start to pehle (starting mein) head to uske aage hi hoga na, phir uske bad assign kr denge to waha pe declare ~~krne ke baad~~ use head as:-



∴ prev.next = prev.next.next

③. node.next = head;

④. head = node; (assign node val to head, cz ye head hai ab).

null case : ⑤ if (tail == null) return tail = head; → assign that head to tail also.
↳ exception case after operation since socho iska hone se add mein kya farak, aur jab value milega tabhi na assign karenge.

⑥. size ++;

insert first.
summary :→ void return type, val as arg method → create node + pass value → next of node is head → node now became head → null ptr exception if tail is null ∴ assign new head to tail → size ++.

Insert last
Insert last → same fn. type.

① ~~same~~ If already null ptr. excep. ∴ insert first (val);

② else create node of value + [do same steps but node tail se linked hota hai + uske baad lagta hai]

③ ∴ tail.next = node;

④ now this is tail → ~~node = tail~~; → this will same as insert First → tail = node; (give the val of node to tail)

⑤ size ++ ;

Insert index → void return type ; arg both ~~node~~ + val, int index → jaha dalna hai

{Main file to basis se socho kaise use karenge bina index bataye.} obviously

① if index == 0 → insertfirst(val);
" " " size-1 → " last " ;

② Create temp node starting (initialize) from head → Node temp = head;

③ index tak pahuchao temp. ko → ek pehle → for (i = 1 to index) → since ~~proceed~~ after head ∴ 1
temp = temp.next;

④ You know we made a count ≥ node of val + Node → Create node + pass value + temp.next, ∵ temp jaha pahucha & uske aage rakhna hai

{~~also can use as temp till i==index tak pahuchate aur isme sedha temp pass krte.~~}

★ ∵ when i just one less than index tab temp = temp.next ~~ho gya jo index pe pahuch gya already~~

⑤ put node value on temp.next → temp.next = node;

⑥ size ++;

insert index
Summary → pehle fn. banao void of val + index → if 0 or size use made fn. → temp node iteration from head to index → create node + pass val + ~~[illegible]~~ temp.next → put node in temp.next → size ++

✓* Delete first :- void type & obviously no arg. passed.

* ① get head value → int val = head.value ;
  val ↙ variablename
  value ↳ based on Node class data member
  ↓
  getting this value just to show which one deleted (no need)
  ↓
  ∴ needs return type int

Delete first

① . head = head.next ****** → gets deleted
② null ptr. excep. if head == null ∴ tail = null
③ size --

✓* Use case (can use in insert index too) → make a get fn.

Get O(n)

→ return type node & arg. int index (till where)
① Create node initialized with head.
② for (i = 0 to index) → [since node is item of LL (cant understand) ?]
   learn
   node = node.next ;
③ return node.

✓* Delete last → same fn. type as delete first.

Delete last

① if (size <= 1 → return) → remember this null ptr exception !!!
② create secondlast naam ka node and initialize it as get (size - 2);
③ assign that secondlast as tail → tail = secondlast ✓
④ tail.next = null → parampara + pratistha + anushashan
⑤ size -- ;

✓* Delete index → arg: int index

Delete idx

① if index == 0 → deletefirst ();
   " " " (size - 1) → " last "
② Node prev = get (index - 1); → waha pe pahucho
③ prev.next = prev.next.next → udao value fit
④ size -- ;

✓* Display → void, no arg

Display

① Create temp from head
② while (temp != null) {
      sout (temp.value + " → ");
   } temp = temp.next → ∵ at the end null bhi dikhana tha
   nhi to we would've use → while (temp.next != null)