Dt : 11/10/2023

define ListIterator<T>?

=>ListIterator<T> is an interface from java.util package and which is used to retrieve elements in both directions,which means retrieving elements in forward and backward.

=>ListIterator<T> is extended from Iterator<T> and which works only on List<E> Objects.

=>The following are some important methods of ListIterator<T>:

```
public abstract boolean hasNext();
public abstract E next();
public abstract boolean hasPrevious();
public abstract E previous();
public abstract int nextIndex();
public abstract int previousIndex();
public abstract void remove();
public abstract void set(E);
public abstract void add(E);
```

=>we use listIterator() or listIterator(int) to create implementation object for ListIterator<T> interface.

syntax:

```
ListIterator<Class_name> ob1 = ob.listIterator();
ListIterator<Class_name> ob2 = ob.listIterator(index_val);
```

==========

Limitation of ArrayList<E>:

=>when we perform add(index) operation on ArrayList<E> the elements are moved backward and when we perform remove(index) the elements are moved forward,if add() and remove() operations are performed more number of times then the performance of an application will be degraded,because the execution time is consumed to move elements forward and backward.

=>In realtime ArrayList<E> is used in the applications where we have less number of add() and remove() operations.

Ex:

Customer_Logins of Applications

Note:

=>Limitation of ArrayList<E> can be Overcomed used LinkedList<E>.

==========

*imp

(b)LinkedList<E>:

=>LinkedList<E> is an interface from java.util package and which organizes elements in NonSequence.

=>LinkedList<E> is also a NonSynchronized class.

syntax:

LinkedList<Class_name> ob = new LinkedList<Class_name>();

=>In LinkedList<E> the elements are available in the form of "Nodes".

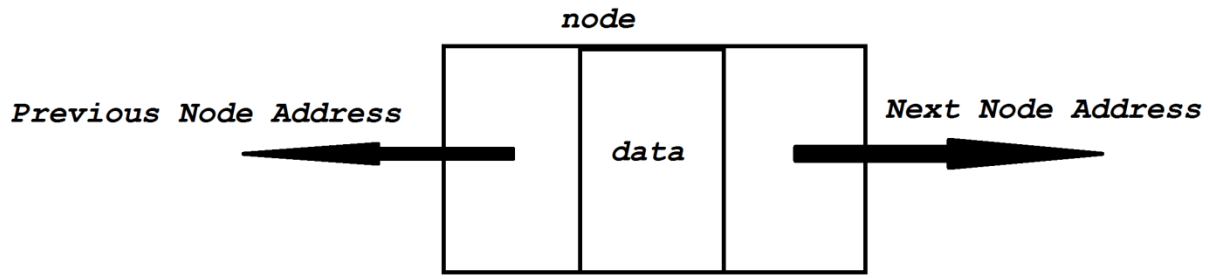
=>The LinkedList<E> node internally having the following partitions:

(i)Previous Node Address

(ii)Data

(iii)Next Node Address

Diagram:



Program : DemoList7.java

```
package p2;
import java.util.*;
public class DemoList7 {
     @SuppressWarnings("removal")
     public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        try(s;){
         try {
```

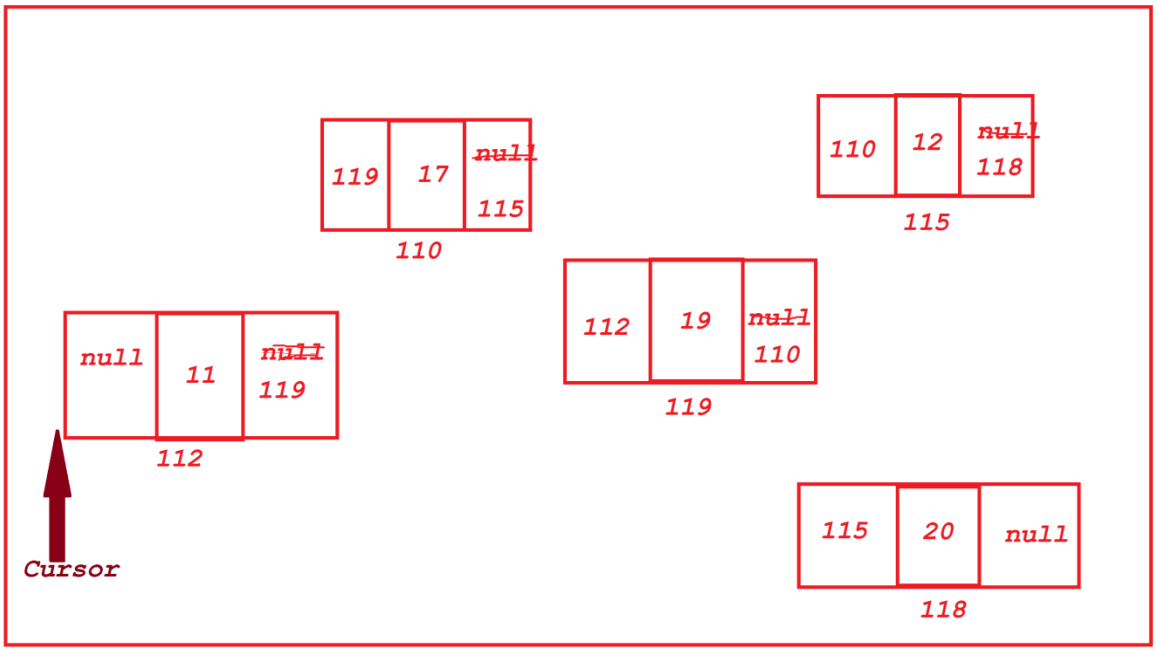
```
            LinkedList<Integer> ob=new
LinkedList<Integer>();
            System.out.println("Enter the number of eles
to be added to LinkedList:");
            int n = s.nextInt();
            System.out.println("Enter "+n+"
elements....");
            for(int i=1;i<=n;i++)
            {
                ob.add(new Integer(s.nextInt()));
            }
            System.out.println("====ListList<E>===");
            System.out.println(ob.toString());
            System.out.println("----ListIterator<E>----
");
            ListIterator<Integer> ob1 =
ob.listIterator();
            System.out.print("Forward : ");
            while(ob1.hasNext()) {
                System.out.print(ob1.next()+" ");
            }
            System.out.print("\nBackward : ");
            while(ob1.hasPrevious()) {
                System.out.print(ob1.previous()+" ");
            }

System.out.println("\n====listIterator(int)===");
            ListIterator<Integer> ob2 =
ob.listIterator(2);
            System.out.print("Forward : ");
            while(ob2.hasNext()) {
                System.out.print(ob2.next()+" ");
            }
            System.out.print("\nBackward : ");
            while(ob2.hasPrevious()) {
                System.out.print(ob2.previous()+" ");
            }
        }catch(Exception e) {e.printStackTrace();}
        }//end of try with resource
    }
```

```
}
```

o/p:

```
Enter the number of eles to be added to LinkedList:
5
Enter 5 elements....
11
19
17
12
20
====ListList<E>===
[11, 19, 17, 12, 20]
----ListIterator<E>----
Forward : 11 19 17 12 20
Backward : 20 12 17 19 11
====listIterator(int)===
Forward : 17 12 20
Backward : 20 12 17 19 11
```

==========



0x2

```
LinkedList<Integer> ob=new LinkedList<Integer>();
```

This ref will hold
UnLimited Integer Objects