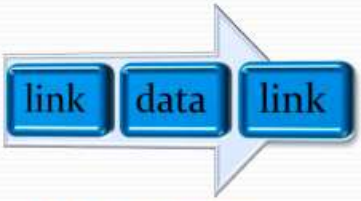
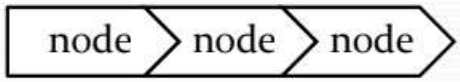
LinkedList Class:

- LinkedList is available since jdk1.2V.
- It allows duplicates, null & insertion order is maintained.
- Default capacity when creating an LinkedList is 0.
- In linked list elements are stored in the form of nodes.
- Each node will have three fields, the data field contains data and the link fields contain references to previous and next nodes.
- It occupies more memory than ArrayList and Construction time is also high. **Syntax: LinkedList ll=new LinkedList();**



Node Structure



Linked List

LinkedList Methods

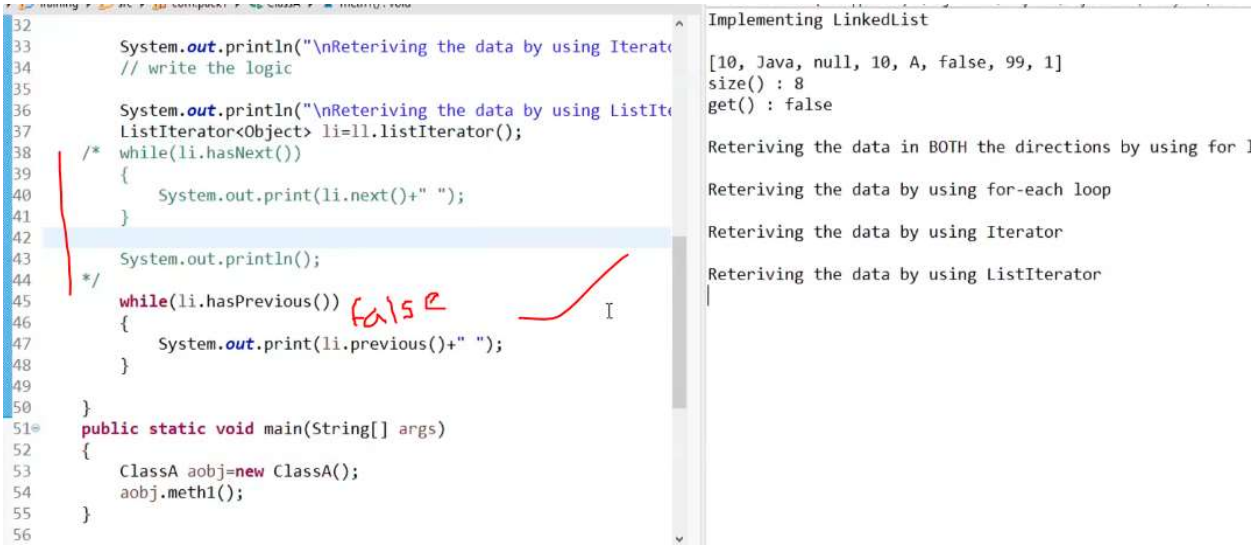
Methods	Description
Object getFirst();	Returns the first element in this list
Object getLast();	Returns the last element in this list
Object removeFirst();	Removes and returns the first element from this list
Object removeLast();	Removes and returns the last element from this list.
void addFirst(Element e);	Inserts the specified element at the beginning of this list.
addLast(Element e);	Appends the specified element to the end of this list.

```java
public class ClassA
{
    void meth1()
    {
        System.out.println("Implementing LinkedList\n");

        LinkedList<Object> ll=new LinkedList<Object>();

        ll.add(10); // Insertion order is
        ll.add("Java");// Heterogeneous data is
        ll.add(null);// null value is
        ll.add(10);// Duplicates are Allowed
        ll.add('A'); // It is available from Java 1.2v
        ll.add(false);// Its default capacity is 0  // In linkedlist elements will be stored in  the form of NODES
        ll.add(99);// Its size increases  by DOUBLE
        ll.add(1); // It is NOT synchronized

        System.out.println(ll);
    }
    public static void main(String[] args)
    {
        ClassA aobj=new ClassA();
        aobj.meth1();
    }
```

```java

        System.out.println("\nReteriving the data by using Iterat
        // write the logic

        System.out.println("\nReteriving the data by using ListIt
        ListIterator<Object> li=ll.listIterator();
        while(li.hasNext())
        {
            System.out.print(li.next()+" ");
        }

        System.out.println();

        while(li.hasPrevious())
        {
            System.out.print(li.previous()+" ");
        }

    }
    public static void main(String[] args)
    {
        ClassA aobj=new ClassA();
        aobj.meth1();
    }
```

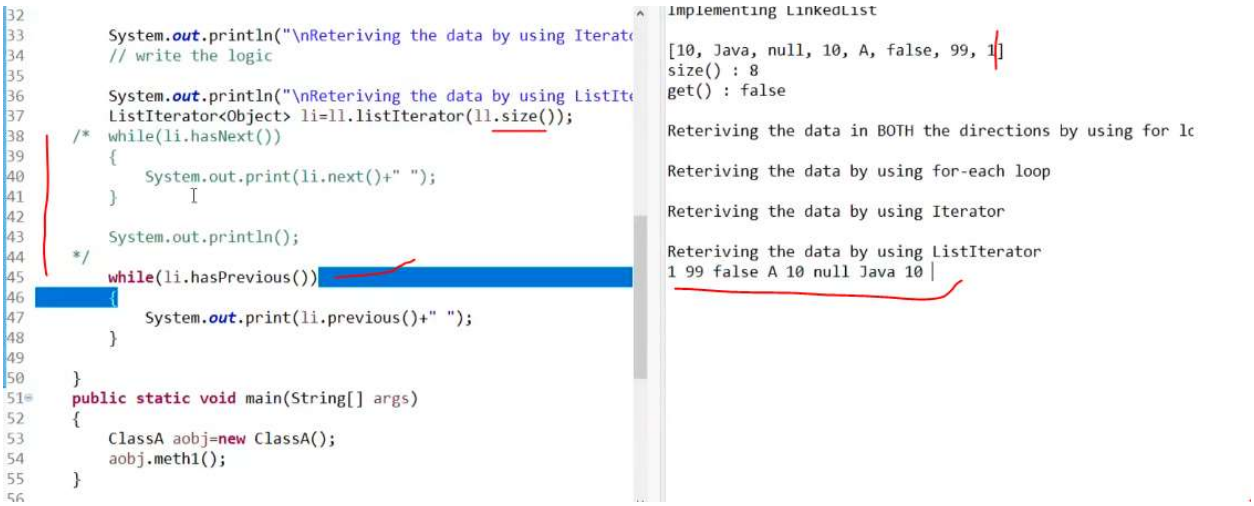
```
Implementing LinkedList

[10, Java, null, 10, A, false, 99, 1]
size() : 8
get() : false

Reteriving the data in BOTH the directions by using for lo

Reteriving the data by using for-each loop

Reteriving the data by using Iterator

Reteriving the data by using ListIterator
10 Java null 10 A false 99 1
1 99 false A 10 null Java 10
```

After the end of while loop over cursor is after 1 in above case



```
        System.out.println("\nReteriving the data by using Iterat
        // write the logic

        System.out.println("\nReteriving the data by using ListIt
        ListIterator<Object> li=ll.listIterator();
    /*  while(li.hasNext())
        {
            System.out.print(li.next()+" ");
        }

        System.out.println();
    */
        while(li.hasPrevious())
        {
            System.out.print(li.previous()+" ");
        }

    }
    public static void main(String[] args)
    {
        ClassA aobj=new ClassA();
        aobj.meth1();
    }
```

```
Implementing LinkedList

[10, Java, null, 10, A, false, 99, 1]
size() : 8
get() : false

Reteriving the data in BOTH the directions by using for l

Reteriving the data by using for-each loop

Reteriving the data by using Iterator

Reteriving the data by using ListIterator
```



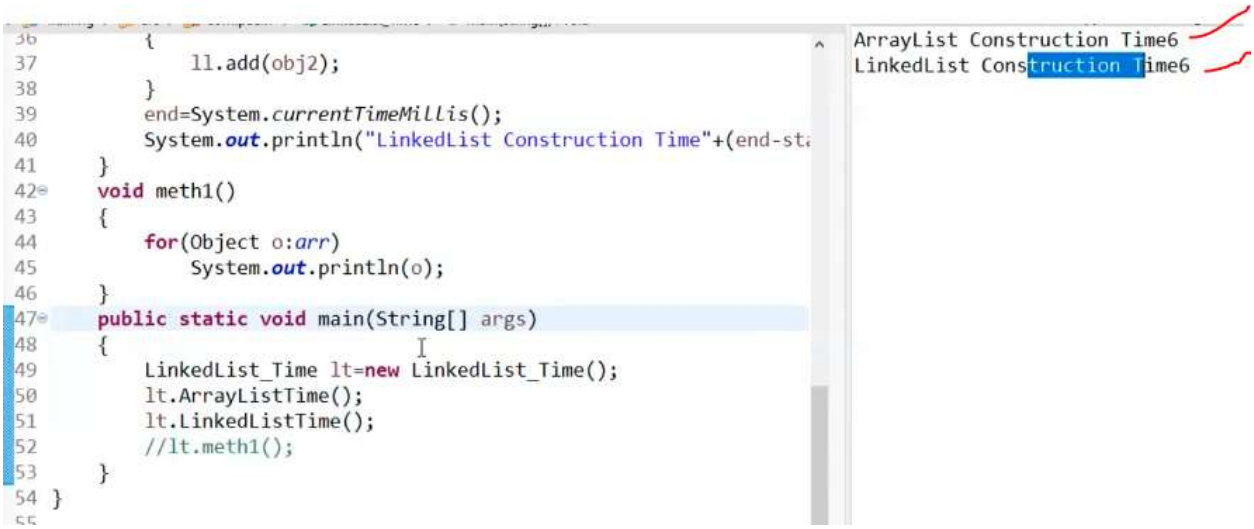
```
        System.out.println("\nReteriving the data by using Iterat
        // write the logic

        System.out.println("\nReteriving the data by using ListIt
        ListIterator<Object> li=ll.listIterator(ll.size());
    /*  while(li.hasNext())
        {
            System.out.print(li.next()+" ");
        }

        System.out.println();
    */
        while(li.hasPrevious())
        {
            System.out.print(li.previous()+" ");
        }

    }
    public static void main(String[] args)
    {
        ClassA aobj=new ClassA();
        aobj.meth1();
    }
```

```
Implementing LinkedList

[10, Java, null, 10, A, false, 99, 1]
size() : 8
get() : false

Reteriving the data in BOTH the directions by using for lo

Reteriving the data by using for-each loop

Reteriving the data by using Iterator

Reteriving the data by using ListIterator
1 99 false A 10 null Java 10
```

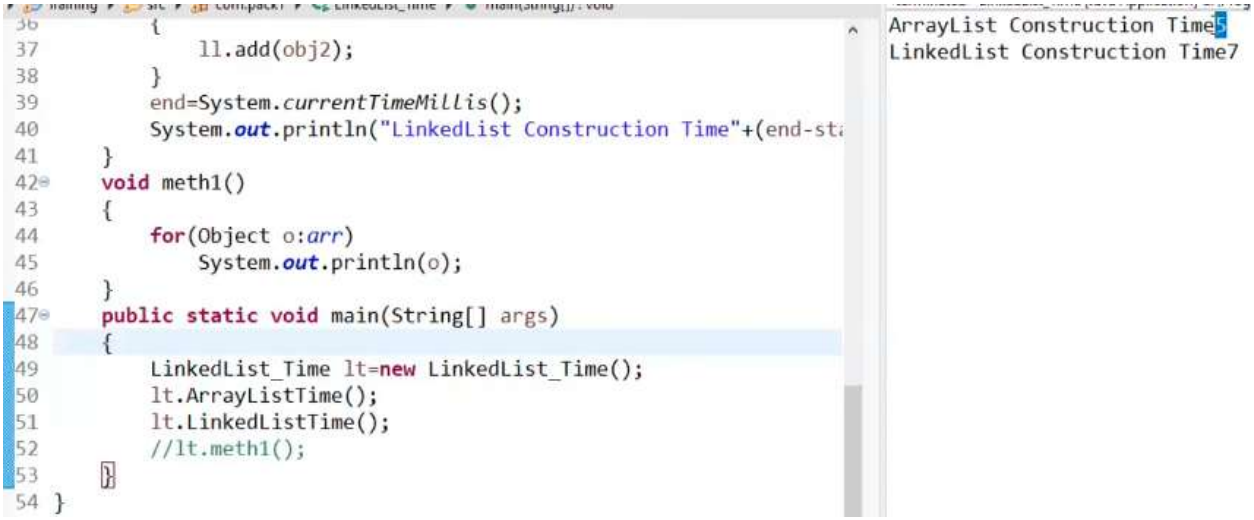
```java
package com.pack1;

import java.util.LinkedList;
import java.util.ListIterator;

public class ClassA
{
    void meth1()
    {
        System.out.println("Implementing LinkedList\n");

        LinkedList<Object> ll=new LinkedList<Object>();

        ll.add(10); // Insertion order is maintained
        ll.add("Java");// Heterogeneous data is allowed
        ll.add(null);// null value is  allowed
        ll.add(10);// Duplicates are Allowed
        ll.add('A'); // It is available from Java 1.2v
        ll.add(false);// Its default capacity is 0  // In linkedlist elements will be stored in  the form of NODES
        ll.add(99);// Its size increases  by DOUBLE
        ll.add(1); // It is NOT synchronized

        System.out.println(ll);
        System.out.println("size() : "+ll.size());
        System.out.println("get() : "+ll.get(5));


        System.out.println("\nReteriving the data in BOTH the directions by using for loop");
        // write the logic

        System.out.println("\nReteriving the data by using for-each loop");
        // write the logic

        System.out.println("\nReteriving the data by using Iterator");
        // write the logic

        System.out.println("\nReteriving the data by using ListIterator");
        ListIterator<Object> li=ll.listIterator(ll.size());
    /*  while(li.hasNext())
        {
            System.out.print(li.next()+" ");
        }

        System.out.println();
    */


        while(li.hasPrevious())
        {
            System.out.print(li.previous()+" ");
        }
    }
    void meth2()
    {
        // TASK : WAP to pass userdefined class Object into linkedlist & reterive it back.
    }
    public static void main(String[] args)
    {
        ClassA aobj=new ClassA();
        aobj.meth1();
    }
}
```

```java
import java.util.ArrayList;
import java.util.LinkedList;

public class LinkedList_Time
{
    private static Object arr[];

    static
    {
        arr=new Object[100000];
        for(int i=0;i<arr.length;i++) // using for loop to pass 100000 Objects
            arr[i]=new Object();
    }
    void ArrayListTime()
    {
        long start;
        long end;

        ArrayList<Object> al=new ArrayList<Object>();
        start=System.currentTimeMillis();//Its a static method gives the current system time in long millisec
        for(Object obj1:arr)
        {
            al.add(obj1);
        }
        end=System.currentTimeMillis();
        System.out.println("ArrayList Construction Time"+(end-start));
    }
    void LinkedListTime()
    {
        long start, end;
        LinkedList <Object>ll=new LinkedList<Object>();
        start=System.currentTimeMillis();
        for(Object obj2:arr)
        {
            ll.add(obj2);
        }
        end=System.currentTimeMillis();
        System.out.println("LinkedList Construction Time"+(end-start));
    }
    void meth1()
    {
        for(Object o:arr)
            System.out.println(o);
    }
```

```
47      public static void main(String[] args)
48      {
49          LinkedList_Time lt=new LinkedList_Time();
50          lt.ArrayListTime();
51          lt.LinkedListTime();
52          //lt.meth1();
53      }
54  }
```



We are getting the same time, but LinkedList should take more time than array List because data transfer in nodes execute the program for one more time



1) What is a Java LinkedList and how is it different from an ArrayList?
2) How do you add an element to the beginning of a Java LinkedList?
3) How do you add an element to the end of a Java LinkedList?
4) How do you remove the first element of a Java LinkedList?
5) How do you remove the last element of a Java LinkedList?
6) How do you iterate through a Java LinkedList?
7) How do you get the size of a Java LinkedList?
8) How do you check if a Java LinkedList contains a specific element?
9)How do you get the first element of a Java LinkedList?
10)How do you get the last element of a Java LinkedList?

Write a Java program that creates a LinkedList of strings and then removes all strings that contain the letter "a". Finally, print out the LinkedList to the console.