Appendix A

Linked List All-DU pairs tests for the add() method definition:

```
1. public void testobj_D1U1() {
   LL1= new LinkedList();
   LL1.add(0, new Integer(7));
   assertTrue(LL1.contains(new Integer(7)));
}

2. public void testobj_D1U2() {
   LL1= new LinkedList();
   LL1.add(new String("abc"));
   assertTrue(LL1.contains(new String("abc")));
}

3. public void testobj_D1U3() {
   LL1= new LinkedList();
   LL1.addAll(collecshan);
   assertTrue(LL1.indexOf(new Integer(3))==2);
}

4. public void testobj_D1U4() {
   LL1= new LinkedList();
   LL1.addAll(0, collecshan);
   assertTrue(LL1.indexOf(new Integer(3))==2);
}

5. public void testobj_D1U5() {
   LL1= new LinkedList();
   LL1.addFirst(new Integer(4));
   assertTrue(LL1.indexOf(new Integer(4))==0);
}

6. public void testobj_D1U6() {
   LL1= new LinkedList();
   LL1.addLast(new Integer(4));
   assertTrue(LL1.indexOf(new Integer(4))==0);
}

7. public void testobj_D1U7() {
   LL1= new LinkedList();
   LL1.clear();
   assertTrue(LL1.isEmpty());
}

8. public void testobj_D1U8() {
   LL1= new LinkedList();
   LinkedList obj=(LinkedList)LL1.clone();
```

```
   assertTrue(obj.equals(LL1));
}

9. public void testobj_D1U9() {
   LL1= new LinkedList();
   assertTrue(!LL1.contains(new Integer(1)));
}

10.     public void testobj_D1U10() {
   int excep=1;
   LL1= new LinkedList();
   try {
        LL1.get(0);
   } catch(IndexOutOfBoundsException i) {
        System.out.println("Exception");
        excep=0;
   }
   if(excep==1) assertTrue(false);
}

11.     public void testobj_D1U11() {
   int excep=1;
   LL1= new LinkedList();
   try {
   LL1.getFirst();
   } catch(NoSuchElementException i) {
   System.out.println("Exception");
   excep=0;
   }
   if(excep==1) assertTrue(false);
}

12.     public void testobj_D1U12() {
   int excep=1;
   LL1= new LinkedList();
   try {
   LL1.getLast();
   } catch(NoSuchElementException i) {
   System.out.println("Exception");
   excep=0;
   }
   if(excep==1) assertTrue(false);
}


13.     public void testobj_D1U13() {
   LL1= new LinkedList();
   assertTrue(LL1.indexOf(new Integer(3))==-1);
}

14.     public void testobj_D1U14() {
   LL1= new LinkedList();
```

```
   assertTrue(LL1.lastIndexOf(new Integer(3))==-1);
}

15.     public void testobj_D1U15() {
   LL1= new LinkedList();
   LL1.listIterator(0);
}

16.     public void testobj_D1U16() {
   int excep=1;
   LL1= new LinkedList();
   try {
   LL1.remove(0);
   } catch(IndexOutOfBoundsException i) {
   System.out.println("Exception");
   excep=0;
   }
   if(excep==1) assertTrue(false);
}

17.     public void testobj_D1U17() {
   LL1= new LinkedList();
   assertTrue(LL1.remove(new Integer(1))==false);
}


18.     public void testobj_D1U18() {
   int excep=1;
   LL1= new LinkedList();
   try {
   LL1.removeFirst();
   } catch(NoSuchElementException i) {
   System.out.println("Exception");
   excep=0;
   }
   if(excep==1) assertTrue(false);
}


19.     public void testobj_D1U19() {
   int excep=1;
   LL1= new LinkedList();
   try {
   LL1.removeLast();
   } catch(NoSuchElementException i) {
   System.out.println("Exception");
   excep=0;
   }
   if(excep==1) assertTrue(false);
}

20.     public void testobj_D1U20() {
```

```
            int excep=1;
            LL1= new LinkedList();
            try {
            LL1.set(0, new Integer(2));
            } catch(IndexOutOfBoundsException i) {
            System.out.println("Exception");
            excep=0;
            }
            if(excep==1) assertTrue(false);
         }

         21.     public void testobj_D1U21() {
            LL1= new LinkedList();
            assertTrue(LL1.size()==0);
         }

         22.     public void testobj_D1U22() {
            LL1= new LinkedList();
            LL1.toArray();
         }

         23.     public void testobj_D1U23() {
            Integer a[]=new Integer[10];
            LL1= new LinkedList();
            LL1.toArray(a);
}
```

LinkedList All-DU-pairs 665 tests:

```
import junit.framework.*;
import LinkedList;
import junit.extensions.*;
import java.util.NoSuchElementException;



public class TestLinkedList extends TestCase {


public TestLinkedList(String name) {
            super(name);
      }

LinkedList LL1, LL2, LL3, LL4, LL5, LL6, LL7, LL8,
LL9, LL10, LL11, LL12, LL13, LL14, LL15, LL16, LL17,
LL18, LL19, LL20, LL21, LL22, LL23, LL24, LL25,
collecshan, tempobj;
```

```
LinkedList lobjt1, lobjt2, lobjt3, lobjt4, lobjt5,
lobjt6, lobjt7, lobjt8, objt1;
Integer objt2, objt3, objt4, objt5, objt6, objt7,
objt8;

LinkedList lint1, lint2, lint3;
int int1, int2, int3;



public static void main(String[] args) {
     junit.textui.TestRunner.run (suite());
}

protected void setUp() {

     collecshan= new LinkedList();
     collecshan.add(new Integer(7));
     collecshan.add(new Integer(5));
     collecshan.add(new Integer(3));


//#####################objt defs start



//#########integer defs start



}

public static Test suite() {
     return new TestSuite(TestLinkedList.class);
}

public void testobj_D1U1() {
     LL1= new LinkedList();
     LL1.add(0, new Integer(7));
     assertTrue(LL1.contains(new Integer(7)));
}

public void testobj_D1U2() {
     LL1= new LinkedList();
     LL1.add(new String("abc"));
     assertTrue(LL1.contains(new String("abc")));
}

public void testobj_D1U3() {
     LL1= new LinkedList();
     LL1.addAll(collecshan);
```

```
      assertTrue(LL1.indexOf(new Integer(3))==2);
}

public void testobj_D1U4() {
      LL1= new LinkedList();
      LL1.addAll(0, collecshan);
      assertTrue(LL1.indexOf(new Integer(3))==2);
}

public void testobj_D1U5() {
      LL1= new LinkedList();
      LL1.addFirst(new Integer(4));
      assertTrue(LL1.indexOf(new Integer(4))==0);
}

public void testobj_D1U6() {
      LL1= new LinkedList();
      LL1.addLast(new Integer(4));
      assertTrue(LL1.indexOf(new Integer(4))==0);
}

public void testobj_D1U7() {
      LL1= new LinkedList();
      LL1.clear();
      assertTrue(LL1.isEmpty());
}

public void testobj_D1U8() {
      LL1= new LinkedList();
      LinkedList obj=(LinkedList)LL1.clone();
      assertTrue(obj.equals(LL1));
}

public void testobj_D1U9() {
      LL1= new LinkedList();
      assertTrue(!LL1.contains(new Integer(1)));
}

public void testobj_D1U10() {
      int excep=1;
      LL1= new LinkedList();
      try {
      LL1.get(0);
      } catch(IndexOutOfBoundsException i) {
            System.out.println("Exception");
            excep=0;
      }
      if(excep==1) assertTrue(false);
}

public void testobj_D1U11() {
      int excep=1;
```

```
        LL1= new LinkedList();
        try {
        LL1.getFirst();
        } catch(NoSuchElementException i) {
              System.out.println("Exception");
              excep=0;
        }
        if(excep==1) assertTrue(false);
  }

  public void testobj_D1U12() {
        int excep=1;
        LL1= new LinkedList();
        try {
        LL1.getLast();
        } catch(NoSuchElementException i) {
              System.out.println("Exception");
              excep=0;
        }
        if(excep==1) assertTrue(false);
  }

  public void testobj_D1U13() {
        LL1= new LinkedList();
        assertTrue(LL1.indexOf(new Integer(3))==-1);
  }

  public void testobj_D1U14() {
        LL1= new LinkedList();
        assertTrue(LL1.lastIndexOf(new Integer(3))==-1);
  }

  public void testobj_D1U15() {
        LL1= new LinkedList();
        LL1.listIterator(0);
  }

  public void testobj_D1U16() {
        int excep=1;
        LL1= new LinkedList();
        try {
        LL1.remove(0);
        } catch(IndexOutOfBoundsException i) {
              System.out.println("Exception");
              excep=0;
        }
        if(excep==1) assertTrue(false);
  }

  public void testobj_D1U17() {
        LL1= new LinkedList();
        assertTrue(LL1.remove(new Integer(1))==false);
```

```
}


public void testobj_D1U18() {
      int excep=1;
      LL1= new LinkedList();
      try {
      LL1.removeFirst();
      } catch(NoSuchElementException i) {
            System.out.println("Exception");
            excep=0;
      }
      if(excep==1) assertTrue(false);
}

public void testobj_D1U19() {
      int excep=1;
      LL1= new LinkedList();
      try {
      LL1.removeLast();
      } catch(NoSuchElementException i) {
            System.out.println("Exception");
            excep=0;
      }
      if(excep==1) assertTrue(false);
}

public void testobj_D1U20() {
      int excep=1;
      LL1= new LinkedList();
      try {
      LL1.set(0, new Integer(2));
      } catch(IndexOutOfBoundsException i) {
            System.out.println("Exception");
            excep=0;
      }
      if(excep==1) assertTrue(false);
}

public void testobj_D1U21() {
      LL1= new LinkedList();
      assertTrue(LL1.size()==0);
}

public void testobj_D1U22() {
      LL1= new LinkedList();
      LL1.toArray();
}

public void testobj_D1U23() {
      Integer a[]=new Integer[10];
      LL1= new LinkedList();
```

```
        LL1.toArray(a);
    }


    //##############2ND OBJ DEF USES START#########

    public void testobj_D2U1() {
        collecshan= new LinkedList();
        collecshan.add(new Integer(7));
        collecshan.add(new Integer(5));
        collecshan.add(new Integer(3));
        LL2= new LinkedList(collecshan);
        LL2.add(3, new Integer(1));
        assertTrue(LL2.indexOf(new Integer(1))==3);
    }

    public void testobj_D2U2() {
        collecshan= new LinkedList();
        collecshan.add(new Integer(7));
        collecshan.add(new Integer(5));
        collecshan.add(new Integer(3));
        LL2= new LinkedList(collecshan);
        LL2.add(new String("abc"));
        assertTrue(LL2.indexOf(new String("abc"))==3);
    }

    public void testobj_D2U3() {
        collecshan= new LinkedList();
        collecshan.add(new Integer(7));
        collecshan.add(new Integer(5));
        collecshan.add(new Integer(3));
        LL2= new LinkedList(collecshan);
        LL2.addAll(collecshan);
        assertTrue(LL2.lastIndexOf(new Integer(3))==5);
    }

    public void testobj_D2U4() {
        collecshan= new LinkedList();
        collecshan.add(new Integer(7));
        collecshan.add(new Integer(5));
        collecshan.add(new Integer(3));
        LL2= new LinkedList(collecshan);
        LL2.addAll(3, collecshan);
        assertTrue(LL2.indexOf(new Integer(3))==2);
    }

    public void testobj_D2U5() {
        collecshan= new LinkedList();
        collecshan.add(new Integer(7));
        collecshan.add(new Integer(5));
        collecshan.add(new Integer(3));
        LL2= new LinkedList(collecshan);
```

```
        LL2.addFirst(new Integer(4));
        assertTrue(LL2.indexOf(new Integer(4))==0);
        assertTrue(LL2.indexOf(new Integer(7))==1);
    }

    public void testobj_D2U6() {
        collecshan= new LinkedList();
        collecshan.add(new Integer(7));
        collecshan.add(new Integer(5));
        collecshan.add(new Integer(3));
        LL2= new LinkedList(collecshan);
        LL2.addLast(new Integer(4));
        assertTrue(LL2.indexOf(new Integer(4))==3);
    }

    public void testobj_D2U7() {
        collecshan= new LinkedList();
        collecshan.add(new Integer(7));
        collecshan.add(new Integer(5));
        collecshan.add(new Integer(3));
        LL2= new LinkedList(collecshan);
        LL2.clear();
        assertTrue(LL2.isEmpty());
    }

    public void testobj_D2U8() {
        collecshan= new LinkedList();
        collecshan.add(new Integer(7));
        collecshan.add(new Integer(5));
        collecshan.add(new Integer(3));
        LL2= new LinkedList(collecshan);
        LinkedList obj=(LinkedList)LL2.clone();
        assertTrue(obj.contains(new Integer(7)));
    }

    public void testobj_D2U9() {
        collecshan= new LinkedList();
        collecshan.add(new Integer(7));
        collecshan.add(new Integer(5));
        collecshan.add(new Integer(3));
        LL2= new LinkedList(collecshan);
        assertTrue(LL2.contains(new Integer(7)));
    }

    public void testobj_D2U10() {
        collecshan= new LinkedList();
        collecshan.add(new Integer(7));
        collecshan.add(new Integer(5));
        collecshan.add(new Integer(3));
        LL2= new LinkedList(collecshan);
        assertTrue(LL2.get(0).equals(new Integer(7)));
    }
```

```
public void testobj_D2U11() {
      collecshan= new LinkedList();
      collecshan.add(new Integer(7));
      collecshan.add(new Integer(5));
      collecshan.add(new Integer(3));
      LL2= new LinkedList(collecshan);
      assertTrue(LL2.getFirst().equals(new
Integer(7)));
}

public void testobj_D2U12() {
      collecshan= new LinkedList();
      collecshan.add(new Integer(7));
      collecshan.add(new Integer(5));
      collecshan.add(new Integer(3));
      LL2= new LinkedList(collecshan);
      assertTrue(LL2.getLast().equals(new Integer(3)));
}

public void testobj_D2U13() {
      collecshan= new LinkedList();
      collecshan.add(new Integer(7));
      collecshan.add(new Integer(5));
      collecshan.add(new Integer(3));
      LL2= new LinkedList(collecshan);
      assertTrue(LL2.indexOf(new Integer(3))==2);
}

public void testobj_D2U14() {
      collecshan= new LinkedList();
      collecshan.add(new Integer(7));
      collecshan.add(new Integer(5));
      collecshan.add(new Integer(3));
      LL2= new LinkedList(collecshan);
      assertTrue(LL2.lastIndexOf(new Integer(5))==1);
}

public void testobj_D2U15() {
      collecshan= new LinkedList();
      collecshan.add(new Integer(7));
      collecshan.add(new Integer(5));
      collecshan.add(new Integer(3));
      LL2= new LinkedList(collecshan);
      LL2.listIterator(0);
}

public void testobj_D2U16() {
      collecshan= new LinkedList();
      collecshan.add(new Integer(7));
      collecshan.add(new Integer(5));
      collecshan.add(new Integer(3));
```

```
        LL2= new LinkedList(collecshan);
        LL2.remove(0);
        assertTrue(!LL2.contains(new Integer(7)));
    }

    public void testobj_D2U17() {
        collecshan= new LinkedList();
        collecshan.add(new Integer(7));
        collecshan.add(new Integer(5));
        collecshan.add(new Integer(3));
        LL2= new LinkedList(collecshan);
        assertTrue(LL2.remove(new Integer(5))==true);
        assertTrue(LL2.indexOf(new Integer(3))==1);
    }


    public void testobj_D2U18() {
        collecshan= new LinkedList();
        collecshan.add(new Integer(7));
        collecshan.add(new Integer(5));
        collecshan.add(new Integer(3));
        LL2= new LinkedList(collecshan);
        LL2.removeFirst();
        assertTrue(LL2.indexOf(new Integer(3))==1);
    }

    public void testobj_D2U19() {
        collecshan= new LinkedList();
        collecshan.add(new Integer(7));
        collecshan.add(new Integer(5));
        collecshan.add(new Integer(3));
        LL2= new LinkedList(collecshan);
        LL2.removeLast();
        assertTrue(!LL2.contains(new Integer(3)));
    }

    public void testobj_D2U20() {
        collecshan= new LinkedList();
        collecshan.add(new Integer(7));
        collecshan.add(new Integer(5));
        collecshan.add(new Integer(3));
        LL2= new LinkedList(collecshan);
        LL2.set(0, new Integer(2));
        assertTrue(!LL2.contains(new Integer(7)));
    }

    public void testobj_D2U21() {
        collecshan= new LinkedList();
        collecshan.add(new Integer(7));
        collecshan.add(new Integer(5));
        collecshan.add(new Integer(3));
        LL2= new LinkedList(collecshan);
```

```
      assertTrue(LL2.size()==3);
}

public void testobj_D2U22() {
      collecshan= new LinkedList();
      collecshan.add(new Integer(7));
      collecshan.add(new Integer(5));
      collecshan.add(new Integer(3));
      LL2= new LinkedList(collecshan);
      LL2.toArray();
}

public void testobj_D2U23() {
      Integer a[]=new Integer[10];
      collecshan= new LinkedList();
      collecshan.add(new Integer(7));
      collecshan.add(new Integer(5));
      collecshan.add(new Integer(3));
      LL2= new LinkedList(collecshan);
      LL2.toArray(a);
}


//##############3RD OBJ DEF USES START#########

public void testobj_D3U1() {
      LL3= new LinkedList();
      LL3.add(0, new String("abc"));
      LL3.add(1, new Integer(1));
      assertTrue(LL3.indexOf(new Integer(1))==1);
}

public void testobj_D3U2() {
      LL3= new LinkedList();
      LL3.add(0, new String("abc"));
      LL3.add(new String("abc"));
      assertTrue(LL3.lastIndexOf(new
String("abc"))==1);
}

public void testobj_D3U3() {
      LL3= new LinkedList();
      LL3.add(0, new String("abc"));
      LL3.addAll(collecshan);
      assertTrue(LL3.lastIndexOf(new Integer(3))==3);
}

public void testobj_D3U4() {
      LL3= new LinkedList();
      LL3.add(0, new String("abc"));
      LL3.addAll(1, collecshan);
      assertTrue(LL3.indexOf(new Integer(3))==3);
```

```
}

public void testobj_D3U5() {
      LL3= new LinkedList();
      LL3.add(0, new String("abc"));
      LL3.addFirst(new Integer(4));
      assertTrue(LL3.indexOf(new Integer(4))==0);
      assertTrue(LL3.indexOf(new String("abc"))==1);
}

public void testobj_D3U6() {
      LL3= new LinkedList();
      LL3.add(0, new String("abc"));
      LL3.addLast(new Integer(4));
      assertTrue(LL3.indexOf(new Integer(4))==1);
}

public void testobj_D3U7() {
      LL3= new LinkedList();
      LL3.add(0, new String("abc"));
      LL3.clear();
      assertTrue(!LL3.contains(new String("abc")));
}

public void testobj_D3U8() {
      LL3= new LinkedList();
      LL3.add(0, new String("abc"));
      LinkedList obj=(LinkedList)LL3.clone();
      assertTrue(obj.contains(new String("abc")));
}

public void testobj_D3U9() {
      LL3= new LinkedList();
      LL3.add(0, new String("abc"));
      assertTrue(LL3.contains(new String("abc")));
}

public void testobj_D3U10() {
      LL3= new LinkedList();
      LL3.add(0, new String("abc"));
      assertTrue(LL3.get(0).equals(new String("abc")));
}

public void testobj_D3U11() {
      LL3= new LinkedList();
      LL3.add(0, new String("abc"));
      assertTrue(LL3.getFirst().equals(new
String("abc")));
}

public void testobj_D3U12() {
      LL3= new LinkedList();
```

```
      LL3.add(0, new String("abc"));
      assertTrue(LL3.getLast().equals(new
String("abc")));
}

public void testobj_D3U13() {
      LL3= new LinkedList();
      LL3.add(0, new String("abc"));
      assertTrue(LL3.indexOf(new Integer(3))==-1);
}

public void testobj_D3U14() {
      LL3= new LinkedList();
      LL3.add(0, new String("abc"));
      assertTrue(LL3.lastIndexOf(new
String("abc"))==0);
}

public void testobj_D3U15() {
      LL3= new LinkedList();
      LL3.add(0, new String("abc"));
      LL3.listIterator(0);
}

public void testobj_D3U16() {
      LL3= new LinkedList();
      LL3.add(0, new String("abc"));
      LL3.remove(0);
      assertTrue(!LL3.contains(new String("abc")));
}

public void testobj_D3U17() {
      LL3= new LinkedList();
      LL3.add(0, new String("abc"));
      assertTrue(LL3.remove(new String("abc"))==true);
      assertTrue(LL3.indexOf(new String("abc"))==-1);
}


public void testobj_D3U18() {
      LL3= new LinkedList();
      LL3.add(0, new String("abc"));
      LL3.removeFirst();
      assertTrue(!LL3.contains(new String("abc")));
}

public void testobj_D3U19() {
      LL3= new LinkedList();
      LL3.add(0, new String("abc"));
      LL3.removeLast();
      assertTrue(!LL3.contains(new String("abc")));
}
```

```
public void testobj_D3U20() {
      LL3= new LinkedList();
      LL3.add(0, new String("abc"));
      LL3.set(0, new Integer(2));
      assertTrue(LL3.contains(new Integer(2)));
      assertTrue(!LL3.contains(new String("abc")));
}

public void testobj_D3U21() {
      LL3= new LinkedList();
      LL3.add(0, new String("abc"));
      assertTrue(LL3.size()==1);
}

public void testobj_D3U22() {
      LL3= new LinkedList();
      LL3.add(0, new String("abc"));
      LL3.toArray();
}

public void testobj_D3U23() {
      int jj=0;
      Integer a[]=new Integer[10];
      LinkedList arr[]=new LinkedList[10];
      LL3= new LinkedList();
      LL3.add(0, new String("abc"));
      try {
      LL3.toArray(a);
      } catch(ArrayStoreException ase) {
            jj=1;
      }
      if(jj==0) assertTrue(false);
}


//##############4TH OBJ DEF USES START#########

public void testobj_D4U1() {
      LL4= new LinkedList();
      LL4.add(new Boolean(true));
      LL4.add(1, new Integer(1));
      assertTrue(LL4.indexOf(new Integer(1))==1);
}

public void testobj_D4U2() {
      LL4= new LinkedList();
      LL4.add(new Boolean(true));
      LL4.add(new String("abc"));
      assertTrue(LL4.lastIndexOf(new
String("abc"))==1);
}
```

```
public void testobj_D4U3() {
      LL4= new LinkedList();
      LL4.add(new Boolean(true));
      LL4.addAll(collecshan);
      assertTrue(LL4.lastIndexOf(new Integer(3))==3);
}

public void testobj_D4U4() {
      LL4= new LinkedList();
      LL4.add(new Boolean(true));
      LL4.addAll(1, collecshan);
      assertTrue(LL4.indexOf(new Integer(3))==3);
}

public void testobj_D4U5() {
      LL4= new LinkedList();
      LL4.add(new Boolean(true));
      LL4.addFirst(new Integer(4));
      assertTrue(LL4.indexOf(new Integer(4))==0);
      assertTrue(LL4.indexOf(new Boolean(true))==1);
}

public void testobj_D4U6() {
      LL4= new LinkedList();
      LL4.add(new Boolean(true));
      LL4.addLast(new Integer(4));
      assertTrue(LL4.indexOf(new Integer(4))==1);
}

public void testobj_D4U7() {
      LL4= new LinkedList();
      LL4.add(new Boolean(true));
      LL4.clear();
      assertTrue(!LL4.contains(new Boolean(true)));
}

public void testobj_D4U8() {
      LL4= new LinkedList();
      LL4.add(new Boolean(true));
      LinkedList obj=(LinkedList)LL4.clone();
      assertTrue(obj.contains(new Boolean(true)));
}

public void testobj_D4U9() {
      LL4= new LinkedList();
      LL4.add(new Boolean(true));
      assertTrue(LL4.contains(new Boolean(true)));
}

public void testobj_D4U10() {
      LL4= new LinkedList();
```

```
      LL4.add(new Boolean(true));
      assertTrue(LL4.get(0).equals(new Boolean(true)));
}

public void testobj_D4U11() {
      LL4= new LinkedList();
      LL4.add(new Boolean(true));
      assertTrue(LL4.getFirst().equals(new
Boolean(true)));
}

public void testobj_D4U12() {
      LL4= new LinkedList();
      LL4.add(new Boolean(true));
      assertTrue(LL4.getLast().equals(new
Boolean(true)));
}

public void testobj_D4U13() {
      LL4= new LinkedList();
      LL4.add(new Boolean(true));
      assertTrue(LL4.indexOf(new Integer(3))==-1);
}

public void testobj_D4U14() {
      LL4= new LinkedList();
      LL4.add(new Boolean(true));
      assertTrue(LL4.lastIndexOf(new
Boolean(true))==0);
}

public void testobj_D4U15() {
      LL4= new LinkedList();
      LL4.add(new Boolean(true));
      LL4.listIterator(0);
}

public void testobj_D4U16() {
      LL4= new LinkedList();
      LL4.add(new Boolean(true));
      LL4.remove(0);
      assertTrue(!LL4.contains(new Boolean(true)));
}

public void testobj_D4U17() {
      LL4= new LinkedList();
      LL4.add(new Boolean(true));
      assertTrue(LL4.remove(new Boolean(true))==true);
      assertTrue(LL4.indexOf(new Boolean(true))==-1);
}
```

```
public void testobj_D4U18() {
      LL4= new LinkedList();
      LL4.add(new Boolean(true));
      LL4.removeFirst();
      assertTrue(!LL4.contains(new Boolean(true)));
}

public void testobj_D4U19() {
      LL4= new LinkedList();
      LL4.add(new Boolean(true));
      LL4.removeLast();
      assertTrue(!LL4.contains(new Boolean(true)));
}

public void testobj_D4U20() {
      LL4= new LinkedList();
      LL4.add(new Boolean(true));
      LL4.set(0, new Integer(2));
      assertTrue(LL4.contains(new Integer(2)));
      assertTrue(!LL4.contains(new Boolean(true)));
}

public void testobj_D4U21() {
      LL4= new LinkedList();
      LL4.add(new Boolean(true));
      assertTrue(LL4.size()==1);
}

public void testobj_D4U22() {
      LL4= new LinkedList();
      LL4.add(new Boolean(true));
      LL4.toArray();
}

public void testobj_D4U23() {
      int jj=0;
      Integer a[]=new Integer[10];
      LinkedList arr[]=new LinkedList[10];
      LL4= new LinkedList();
      LL4.add(new Boolean(true));
      try {
      LL4.toArray(a);
      } catch(ArrayStoreException ase) {
            jj=1;
      }
      if(jj==0) assertTrue(false);
}


//##############5TH OBJ DEF USES START#########

public void testobj_D5U1() {
```

```
        LL5= new LinkedList();
        LL5.addAll(collecshan);
        LL5.add(3, new Integer(1));
        assertTrue(LL5.indexOf(new Integer(1))==3);
    }

    public void testobj_D5U2() {
        LL5= new LinkedList();
        LL5.addAll(collecshan);
        LL5.add(new String("abc"));
        assertTrue(LL5.indexOf(new String("abc"))==3);
    }

    public void testobj_D5U3() {
        LL5= new LinkedList();
        LL5.addAll(collecshan);
        LL5.addAll(collecshan);
        assertTrue(LL5.lastIndexOf(new Integer(3))==5);
    }

    public void testobj_D5U4() {
        LL5= new LinkedList();
        LL5.addAll(collecshan);
        LL5.addAll(3, collecshan);
        assertTrue(LL5.indexOf(new Integer(3))==2);
    }

    public void testobj_D5U5() {
        LL5= new LinkedList();
        LL5.addAll(collecshan);
        LL5.addFirst(new Integer(4));
        assertTrue(LL5.indexOf(new Integer(4))==0);
        assertTrue(LL5.indexOf(new Integer(7))==1);
    }

    public void testobj_D5U6() {
        LL5= new LinkedList();
        LL5.addAll(collecshan);
        LL5.addLast(new Integer(4));
        assertTrue(LL5.indexOf(new Integer(4))==3);
    }

    public void testobj_D5U7() {
        LL5= new LinkedList();
        LL5.addAll(collecshan);
        LL5.clear();
        assertTrue(LL5.isEmpty());
    }

    public void testobj_D5U8() {
        LL5= new LinkedList();
        LL5.addAll(collecshan);
```

```
        LinkedList obj=(LinkedList)LL5.clone();
        assertTrue(obj.contains(new Integer(7)));
    }

    public void testobj_D5U9() {
        LL5= new LinkedList();
        LL5.addAll(collecshan);
        assertTrue(LL5.contains(new Integer(7)));
    }

    public void testobj_D5U10() {
        LL5= new LinkedList();
        LL5.addAll(collecshan);
        assertTrue(LL5.get(0).equals(new Integer(7)));
    }

    public void testobj_D5U11() {
        LL5= new LinkedList();
        LL5.addAll(collecshan);
        assertTrue(LL5.getFirst().equals(new
    Integer(7)));
    }

    public void testobj_D5U12() {
        LL5= new LinkedList();
        LL5.addAll(collecshan);
        assertTrue(LL5.getLast().equals(new Integer(3)));
    }

    public void testobj_D5U13() {
        LL5= new LinkedList();
        LL5.addAll(collecshan);
        assertTrue(LL5.indexOf(new Integer(3))==2);
    }

    public void testobj_D5U14() {
        LL5= new LinkedList();
        LL5.addAll(collecshan);
        assertTrue(LL5.lastIndexOf(new Integer(5))==1);
    }

    public void testobj_D5U15() {
        LL5= new LinkedList();
        LL5.addAll(collecshan);
        LL5.listIterator(0);
    }

    public void testobj_D5U16() {
        LL5= new LinkedList();
        LL5.addAll(collecshan);
        LL5.remove(0);
        assertTrue(!LL5.contains(new Integer(7)));
```

```
}

public void testobj_D5U17() {
      LL5= new LinkedList();
      LL5.addAll(collecshan);
      assertTrue(LL5.remove(new Integer(5))==true);
      assertTrue(LL5.indexOf(new Integer(3))==1);
}


public void testobj_D5U18() {
      LL5= new LinkedList();
      LL5.addAll(collecshan);
      LL5.removeFirst();
      assertTrue(LL5.indexOf(new Integer(3))==1);
}

public void testobj_D5U19() {
      LL5= new LinkedList();
      LL5.addAll(collecshan);
      LL5.removeLast();
      assertTrue(!LL5.contains(new Integer(3)));
}

public void testobj_D5U20() {
      LL5= new LinkedList();
      LL5.addAll(collecshan);
      LL5.set(0, new Integer(2));
      assertTrue(!LL5.contains(new Integer(7)));
}

public void testobj_D5U21() {
      LL5= new LinkedList();
      LL5.addAll(collecshan);
      assertTrue(LL5.size()==3);
}

public void testobj_D5U22() {
      LL5= new LinkedList();
      LL5.addAll(collecshan);
      LL5.toArray();
}

public void testobj_D5U23() {
      Integer a[]=new Integer[10];
      LinkedList arr[]=new LinkedList[10];
      LL5= new LinkedList();
      LL5.addAll(collecshan);
      LL5.toArray(a);
}
```

```
//##############6TH OBJ DEF USES START#########

public void testobj_D6U1() {
      LL6= new LinkedList();
      LL6.add(0, collecshan);
      LL6.add(1, new Integer(1));
      assertTrue(LL6.indexOf(new Integer(1))==1);
}

public void testobj_D6U2() {
      LL6= new LinkedList();
      LL6.add(0, collecshan);
      LL6.add(new String("abc"));
      assertTrue(LL6.indexOf(new String("abc"))==1);
}

public void testobj_D6U3() {
      LL6= new LinkedList();
      LL6.add(0, collecshan);
      LL6.addAll(collecshan);
      assertTrue(LL6.lastIndexOf(new Integer(3))==3);
}

public void testobj_D6U4() {
      LL6= new LinkedList();
      LL6.add(0, collecshan);
      LL6.addAll(1, collecshan);
      assertTrue(LL6.lastIndexOf(new Integer(3))==3);
}

public void testobj_D6U5() {
      LL6= new LinkedList();
      LL6.add(0, collecshan);
      LL6.addFirst(new Integer(4));
      assertTrue(LL6.indexOf(new Integer(4))==0);
      assertTrue(LL6.indexOf(collecshan)==1);
}

public void testobj_D6U6() {
      LL6= new LinkedList();
      LL6.add(0, collecshan);
      LL6.addLast(new Integer(4));
      assertTrue(LL6.indexOf(new Integer(4))==1);
}

public void testobj_D6U7() {
      LL6= new LinkedList();
      LL6.add(0, collecshan);
      LL6.clear();
      assertTrue(LL6.isEmpty());
}
```

```
public void testobj_D6U8() {
      LL6= new LinkedList();
      LL6.add(0, collecshan);
      LinkedList obj=(LinkedList)LL6.clone();
      assertTrue(obj.contains(collecshan));
}

public void testobj_D6U9() {
      LL6= new LinkedList();
      LL6.add(0, collecshan);
      assertTrue(LL6.contains(collecshan));
}

public void testobj_D6U10() {
      LL6= new LinkedList();
      LL6.add(0, collecshan);
      assertTrue(LL6.get(0).equals(collecshan));
}

public void testobj_D6U11() {
      LL6= new LinkedList();
      LL6.add(0, collecshan);
      assertTrue(LL6.getFirst().equals(collecshan));
}

public void testobj_D6U12() {
      LL6= new LinkedList();
      LL6.add(0, collecshan);
      assertTrue(LL6.getLast().equals(collecshan));
}

public void testobj_D6U13() {
      LL6= new LinkedList();
      LL6.add(0, collecshan);
      assertTrue(LL6.indexOf(collecshan)==0);
}

public void testobj_D6U14() {
      LL6= new LinkedList();
      LL6.add(0, collecshan);
      assertTrue(LL6.lastIndexOf(collecshan)==0);
}

public void testobj_D6U15() {
      LL6= new LinkedList();
      LL6.add(0, collecshan);
      LL6.listIterator(0);
}

public void testobj_D6U16() {
      LL6= new LinkedList();
      LL6.add(0, collecshan);
```

```
        LL6.remove(0);
        assertTrue(!LL6.contains(new Integer(7)));
    }

    public void testobj_D6U17() {
        LL6= new LinkedList();
        LL6.add(0, collecshan);
        assertTrue(LL6.remove(collecshan)==true);
        assertTrue(LL6.indexOf(collecshan)==-1);
    }


    public void testobj_D6U18() {
        LL6= new LinkedList();
        LL6.add(0, collecshan);
        LL6.removeFirst();
        assertTrue(LL6.indexOf(collecshan)==-1);
    }

    public void testobj_D6U19() {
        LL6= new LinkedList();
        LL6.add(0, collecshan);
        LL6.removeLast();
        assertTrue(!LL6.contains(collecshan));
    }

    public void testobj_D6U20() {
        LL6= new LinkedList();
        LL6.add(0, collecshan);
        LL6.set(0, new Integer(2));
        assertTrue(!LL6.contains(new Integer(7)));
    }

    public void testobj_D6U21() {
        LL6= new LinkedList();
        LL6.add(0, collecshan);
        assertTrue(LL6.size()==1);
    }

    public void testobj_D6U22() {
        LL6= new LinkedList();
        LL6.add(0, collecshan);
        LL6.toArray();
    }

    public void testobj_D6U23() {
        int jj=0;
        Integer a[]=new Integer[10];
        LinkedList arr[]=new LinkedList[10];
        LL6= new LinkedList();
        LL6.add(0, collecshan);
        try {
```

```
      LL6.toArray(a);
      } catch(ArrayStoreException ase) {
      jj=1;
      }
      if(jj==0) assertTrue(false);
}

//##############7TH OBJ DEF USES START#########

public void testobj_D7U1() {
      LL7= new LinkedList();
      LL7.addFirst(new Integer(5));
      LL7.add(1, new Integer(1));
      assertTrue(LL7.indexOf(new Integer(1))==1);
}

public void testobj_D7U2() {
      LL7= new LinkedList();
      LL7.addFirst(new Integer(5));
      LL7.add(new String("abc"));
      assertTrue(LL7.lastIndexOf(new
String("abc"))==1);
}

public void testobj_D7U3() {
      LL7= new LinkedList();
      LL7.addFirst(new Integer(5));
      LL7.addAll(collecshan);
      assertTrue(LL7.lastIndexOf(new Integer(3))==3);
}

public void testobj_D7U4() {
      LL7= new LinkedList();
      LL7.addFirst(new Integer(5));
      LL7.addAll(1, collecshan);
      assertTrue(LL7.indexOf(new Integer(3))==3);
}

public void testobj_D7U5() {
      LL7= new LinkedList();
      LL7.addFirst(new Integer(5));
      LL7.addFirst(new Integer(4));
      assertTrue(LL7.indexOf(new Integer(4))==0);
      assertTrue(LL7.indexOf(new Integer(5))==1);
}

public void testobj_D7U6() {
      LL7= new LinkedList();
      LL7.addFirst(new Integer(5));
      LL7.addLast(new Integer(4));
      assertTrue(LL7.indexOf(new Integer(4))==1);
}
```

```
public void testobj_D7U7() {
      LL7= new LinkedList();
      LL7.addFirst(new Integer(5));
      LL7.clear();
      assertTrue(!LL7.contains(new Integer(5)));
}

public void testobj_D7U8() {
      LL7= new LinkedList();
      LL7.addFirst(new Integer(5));
      LinkedList obj=(LinkedList)LL7.clone();
      assertTrue(obj.contains(new Integer(5)));
}

public void testobj_D7U9() {
      LL7= new LinkedList();
      LL7.addFirst(new Integer(5));
      assertTrue(LL7.contains(new Integer(5)));
}

public void testobj_D7U10() {
      LL7= new LinkedList();
      LL7.addFirst(new Integer(5));
      assertTrue(LL7.get(0).equals(new Integer(5)));
}

public void testobj_D7U11() {
      LL7= new LinkedList();
      LL7.addFirst(new Integer(5));
      assertTrue(LL7.getFirst().equals(new
Integer(5)));
}

public void testobj_D7U12() {
      LL7= new LinkedList();
      LL7.addFirst(new Integer(5));
      assertTrue(LL7.getLast().equals(new Integer(5)));
}

public void testobj_D7U13() {
      LL7= new LinkedList();
      LL7.addFirst(new Integer(5));
      assertTrue(LL7.indexOf(new Integer(3))==-1);
}

public void testobj_D7U14() {
      LL7= new LinkedList();
      LL7.addFirst(new Integer(5));
      assertTrue(LL7.lastIndexOf(new Integer(5))==0);
}
```

```
public void testobj_D7U15() {
      LL7= new LinkedList();
      LL7.addFirst(new Integer(5));
      LL7.listIterator(0);
}

public void testobj_D7U16() {
      LL7= new LinkedList();
      LL7.addFirst(new Integer(5));
      LL7.remove(0);
      assertTrue(!LL7.contains(new Integer(5)));
}

public void testobj_D7U17() {
      LL7= new LinkedList();
      LL7.addFirst(new Integer(5));
      assertTrue(LL7.remove(new Integer(5))==true);
      assertTrue(LL7.indexOf(new Integer(5))==-1);
}


public void testobj_D7U18() {
      LL7= new LinkedList();
      LL7.addFirst(new Integer(5));
      LL7.removeFirst();
      assertTrue(!LL7.contains(new Integer(5)));
}

public void testobj_D7U19() {
      LL7= new LinkedList();
      LL7.addFirst(new Integer(5));
      LL7.removeLast();
      assertTrue(!LL7.contains(new Integer(5)));
}

public void testobj_D7U20() {
      LL7= new LinkedList();
      LL7.addFirst(new Integer(5));
      LL7.set(0, new Integer(2));
      assertTrue(LL7.contains(new Integer(2)));
      assertTrue(!LL7.contains(new Integer(5)));
}

public void testobj_D7U21() {
      LL7= new LinkedList();
      LL7.addFirst(new Integer(5));
      assertTrue(LL7.size()==1);
}

public void testobj_D7U22() {
      LL7= new LinkedList();
      LL7.addFirst(new Integer(5));
```

```
      LinkedList arr[]=new LinkedList[10];
      LL7.toArray();
}

public void testobj_D7U23() {
      int jj=0;
      Integer a[]=new Integer[10];
      LinkedList arr[]=new LinkedList[10];
      LL7= new LinkedList();
      LL7.addFirst(new Integer(5));
      LL7.toArray(a);
}


//##############8TH OBJ DEF USES START#########

public void testobj_D8U1() {
      LL8= new LinkedList();
      LL8.addLast(new String("abc"));
      LL8.add(1, new Integer(1));
      assertTrue(LL8.indexOf(new Integer(1))==1);
}

public void testobj_D8U2() {
      LL8= new LinkedList();
      LL8.addLast(new String("abc"));
      LL8.add(new String("abc"));
      assertTrue(LL8.lastIndexOf(new
String("abc"))==1);
}

public void testobj_D8U3() {
      LL8= new LinkedList();
      LL8.addLast(new String("abc"));
      LL8.addAll(collecshan);
      assertTrue(LL8.lastIndexOf(new Integer(3))==3);
}

public void testobj_D8U4() {
      LL8= new LinkedList();
      LL8.addLast(new String("abc"));
      LL8.addAll(1, collecshan);
      assertTrue(LL8.indexOf(new Integer(3))==3);
}

public void testobj_D8U5() {
      LL8= new LinkedList();
      LL8.addLast(new String("abc"));
      LL8.addFirst(new Integer(4));
      assertTrue(LL8.indexOf(new Integer(4))==0);
      assertTrue(LL8.indexOf(new String("abc"))==1);
}
```

```
public void testobj_D8U6() {
      LL8= new LinkedList();
      LL8.addLast(new String("abc"));
      LL8.addLast(new Integer(4));
      assertTrue(LL8.indexOf(new Integer(4))==1);
}

public void testobj_D8U7() {
      LL8= new LinkedList();
      LL8.addLast(new String("abc"));
      LL8.clear();
      assertTrue(!LL8.contains(new String("abc")));
}

public void testobj_D8U8() {
      LL8= new LinkedList();
      LL8.addLast(new String("abc"));
      LinkedList obj=(LinkedList)LL8.clone();
      assertTrue(obj.contains(new String("abc")));
}

public void testobj_D8U9() {
      LL8= new LinkedList();
      LL8.addLast(new String("abc"));
      assertTrue(LL8.contains(new String("abc")));
}

public void testobj_D8U10() {
      LL8= new LinkedList();
      LL8.addLast(new String("abc"));
      assertTrue(LL8.get(0).equals(new String("abc")));
}

public void testobj_D8U11() {
      LL8= new LinkedList();
      LL8.addLast(new String("abc"));
      assertTrue(LL8.getFirst().equals(new
String("abc")));
}

public void testobj_D8U12() {
      LL8= new LinkedList();
      LL8.addLast(new String("abc"));
      assertTrue(LL8.getLast().equals(new
String("abc")));
}

public void testobj_D8U13() {
      LL8= new LinkedList();
      LL8.addLast(new String("abc"));
      assertTrue(LL8.indexOf(new Integer(3))==-1);
```

```
}

public void testobj_D8U14() {
      LL8= new LinkedList();
      LL8.addLast(new String("abc"));
      assertTrue(LL8.lastIndexOf(new
String("abc"))==0);
}

public void testobj_D8U15() {
      LL8= new LinkedList();
      LL8.addLast(new String("abc"));
      LL8.listIterator(0);
}

public void testobj_D8U16() {
      LL8= new LinkedList();
      LL8.addLast(new String("abc"));
      LL8.remove(0);
      assertTrue(!LL8.contains(new String("abc")));
}

public void testobj_D8U17() {
      LL8= new LinkedList();
      LL8.addLast(new String("abc"));
      assertTrue(LL8.remove(new String("abc"))==true);
      assertTrue(LL8.indexOf(new String("abc"))==-1);
}


public void testobj_D8U18() {
      LL8= new LinkedList();
      LL8.addLast(new String("abc"));
      LL8.removeFirst();
      assertTrue(!LL8.contains(new String("abc")));
}

public void testobj_D8U19() {
      LL8= new LinkedList();
      LL8.addLast(new String("abc"));
      LL8.removeLast();
      assertTrue(!LL8.contains(new String("abc")));
}

public void testobj_D8U20() {
      LL8= new LinkedList();
      LL8.addLast(new String("abc"));
      LL8.set(0, new Integer(2));
      assertTrue(LL8.contains(new Integer(2)));
      assertTrue(!LL8.contains(new String("abc")));
}
```

```
public void testobj_D8U21() {
      LL8= new LinkedList();
      LL8.addLast(new String("abc"));
      assertTrue(LL8.size()==1);
}

public void testobj_D8U22() {
      LL8= new LinkedList();
      LL8.addLast(new String("abc"));
      LL8.toArray();
}

public void testobj_D8U23() {
      int jj=0;
      Integer a[]=new Integer[10];
      LinkedList arr[]=new LinkedList[10];
      LL8= new LinkedList();
      LL8.addLast(new String("abc"));
      try {
      LL8.toArray(a);
      } catch(ArrayStoreException ase) {
            jj=1;
      }
      if(jj==0) assertTrue(false);
}


//############9TH OBJ DEF USE#######################

public void testobj_D9U1() {
      LL9= new LinkedList(collecshan);
      LL9.clear();
      LL9.add(0, new Integer(7));
      assertTrue(LL9.contains(new Integer(7)));
}

public void testobj_D9U2() {
      LL9= new LinkedList(collecshan);
      LL9.clear();
      LL9.add(new String("abc"));
      assertTrue(LL9.contains(new String("abc")));
}

public void testobj_D9U3() {
      LL9= new LinkedList(collecshan);
      LL9.clear();
      LL9.addAll(collecshan);
      assertTrue(LL9.indexOf(new Integer(3))==2);
}

public void testobj_D9U4() {
      LL9= new LinkedList(collecshan);
```

```
        LL9.clear();
        LL9.addAll(0, collecshan);
        assertTrue(LL9.indexOf(new Integer(3))==2);
    }

    public void testobj_D9U5() {
        LL9= new LinkedList(collecshan);
        LL9.clear();
        LL9.addFirst(new Integer(4));
        assertTrue(LL9.indexOf(new Integer(4))==0);
    }

    public void testobj_D9U6() {
        LL9= new LinkedList(collecshan);
        LL9.clear();
        LL9.addLast(new Integer(4));
        assertTrue(LL9.indexOf(new Integer(4))==0);
    }

    public void testobj_D9U7() {
        LL9= new LinkedList(collecshan);
        LL9.clear();
        LL9.clear();
        assertTrue(LL9.isEmpty());
    }

    public void testobj_D9U8() {
        LL9= new LinkedList(collecshan);
        LL9.clear();
        LinkedList obj=(LinkedList)LL9.clone();
        assertTrue(obj.equals(LL9));
    }

    public void testobj_D9U9() {
        LL9= new LinkedList(collecshan);
        LL9.clear();
        assertTrue(!LL9.contains(new Integer(1)));
    }

    public void testobj_D9U10() {
        int excep=1;
        LL9= new LinkedList(collecshan);
        LL9.clear();
        try {
        LL9.get(0);
        } catch(IndexOutOfBoundsException i) {
              System.out.println("Exception");
              excep=0;
        }
        if(excep==1) assertTrue(false);
    }
```

```
public void testobj_D9U11() {
      int excep=1;
      LL9= new LinkedList(collecshan);
      LL9.clear();
      try {
      LL9.getFirst();
      } catch(NoSuchElementException i) {
            System.out.println("Exception");
            excep=0;
      }
      if(excep==1) assertTrue(false);
}

public void testobj_D9U12() {
      int excep=1;
      LL9= new LinkedList(collecshan);
      LL9.clear();
      try {
      LL9.getLast();
      } catch(NoSuchElementException i) {
            System.out.println("Exception");
            excep=0;
      }
      if(excep==1) assertTrue(false);
}

public void testobj_D9U13() {
      LL9= new LinkedList(collecshan);
      LL9.clear();
      assertTrue(LL9.indexOf(new Integer(3))==-1);
}

public void testobj_D9U14() {
      LL9= new LinkedList(collecshan);
      LL9.clear();
      assertTrue(LL9.lastIndexOf(new Integer(3))==-1);
}

public void testobj_D9U15() {
      LL9= new LinkedList(collecshan);
      LL9.clear();
      LL9.listIterator(0);
}

public void testobj_D9U16() {
      int excep=1;
      LL9= new LinkedList(collecshan);
      LL9.clear();
      try {
      LL9.remove(0);
      } catch(IndexOutOfBoundsException i) {
            System.out.println("Exception");
```

```
            excep=0;
      }
      if(excep==1) assertTrue(false);
}

public void testobj_D9U17() {
      LL9= new LinkedList(collecshan);
      LL9.clear();
      assertTrue(LL9.remove(new Integer(1))==false);
}


public void testobj_D9U18() {
      int excep=1;
      LL9= new LinkedList(collecshan);
      LL9.clear();
      try {
      LL9.removeFirst();
      } catch(NoSuchElementException i) {
            System.out.println("Exception");
            excep=0;
      }
      if(excep==1) assertTrue(false);
}

public void testobj_D9U19() {
      int excep=1;
      LL9= new LinkedList(collecshan);
      LL9.clear();
      try {
      LL9.removeLast();
      } catch(NoSuchElementException i) {
            System.out.println("Exception");
            excep=0;
      }
      if(excep==1) assertTrue(false);
}

public void testobj_D9U20() {
      int excep=1;
      LL9= new LinkedList(collecshan);
      LL9.clear();
      try {
      LL9.set(0, new Integer(2));
      } catch(IndexOutOfBoundsException i) {
            System.out.println("Exception");
            excep=0;
      }
      if(excep==1) assertTrue(false);
}

public void testobj_D9U21() {
```

```
      LL9= new LinkedList(collecshan);
      LL9.clear();
      assertTrue(LL9.size()==0);
}

public void testobj_D9U22() {
      LL9= new LinkedList(collecshan);
      LL9.clear();
      LL9.toArray();
}

public void testobj_D9U23() {
      Integer a[]=new Integer[10];
      LinkedList arr[]=new LinkedList[10];
      LL9= new LinkedList(collecshan);
      LL9.clear();
      LL9.toArray(a);
}


//##############10TH OBJ DEF USES START#########

public void testobj_D10U1() {
      LL10= new LinkedList(collecshan);
      LL10.clone();
      LL10.add(3, new Integer(1));
      assertTrue(LL10.indexOf(new Integer(1))==3);
}

public void testobj_D10U2() {
      LL10= new LinkedList(collecshan);
      LL10.clone();
      LL10.add(new String("abc"));
      assertTrue(LL10.indexOf(new String("abc"))==3);
}

public void testobj_D10U3() {
      LL10= new LinkedList(collecshan);
      LL10.clone();
      LL10.addAll(collecshan);
      assertTrue(LL10.lastIndexOf(new Integer(3))==5);
}

public void testobj_D10U4() {
      LL10= new LinkedList(collecshan);
      LL10.clone();
      LL10.addAll(3, collecshan);
      assertTrue(LL10.indexOf(new Integer(3))==2);
}

public void testobj_D10U5() {
      LL10= new LinkedList(collecshan);
```

```
        LL10.clone();
        LL10.addFirst(new Integer(4));
        assertTrue(LL10.indexOf(new Integer(4))==0);
        assertTrue(LL10.indexOf(new Integer(7))==1);
    }

    public void testobj_D10U6() {
        LL10= new LinkedList(collecshan);
        LL10.clone();
        LL10.addLast(new Integer(4));
        assertTrue(LL10.indexOf(new Integer(4))==3);
    }

    public void testobj_D10U7() {
        LL10= new LinkedList(collecshan);
        LL10.clone();
        LL10.clear();
        assertTrue(LL10.isEmpty());
    }

    public void testobj_D10U8() {
        LL10= new LinkedList(collecshan);
        LL10.clone();
        LinkedList obj=(LinkedList)LL10.clone();
        assertTrue(obj.contains(new Integer(7)));
    }

    public void testobj_D10U9() {
        LL10= new LinkedList(collecshan);
        LL10.clone();
        assertTrue(LL10.contains(new Integer(7)));
    }

    public void testobj_D10U10() {
        LL10= new LinkedList(collecshan);
        LL10.clone();
        assertTrue(LL10.get(0).equals(new Integer(7)));
    }

    public void testobj_D10U11() {
        LL10= new LinkedList(collecshan);
        LL10.clone();
        assertTrue(LL10.getFirst().equals(new
    Integer(7)));
    }

    public void testobj_D10U12() {
        LL10= new LinkedList(collecshan);
        LL10.clone();
        assertTrue(LL10.getLast().equals(new
    Integer(3)));
    }
```

```
public void testobj_D10U13() {
      LL10= new LinkedList(collecshan);
      LL10.clone();
      assertTrue(LL10.indexOf(new Integer(3))==2);
}

public void testobj_D10U14() {
      LL10= new LinkedList(collecshan);
      LL10.clone();
      assertTrue(LL10.lastIndexOf(new Integer(5))==1);
}

public void testobj_D10U15() {
      LL10= new LinkedList(collecshan);
      LL10.clone();
      LL10.listIterator(0);
}

public void testobj_D10U16() {
      LL10= new LinkedList(collecshan);
      LL10.clone();
      LL10.remove(0);
      assertTrue(!LL10.contains(new Integer(7)));
}

public void testobj_D10U17() {
      LL10= new LinkedList(collecshan);
      LL10.clone();
      assertTrue(LL10.remove(new Integer(5))==true);
      assertTrue(LL10.indexOf(new Integer(3))==1);
}


public void testobj_D10U18() {
      LL10= new LinkedList(collecshan);
      LL10.clone();
      LL10.removeFirst();
      assertTrue(LL10.indexOf(new Integer(3))==1);
}

public void testobj_D10U19() {
      LL10= new LinkedList(collecshan);
      LL10.clone();
      LL10.removeLast();
      assertTrue(!LL10.contains(new Integer(3)));
}

public void testobj_D10U20() {
      LL10= new LinkedList(collecshan);
      LL10.clone();
      LL10.set(0, new Integer(2));
```

```
      assertTrue(!LL10.contains(new Integer(7)));
}

public void testobj_D10U21() {
      LL10= new LinkedList(collecshan);
      LL10.clone();
      assertTrue(LL10.size()==3);
}

public void testobj_D10U22() {
      LL10= new LinkedList(collecshan);
      LL10.clone();
      LL10.toArray();
}

public void testobj_D10U23() {
      Integer a[]=new Integer[10];
      LinkedList arr[]=new LinkedList[10];
      LL10= new LinkedList(collecshan);
      LL10.clone();
      LL10.toArray(a);
}


//##############11NTH OBJ DEF USES START#########

public void testobj_D11U1() {
      LL11= new LinkedList(collecshan);
      LL11.contains(new Integer(3));
      LL11.add(3, new Integer(1));
      assertTrue(LL11.indexOf(new Integer(1))==3);
}

public void testobj_D11U2() {
      LL11= new LinkedList(collecshan);
      LL11.contains(new Integer(3));
      LL11.add(new String("abc"));
      assertTrue(LL11.indexOf(new String("abc"))==3);
}

public void testobj_D11U3() {
      LL11= new LinkedList(collecshan);
      LL11.contains(new Integer(3));
      LL11.addAll(collecshan);
      assertTrue(LL11.lastIndexOf(new Integer(3))==5);
}

public void testobj_D11U4() {
      LL11= new LinkedList(collecshan);
      LL11.contains(new Integer(3));
      LL11.addAll(3, collecshan);
      assertTrue(LL11.indexOf(new Integer(3))==2);
```

```
}

public void testobj_D11U5() {
      LL11= new LinkedList(collecshan);
      LL11.contains(new Integer(3));
      LL11.addFirst(new Integer(4));
      assertTrue(LL11.indexOf(new Integer(4))==0);
      assertTrue(LL11.indexOf(new Integer(7))==1);
}

public void testobj_D11U6() {
      LL11= new LinkedList(collecshan);
      LL11.contains(new Integer(3));
      LL11.addLast(new Integer(4));
      assertTrue(LL11.indexOf(new Integer(4))==3);
}

public void testobj_D11U7() {
      LL11= new LinkedList(collecshan);
      LL11.contains(new Integer(3));
      LL11.clear();
      assertTrue(LL11.isEmpty());
}

public void testobj_D11U8() {
      LL11= new LinkedList(collecshan);
      LL11.contains(new Integer(3));
      LinkedList obj=(LinkedList)LL11.clone();
      assertTrue(obj.contains(new Integer(7)));
}

public void testobj_D11U9() {
      LL11= new LinkedList(collecshan);
      LL11.contains(new Integer(3));
      assertTrue(LL11.contains(new Integer(7)));
}

public void testobj_D11U10() {
      LL11= new LinkedList(collecshan);
      LL11.contains(new Integer(3));
      assertTrue(LL11.get(0).equals(new Integer(7)));
}

public void testobj_D11U11() {
      LL11= new LinkedList(collecshan);
      LL11.contains(new Integer(3));
      assertTrue(LL11.getFirst().equals(new
Integer(7)));
}

public void testobj_D11U12() {
      LL11= new LinkedList(collecshan);
```

```
      LL11.contains(new Integer(3));
      assertTrue(LL11.getLast().equals(new
Integer(3)));
}

public void testobj_D11U13() {
      LL11= new LinkedList(collecshan);
      LL11.contains(new Integer(3));
      assertTrue(LL11.indexOf(new Integer(3))==2);
}

public void testobj_D11U14() {
      LL11= new LinkedList(collecshan);
      LL11.contains(new Integer(3));
      assertTrue(LL11.lastIndexOf(new Integer(5))==1);
}

public void testobj_D11U15() {
      LL11= new LinkedList(collecshan);
      LL11.contains(new Integer(3));
      LL11.listIterator(0);
}

public void testobj_D11U16() {
      LL11= new LinkedList(collecshan);
      LL11.contains(new Integer(3));
      LL11.remove(0);
      assertTrue(!LL11.contains(new Integer(7)));
}

public void testobj_D11U17() {
      LL11= new LinkedList(collecshan);
      LL11.contains(new Integer(3));
      assertTrue(LL11.remove(new Integer(5))==true);
      assertTrue(LL11.indexOf(new Integer(3))==1);
}


public void testobj_D11U18() {
      LL11= new LinkedList(collecshan);
      LL11.contains(new Integer(3));
      LL11.removeFirst();
      assertTrue(LL11.indexOf(new Integer(3))==1);
}

public void testobj_D11U19() {
      LL11= new LinkedList(collecshan);
      LL11.contains(new Integer(3));
      LL11.removeLast();
      assertTrue(!LL11.contains(new Integer(3)));
}
```

```
public void testobj_D11U20() {
      LL11= new LinkedList(collecshan);
      LL11.contains(new Integer(3));
      LL11.set(0, new Integer(2));
      assertTrue(!LL11.contains(new Integer(7)));
}

public void testobj_D11U21() {
      LL11= new LinkedList(collecshan);
      LL11.contains(new Integer(3));
      assertTrue(LL11.size()==3);
}

public void testobj_D11U22() {
      LL11= new LinkedList(collecshan);
      LL11.contains(new Integer(3));
      LL11.toArray();
}

public void testobj_D11U23() {
      Integer a[]=new Integer[10];
      LinkedList arr[]=new LinkedList[10];
      LL11= new LinkedList(collecshan);
      LL11.contains(new Integer(3));
      LL11.toArray(a);
}


//##############12TH OBJ DEF USES START#########

public void testobj_D12U1() {
      LL12= new LinkedList(collecshan);
      LL12.get(0);
      LL12.add(3, new Integer(1));
      assertTrue(LL12.indexOf(new Integer(1))==3);
}

public void testobj_D12U2() {
      LL12= new LinkedList(collecshan);
      LL12.get(0);
      LL12.add(new String("abc"));
      assertTrue(LL12.indexOf(new String("abc"))==3);
}

public void testobj_D12U3() {
      LL12= new LinkedList(collecshan);
      LL12.get(0);
      LL12.addAll(collecshan);
      assertTrue(LL12.lastIndexOf(new Integer(3))==5);
}

public void testobj_D12U4() {
```

```
        LL12= new LinkedList(collecshan);
        LL12.get(0);
        LL12.addAll(3, collecshan);
        assertTrue(LL12.indexOf(new Integer(3))==2);
    }

    public void testobj_D12U5() {
        LL12= new LinkedList(collecshan);
        LL12.get(0);
        LL12.addFirst(new Integer(4));
        assertTrue(LL12.indexOf(new Integer(4))==0);
        assertTrue(LL12.indexOf(new Integer(7))==1);
    }

    public void testobj_D12U6() {
        LL12= new LinkedList(collecshan);
        LL12.get(0);
        LL12.addLast(new Integer(4));
        assertTrue(LL12.indexOf(new Integer(4))==3);
    }

    public void testobj_D12U7() {
        LL12= new LinkedList(collecshan);
        LL12.get(0);
        LL12.clear();
        assertTrue(LL12.isEmpty());
    }

    public void testobj_D12U8() {
        LL12= new LinkedList(collecshan);
        LL12.get(0);
        LinkedList obj=(LinkedList)LL12.clone();
        assertTrue(obj.contains(new Integer(7)));
    }

    public void testobj_D12U9() {
        LL12= new LinkedList(collecshan);
        LL12.get(0);
        assertTrue(LL12.contains(new Integer(7)));
    }

    public void testobj_D12U10() {
        LL12= new LinkedList(collecshan);
        LL12.get(0);
        assertTrue(LL12.get(0).equals(new Integer(7)));
    }

    public void testobj_D12U11() {
        LL12= new LinkedList(collecshan);
        LL12.get(0);
        assertTrue(LL12.getFirst().equals(new
    Integer(7)));
```

```
}

public void testobj_D12U12() {
      LL12= new LinkedList(collecshan);
      LL12.get(0);
      assertTrue(LL12.getLast().equals(new
Integer(3)));
}

public void testobj_D12U13() {
      LL12= new LinkedList(collecshan);
      LL12.get(0);
      assertTrue(LL12.indexOf(new Integer(3))==2);
}

public void testobj_D12U14() {
      LL12= new LinkedList(collecshan);
      LL12.get(0);
      assertTrue(LL12.lastIndexOf(new Integer(5))==1);
}

public void testobj_D12U15() {
      LL12= new LinkedList(collecshan);
      LL12.get(0);
      LL12.listIterator(0);
}

public void testobj_D12U16() {
      LL12= new LinkedList(collecshan);
      LL12.get(0);
      LL12.remove(0);
      assertTrue(!LL12.contains(new Integer(7)));
}

public void testobj_D12U17() {
      LL12= new LinkedList(collecshan);
      LL12.get(0);
      assertTrue(LL12.remove(new Integer(5))==true);
      assertTrue(LL12.indexOf(new Integer(3))==1);
}


public void testobj_D12U18() {
      LL12= new LinkedList(collecshan);
      LL12.get(0);
      LL12.removeFirst();
      assertTrue(LL12.indexOf(new Integer(3))==1);
}

public void testobj_D12U19() {
      LL12= new LinkedList(collecshan);
      LL12.get(0);
```

```
      LL12.removeLast();
      assertTrue(!LL12.contains(new Integer(3)));
}

public void testobj_D12U20() {
      LL12= new LinkedList(collecshan);
      LL12.get(0);
      LL12.set(0, new Integer(2));
      assertTrue(!LL12.contains(new Integer(7)));
}

public void testobj_D12U21() {
      LL12= new LinkedList(collecshan);
      LL12.get(0);
      assertTrue(LL12.size()==3);
}

public void testobj_D12U22() {
      LL12= new LinkedList(collecshan);
      LL12.get(0);
      LL12.toArray();
}

public void testobj_D12U23() {
      LL12= new LinkedList(collecshan);
      LL12.get(0);
      Integer a[]=new Integer[10];
      LinkedList arr[]=new LinkedList[10];
      LL12.toArray(a);
}


//##############12TH OBJ DEF USES START#########

public void testobj_D13U1() {
      LL13= new LinkedList(collecshan);
      LL13.getFirst();
      LL13.add(3, new Integer(1));
      assertTrue(LL13.indexOf(new Integer(1))==3);
}

public void testobj_D13U2() {
      LL13= new LinkedList(collecshan);
      LL13.getFirst();
      LL13.add(new String("abc"));
      assertTrue(LL13.indexOf(new String("abc"))==3);
}

public void testobj_D13U3() {
      LL13= new LinkedList(collecshan);
      LL13.getFirst();
      LL13.addAll(collecshan);
```

```
        assertTrue(LL13.lastIndexOf(new Integer(3))==5);
    }

    public void testobj_D13U4() {
        LL13= new LinkedList(collecshan);
        LL13.getFirst();
        LL13.addAll(3, collecshan);
        assertTrue(LL13.indexOf(new Integer(3))==2);
    }

    public void testobj_D13U5() {
        LL13= new LinkedList(collecshan);
        LL13.getFirst();
        LL13.addFirst(new Integer(4));
        assertTrue(LL13.indexOf(new Integer(4))==0);
        assertTrue(LL13.indexOf(new Integer(7))==1);
    }

    public void testobj_D13U6() {
        LL13= new LinkedList(collecshan);
        LL13.getFirst();
        LL13.addLast(new Integer(4));
        assertTrue(LL13.indexOf(new Integer(4))==3);
    }

    public void testobj_D13U7() {
        LL13= new LinkedList(collecshan);
        LL13.getFirst();
        LL13.clear();
        assertTrue(LL13.isEmpty());
    }

    public void testobj_D13U8() {
        LL13= new LinkedList(collecshan);
        LL13.getFirst();
        LinkedList obj=(LinkedList)LL13.clone();
        assertTrue(obj.contains(new Integer(7)));
    }

    public void testobj_D13U9() {
        LL13= new LinkedList(collecshan);
        LL13.getFirst();
        assertTrue(LL13.contains(new Integer(7)));
    }

    public void testobj_D13U10() {
        LL13= new LinkedList(collecshan);
        LL13.getFirst();
        assertTrue(LL13.get(0).equals(new Integer(7)));
    }

    public void testobj_D13U11() {
```

```
      LL13= new LinkedList(collecshan);
      LL13.getFirst();
      assertTrue(LL13.getFirst().equals(new
Integer(7)));
}

public void testobj_D13U12() {
      LL13= new LinkedList(collecshan);
      LL13.getFirst();
      assertTrue(LL13.getLast().equals(new
Integer(3)));
}

public void testobj_D13U13() {
      LL13= new LinkedList(collecshan);
      LL13.getFirst();
      assertTrue(LL13.indexOf(new Integer(3))==2);
}

public void testobj_D13U14() {
      LL13= new LinkedList(collecshan);
      LL13.getFirst();
      assertTrue(LL13.lastIndexOf(new Integer(5))==1);
}

public void testobj_D13U15() {
      LL13= new LinkedList(collecshan);
      LL13.getFirst();
      LL13.listIterator(0);
}

public void testobj_D13U16() {
      LL13= new LinkedList(collecshan);
      LL13.getFirst();
      LL13.remove(0);
      assertTrue(!LL13.contains(new Integer(7)));
}

public void testobj_D13U17() {
      LL13= new LinkedList(collecshan);
      LL13.getFirst();
      assertTrue(LL13.remove(new Integer(5))==true);
      assertTrue(LL13.indexOf(new Integer(3))==1);
}


public void testobj_D13U18() {
      LL13= new LinkedList(collecshan);
      LL13.getFirst();
      LL13.removeFirst();
      assertTrue(LL13.indexOf(new Integer(3))==1);
}
```

```
public void testobj_D13U19() {
      LL13= new LinkedList(collecshan);
      LL13.getFirst();
      LL13.removeLast();
      assertTrue(!LL13.contains(new Integer(3)));
}

public void testobj_D13U20() {
      LL13= new LinkedList(collecshan);
      LL13.getFirst();
      LL13.set(0, new Integer(2));
      assertTrue(!LL13.contains(new Integer(7)));
}

public void testobj_D13U21() {
      LL13= new LinkedList(collecshan);
      LL13.getFirst();
      assertTrue(LL13.size()==3);
}

public void testobj_D13U22() {
      LL13= new LinkedList(collecshan);
      LL13.getFirst();
      LL13.toArray();
}

public void testobj_D13U23() {
      Integer a[]=new Integer[10];
      LinkedList arr[]=new LinkedList[10];
      LL13= new LinkedList(collecshan);
      LL13.getFirst();
      LL13.toArray(a);
}



//##############14NTH OBJ DEF USES START#########

public void testobj_D14U1() {
      LL14= new LinkedList(collecshan);
      LL14.getLast();
      LL14.add(3, new Integer(1));
      assertTrue(LL14.indexOf(new Integer(1))==3);
}

public void testobj_D14U2() {
      LL14= new LinkedList(collecshan);
      LL14.getLast();
      LL14.add(new String("abc"));
      assertTrue(LL14.indexOf(new String("abc"))==3);
}
```

```
public void testobj_D14U3() {
      LL14= new LinkedList(collecshan);
      LL14.getLast();
      LL14.addAll(collecshan);
      assertTrue(LL14.lastIndexOf(new Integer(3))==5);
}

public void testobj_D14U4() {
      LL14= new LinkedList(collecshan);
      LL14.getLast();
      LL14.addAll(3, collecshan);
      assertTrue(LL14.indexOf(new Integer(3))==2);
}

public void testobj_D14U5() {
      LL14= new LinkedList(collecshan);
      LL14.getLast();
      LL14.addFirst(new Integer(4));
      assertTrue(LL14.indexOf(new Integer(4))==0);
      assertTrue(LL14.indexOf(new Integer(7))==1);
}

public void testobj_D14U6() {
      LL14= new LinkedList(collecshan);
      LL14.getLast();
      LL14.addLast(new Integer(4));
      assertTrue(LL14.indexOf(new Integer(4))==3);
}

public void testobj_D14U7() {
      LL14= new LinkedList(collecshan);
      LL14.getLast();
      LL14.clear();
      assertTrue(LL14.isEmpty());
}

public void testobj_D14U8() {
      LL14= new LinkedList(collecshan);
      LL14.getLast();
      LinkedList obj=(LinkedList)LL14.clone();
      assertTrue(obj.contains(new Integer(7)));
}

public void testobj_D14U9() {
      LL14= new LinkedList(collecshan);
      LL14.getLast();
      assertTrue(LL14.contains(new Integer(7)));
}

public void testobj_D14U10() {
      LL14= new LinkedList(collecshan);
```

```
      LL14.getLast();
      assertTrue(LL14.get(0).equals(new Integer(7)));
}

public void testobj_D14U11() {
      LL14= new LinkedList(collecshan);
      LL14.getLast();
      assertTrue(LL14.getFirst().equals(new
Integer(7)));
}

public void testobj_D14U12() {
      LL14= new LinkedList(collecshan);
      LL14.getLast();
      assertTrue(LL14.getLast().equals(new
Integer(3)));
}

public void testobj_D14U13() {
      LL14= new LinkedList(collecshan);
      LL14.getLast();
      assertTrue(LL14.indexOf(new Integer(3))==2);
}

public void testobj_D14U14() {
      LL14= new LinkedList(collecshan);
      LL14.getLast();
      assertTrue(LL14.lastIndexOf(new Integer(5))==1);
}

public void testobj_D14U15() {
      LL14= new LinkedList(collecshan);
      LL14.getLast();
      LL14.listIterator(0);
}

public void testobj_D14U16() {
      LL14= new LinkedList(collecshan);
      LL14.getLast();
      LL14.remove(0);
      assertTrue(!LL14.contains(new Integer(7)));
}

public void testobj_D14U17() {
      LL14= new LinkedList(collecshan);
      LL14.getLast();
      assertTrue(LL14.remove(new Integer(5))==true);
      assertTrue(LL14.indexOf(new Integer(3))==1);
}


public void testobj_D14U18() {
```

```
      LL14= new LinkedList(collecshan);
      LL14.getLast();
      LL14.removeFirst();
      assertTrue(LL14.indexOf(new Integer(3))==1);
}

public void testobj_D14U19() {
      LL14= new LinkedList(collecshan);
      LL14.getLast();
      LL14.removeLast();
      assertTrue(!LL14.contains(new Integer(3)));
}

public void testobj_D14U20() {
      LL14= new LinkedList(collecshan);
      LL14.getLast();
      LL14.set(0, new Integer(2));
      assertTrue(!LL14.contains(new Integer(7)));
}

public void testobj_D14U21() {
      LL14= new LinkedList(collecshan);
      LL14.getLast();
      assertTrue(LL14.size()==3);
}

public void testobj_D14U22() {
      LL14= new LinkedList(collecshan);
      LL14.getLast();
      LL14.toArray();
}

public void testobj_D14U23() {
      Integer a[]=new Integer[10];
      LinkedList arr[]=new LinkedList[10];
      LL14= new LinkedList(collecshan);
      LL14.getLast();
      LL14.toArray(a);
}


//##############15NTH OBJ DEF USES START#########

public void testobj_D15U1() {
      LL15= new LinkedList(collecshan);
      LL15.indexOf(new Integer(3));
      LL15.add(3, new Integer(1));
      assertTrue(LL15.indexOf(new Integer(1))==3);
}

public void testobj_D15U2() {
      LL15= new LinkedList(collecshan);
```

```
        LL15.indexOf(new Integer(3));
        LL15.add(new String("abc"));
        assertTrue(LL15.indexOf(new String("abc"))==3);
    }

    public void testobj_D15U3() {
        LL15= new LinkedList(collecshan);
        LL15.indexOf(new Integer(3));
        LL15.addAll(collecshan);
        assertTrue(LL15.lastIndexOf(new Integer(3))==5);
    }

    public void testobj_D15U4() {
        LL15= new LinkedList(collecshan);
        LL15.indexOf(new Integer(3));
        LL15.addAll(3, collecshan);
        assertTrue(LL15.indexOf(new Integer(3))==2);
    }

    public void testobj_D15U5() {
        LL15= new LinkedList(collecshan);
        LL15.indexOf(new Integer(3));
        LL15.addFirst(new Integer(4));
        assertTrue(LL15.indexOf(new Integer(4))==0);
        assertTrue(LL15.indexOf(new Integer(7))==1);
    }

    public void testobj_D15U6() {
        LL15= new LinkedList(collecshan);
        LL15.indexOf(new Integer(3));
        LL15.addLast(new Integer(4));
        assertTrue(LL15.indexOf(new Integer(4))==3);
    }

    public void testobj_D15U7() {
        LL15= new LinkedList(collecshan);
        LL15.indexOf(new Integer(3));
        LL15.clear();
        assertTrue(LL15.isEmpty());
    }

    public void testobj_D15U8() {
        LL15= new LinkedList(collecshan);
        LL15.indexOf(new Integer(3));
        LinkedList obj=(LinkedList)LL15.clone();
        assertTrue(obj.contains(new Integer(7)));
    }

    public void testobj_D15U9() {
        LL15= new LinkedList(collecshan);
        LL15.indexOf(new Integer(3));
        assertTrue(LL15.contains(new Integer(7)));
```

```
}

public void testobj_D15U10() {
      LL15= new LinkedList(collecshan);
      LL15.indexOf(new Integer(3));
      assertTrue(LL15.get(0).equals(new Integer(7)));
}

public void testobj_D15U11() {
      LL15= new LinkedList(collecshan);
      LL15.indexOf(new Integer(3));
      assertTrue(LL15.getFirst().equals(new
Integer(7)));
}

public void testobj_D15U12() {
      LL15= new LinkedList(collecshan);
      LL15.indexOf(new Integer(3));
      assertTrue(LL15.getLast().equals(new
Integer(3)));
}

public void testobj_D15U13() {
      LL15= new LinkedList(collecshan);
      LL15.indexOf(new Integer(3));
      assertTrue(LL15.indexOf(new Integer(3))==2);
}

public void testobj_D15U14() {
      LL15= new LinkedList(collecshan);
      LL15.indexOf(new Integer(3));
      assertTrue(LL15.lastIndexOf(new Integer(5))==1);
}

public void testobj_D15U15() {
      LL15= new LinkedList(collecshan);
      LL15.indexOf(new Integer(3));
      LL15.listIterator(0);
}

public void testobj_D15U16() {
      LL15= new LinkedList(collecshan);
      LL15.indexOf(new Integer(3));
      LL15.remove(0);
      assertTrue(!LL15.contains(new Integer(7)));
}

public void testobj_D15U17() {
      LL15= new LinkedList(collecshan);
      LL15.indexOf(new Integer(3));
      assertTrue(LL15.remove(new Integer(5))==true);
      assertTrue(LL15.indexOf(new Integer(3))==1);
```

```
}


public void testobj_D15U18() {
      LL15= new LinkedList(collecshan);
      LL15.indexOf(new Integer(3));
      LL15.removeFirst();
      assertTrue(LL15.indexOf(new Integer(3))==1);
}

public void testobj_D15U19() {
      LL15= new LinkedList(collecshan);
      LL15.indexOf(new Integer(3));
      LL15.removeLast();
      assertTrue(!LL15.contains(new Integer(3)));
}

public void testobj_D15U20() {
      LL15= new LinkedList(collecshan);
      LL15.indexOf(new Integer(3));
      LL15.set(0, new Integer(2));
      assertTrue(!LL15.contains(new Integer(7)));
}

public void testobj_D15U21() {
      LL15= new LinkedList(collecshan);
      LL15.indexOf(new Integer(3));
      assertTrue(LL15.size()==3);
}

public void testobj_D15U22() {
      LL15= new LinkedList(collecshan);
      LL15.indexOf(new Integer(3));
      LL15.toArray();
}

public void testobj_D15U23() {
      Integer a[]=new Integer[10];
      LinkedList arr[]=new LinkedList[10];
      LL15= new LinkedList(collecshan);
      LL15.indexOf(new Integer(3));
      LL15.toArray(a);
}


//##############16NTH OBJ DEF USES START#########

public void testobj_D16U1() {
      LL16= new LinkedList(collecshan);
      LL16.lastIndexOf(new Integer(5));
      LL16.add(3, new Integer(1));
      assertTrue(LL16.indexOf(new Integer(1))==3);
```

```
}

public void testobj_D16U2() {
      LL16= new LinkedList(collecshan);
      LL16.lastIndexOf(new Integer(5));
      LL16.add(new String("abc"));
      assertTrue(LL16.indexOf(new String("abc"))==3);
}

public void testobj_D16U3() {
      LL16= new LinkedList(collecshan);
      LL16.lastIndexOf(new Integer(5));
      LL16.addAll(collecshan);
      assertTrue(LL16.lastIndexOf(new Integer(3))==5);
}

public void testobj_D16U4() {
      LL16= new LinkedList(collecshan);
      LL16.lastIndexOf(new Integer(5));
      LL16.addAll(3, collecshan);
      assertTrue(LL16.indexOf(new Integer(3))==2);
}

public void testobj_D16U5() {
      LL16= new LinkedList(collecshan);
      LL16.lastIndexOf(new Integer(5));
      LL16.addFirst(new Integer(4));
      assertTrue(LL16.indexOf(new Integer(4))==0);
      assertTrue(LL16.indexOf(new Integer(7))==1);
}

public void testobj_D16U6() {
      LL16= new LinkedList(collecshan);
      LL16.lastIndexOf(new Integer(5));
      LL16.addLast(new Integer(4));
      assertTrue(LL16.indexOf(new Integer(4))==3);
}

public void testobj_D16U7() {
      LL16= new LinkedList(collecshan);
      LL16.lastIndexOf(new Integer(5));
      LL16.clear();
      assertTrue(LL16.isEmpty());
}

public void testobj_D16U8() {
      LL16= new LinkedList(collecshan);
      LL16.lastIndexOf(new Integer(5));
      LinkedList obj=(LinkedList)LL16.clone();
      assertTrue(obj.contains(new Integer(7)));
}
```

```
public void testobj_D16U9() {
      LL16= new LinkedList(collecshan);
      LL16.lastIndexOf(new Integer(5));
      assertTrue(LL16.contains(new Integer(7)));
}

public void testobj_D16U10() {
      LL16= new LinkedList(collecshan);
      LL16.lastIndexOf(new Integer(5));
      assertTrue(LL16.get(0).equals(new Integer(7)));
}

public void testobj_D16U11() {
      LL16= new LinkedList(collecshan);
      LL16.lastIndexOf(new Integer(5));
      assertTrue(LL16.getFirst().equals(new
Integer(7)));
}

public void testobj_D16U12() {
      LL16= new LinkedList(collecshan);
      LL16.lastIndexOf(new Integer(5));
      assertTrue(LL16.getLast().equals(new
Integer(3)));
}

public void testobj_D16U13() {
      LL16= new LinkedList(collecshan);
      LL16.lastIndexOf(new Integer(5));
      assertTrue(LL16.indexOf(new Integer(3))==2);
}

public void testobj_D16U14() {
      LL16= new LinkedList(collecshan);
      LL16.lastIndexOf(new Integer(5));
      assertTrue(LL16.lastIndexOf(new Integer(5))==1);
}

public void testobj_D16U15() {
      LL16= new LinkedList(collecshan);
      LL16.lastIndexOf(new Integer(5));
      LL16.listIterator(0);
}

public void testobj_D16U16() {
      LL16= new LinkedList(collecshan);
      LL16.lastIndexOf(new Integer(5));
      LL16.remove(0);
      assertTrue(!LL16.contains(new Integer(7)));
}

public void testobj_D16U17() {
```

```
        LL16= new LinkedList(collecshan);
        LL16.lastIndexOf(new Integer(5));
        assertTrue(LL16.remove(new Integer(5))==true);
        assertTrue(LL16.indexOf(new Integer(3))==1);
    }


    public void testobj_D16U18() {
        LL16= new LinkedList(collecshan);
        LL16.lastIndexOf(new Integer(5));
        LL16.removeFirst();
        assertTrue(LL16.indexOf(new Integer(3))==1);
    }

    public void testobj_D16U19() {
        LL16= new LinkedList(collecshan);
        LL16.lastIndexOf(new Integer(5));
        LL16.removeLast();
        assertTrue(!LL16.contains(new Integer(3)));
    }

    public void testobj_D16U20() {
        LL16= new LinkedList(collecshan);
        LL16.lastIndexOf(new Integer(5));
        LL16.set(0, new Integer(2));
        assertTrue(!LL16.contains(new Integer(7)));
    }

    public void testobj_D16U21() {
        LL16= new LinkedList(collecshan);
        LL16.lastIndexOf(new Integer(5));
        assertTrue(LL16.size()==3);
    }

    public void testobj_D16U22() {
        LL16= new LinkedList(collecshan);
        LL16.lastIndexOf(new Integer(5));
        LL16.toArray();
    }

    public void testobj_D16U23() {
        Integer a[]=new Integer[10];
        LinkedList arr[]=new LinkedList[10];
        LL16= new LinkedList(collecshan);
        LL16.lastIndexOf(new Integer(5));
        LL16.toArray(a);
    }


    //##############17NTH OBJ DEF USES START#########

    public void testobj_D17U1() {
```

```
        LL17= new LinkedList(collecshan);
        LL17.listIterator(1);
        LL17.add(3, new Integer(1));
        assertTrue(LL17.indexOf(new Integer(1))==3);
    }

    public void testobj_D17U2() {
        LL17= new LinkedList(collecshan);
        LL17.listIterator(1);
        LL17.add(new String("abc"));
        assertTrue(LL17.indexOf(new String("abc"))==3);
    }

    public void testobj_D17U3() {
        LL17= new LinkedList(collecshan);
        LL17.listIterator(1);
        LL17.addAll(collecshan);
        assertTrue(LL17.lastIndexOf(new Integer(3))==5);
    }

    public void testobj_D17U4() {
        LL17= new LinkedList(collecshan);
        LL17.listIterator(1);
        LL17.addAll(3, collecshan);
        assertTrue(LL17.indexOf(new Integer(3))==2);
    }

    public void testobj_D17U5() {
        LL17= new LinkedList(collecshan);
        LL17.listIterator(1);
        LL17.addFirst(new Integer(4));
        assertTrue(LL17.indexOf(new Integer(4))==0);
        assertTrue(LL17.indexOf(new Integer(7))==1);
    }

    public void testobj_D17U6() {
        LL17= new LinkedList(collecshan);
        LL17.listIterator(1);
        LL17.addLast(new Integer(4));
        assertTrue(LL17.indexOf(new Integer(4))==3);
    }

    public void testobj_D17U7() {
        LL17= new LinkedList(collecshan);
        LL17.listIterator(1);
        LL17.clear();
        assertTrue(LL17.isEmpty());
    }

    public void testobj_D17U8() {
        LL17= new LinkedList(collecshan);
        LL17.listIterator(1);
```

```
        LinkedList obj=(LinkedList)LL17.clone();
        assertTrue(obj.contains(new Integer(7)));
    }

    public void testobj_D17U9() {
        LL17= new LinkedList(collecshan);
        LL17.listIterator(1);
        assertTrue(LL17.contains(new Integer(7)));
    }

    public void testobj_D17U10() {
        LL17= new LinkedList(collecshan);
        LL17.listIterator(1);
        assertTrue(LL17.get(0).equals(new Integer(7)));
    }

    public void testobj_D17U11() {
        LL17= new LinkedList(collecshan);
        LL17.listIterator(1);
        assertTrue(LL17.getFirst().equals(new
    Integer(7)));
    }

    public void testobj_D17U12() {
        LL17= new LinkedList(collecshan);
        LL17.listIterator(1);
        assertTrue(LL17.getLast().equals(new
    Integer(3)));
    }

    public void testobj_D17U13() {
        LL17= new LinkedList(collecshan);
        LL17.listIterator(1);
        assertTrue(LL17.indexOf(new Integer(3))==2);
    }

    public void testobj_D17U14() {
        LL17= new LinkedList(collecshan);
        LL17.listIterator(1);
        assertTrue(LL17.lastIndexOf(new Integer(5))==1);
    }

    public void testobj_D17U15() {
        LL17= new LinkedList(collecshan);
        LL17.listIterator(1);
        LL17.listIterator(0);
    }

    public void testobj_D17U16() {
        LL17= new LinkedList(collecshan);
        LL17.listIterator(1);
        LL17.remove(0);
```

```
        assertTrue(!LL17.contains(new Integer(7)));
    }

    public void testobj_D17U17() {
        LL17= new LinkedList(collecshan);
        LL17.listIterator(1);
        assertTrue(LL17.remove(new Integer(5))==true);
        assertTrue(LL17.indexOf(new Integer(3))==1);
    }


    public void testobj_D17U18() {
        LL17= new LinkedList(collecshan);
        LL17.listIterator(1);
        LL17.removeFirst();
        assertTrue(LL17.indexOf(new Integer(3))==1);
    }

    public void testobj_D17U19() {
        LL17= new LinkedList(collecshan);
        LL17.listIterator(1);
        LL17.removeLast();
        assertTrue(!LL17.contains(new Integer(3)));
    }

    public void testobj_D17U20() {
        LL17= new LinkedList(collecshan);
        LL17.listIterator(1);
        LL17.set(0, new Integer(2));
        assertTrue(!LL17.contains(new Integer(7)));
    }

    public void testobj_D17U21() {
        LL17= new LinkedList(collecshan);
        LL17.listIterator(1);
        assertTrue(LL17.size()==3);
    }

    public void testobj_D17U22() {
        LL17= new LinkedList(collecshan);
        LL17.listIterator(1);
        LL17.toArray();
    }

    public void testobj_D17U23() {
        Integer a[]=new Integer[10];
        LinkedList arr[]=new LinkedList[10];
        LL17= new LinkedList(collecshan);
        LL17.listIterator(1);
        LL17.toArray(a);
    }
```

```
//##############18NTH OBJ DEF USES START#########

public void testobj_D18U1() {
      LL18= new LinkedList(collecshan);
      LL18.remove(1);
      LL18.add(2, new Integer(1));
      assertTrue(LL18.indexOf(new Integer(1))==2);
      assertTrue(LL18.indexOf(new Integer(3))==1);
}

public void testobj_D18U2() {
      LL18= new LinkedList(collecshan);
      LL18.remove(1);
      LL18.add(new String("abc"));
      assertTrue(LL18.indexOf(new String("abc"))==2);
}

public void testobj_D18U3() {
      LL18= new LinkedList(collecshan);
      LL18.remove(1);
      LL18.addAll(collecshan);
      assertTrue(LL18.lastIndexOf(new Integer(3))==4);
      assertTrue(LL18.indexOf(new Integer(3))==1);
}

public void testobj_D18U4() {
      LL18= new LinkedList(collecshan);
      LL18.remove(1);
      LL18.addAll(2, collecshan);
      assertTrue(LL18.indexOf(new Integer(5))==3);
}

public void testobj_D18U5() {
      LL18= new LinkedList(collecshan);
      LL18.remove(1);
      LL18.addFirst(new Integer(4));
      assertTrue(LL18.indexOf(new Integer(4))==0);
      assertTrue(LL18.indexOf(new Integer(3))==2);
}

public void testobj_D18U6() {
      LL18= new LinkedList(collecshan);
      LL18.remove(1);
      LL18.addLast(new Integer(4));
      assertTrue(LL18.indexOf(new Integer(4))==2);
}

public void testobj_D18U7() {
      LL18= new LinkedList(collecshan);
      LL18.remove(1);
      LL18.clear();
```

```
      assertTrue(LL18.isEmpty());
}

public void testobj_D18U8() {
      LL18= new LinkedList(collecshan);
      LL18.remove(1);
      LinkedList obj=(LinkedList)LL18.clone();
      assertTrue(obj.contains(new Integer(3)));
}

public void testobj_D18U9() {
      LL18= new LinkedList(collecshan);
      LL18.remove(1);
      assertTrue(!LL18.contains(new Integer(5)));
}

public void testobj_D18U10() {
      LL18= new LinkedList(collecshan);
      LL18.remove(1);
      assertTrue(LL18.get(1).equals(new Integer(3)));
}

public void testobj_D18U11() {
      LL18= new LinkedList(collecshan);
      LL18.remove(1);
      assertTrue(LL18.getFirst().equals(new
Integer(7)));
}

public void testobj_D18U12() {
      LL18= new LinkedList(collecshan);
      LL18.remove(1);
      assertTrue(LL18.getLast().equals(new
Integer(3)));
}

public void testobj_D18U13() {
      LL18= new LinkedList(collecshan);
      LL18.remove(1);
      assertTrue(LL18.indexOf(new Integer(5))==-1);
}

public void testobj_D18U14() {
      LL18= new LinkedList(collecshan);
      LL18.remove(1);
      assertTrue(LL18.lastIndexOf(new Integer(5))==-1);
}

public void testobj_D18U15() {
      LL18= new LinkedList(collecshan);
      LL18.remove(1);
      LL18.listIterator(0);
```

```
}

public void testobj_D18U16() {
      LL18= new LinkedList(collecshan);
      LL18.remove(1);
      LL18.remove(0);
      assertTrue(LL18.size()==1);
}

public void testobj_D18U17() {
      LL18= new LinkedList(collecshan);
      LL18.remove(1);
      assertTrue(LL18.remove(new Integer(5))==false);
      assertTrue(LL18.indexOf(new Integer(3))==1);
}


public void testobj_D18U18() {
      LL18= new LinkedList(collecshan);
      LL18.remove(1);
      LL18.removeFirst();
      assertTrue(LL18.indexOf(new Integer(3))==0);
}

public void testobj_D18U19() {
      LL18= new LinkedList(collecshan);
      LL18.remove(1);
      LL18.removeLast();
      assertTrue(!LL18.contains(new Integer(3)));
}

public void testobj_D18U20() {
      LL18= new LinkedList(collecshan);
      LL18.remove(1);
      LL18.set(0, new Integer(5));
      assertTrue(LL18.contains(new Integer(5)));
}

public void testobj_D18U21() {
      LL18= new LinkedList(collecshan);
      LL18.remove(1);
      assertTrue(LL18.size()==2);
}

public void testobj_D18U22() {
      LL18= new LinkedList(collecshan);
      LL18.remove(1);
      LL18.toArray();
}

public void testobj_D18U23() {
      Integer a[]=new Integer[10];
```

```
        LinkedList arr[]=new LinkedList[10];
        LL18= new LinkedList(collecshan);
        LL18.remove(1);
        LL18.toArray(a);
    }


    //##############19NTH OBJ DEF USES START#########

    public void testobj_D19U1() {
        LL19= new LinkedList(collecshan);
        LL19.remove(new Integer(7));
        LL19.add(2, new Integer(1));
        assertTrue(LL19.indexOf(new Integer(1))==2);
        assertTrue(LL19.indexOf(new Integer(3))==1);
    }

    public void testobj_D19U2() {
        LL19= new LinkedList(collecshan);
        LL19.remove(new Integer(7));
        LL19.add(new String("abc"));
        assertTrue(LL19.indexOf(new String("abc"))==2);
    }

    public void testobj_D19U3() {
        LL19= new LinkedList(collecshan);
        LL19.remove(new Integer(7));
        LL19.addAll(collecshan);
        assertTrue(LL19.lastIndexOf(new Integer(3))==4);
        assertTrue(LL19.indexOf(new Integer(7))==2);
    }

    public void testobj_D19U4() {
        LL19= new LinkedList(collecshan);
        LL19.remove(new Integer(7));
        LL19.addAll(2, collecshan);
        assertTrue(LL19.indexOf(new Integer(7))==2);
    }

    public void testobj_D19U5() {
        LL19= new LinkedList(collecshan);
        LL19.remove(new Integer(7));
        LL19.addFirst(new Integer(4));
        assertTrue(LL19.indexOf(new Integer(4))==0);
        assertTrue(LL19.indexOf(new Integer(5))==1);
    }

    public void testobj_D19U6() {
        LL19= new LinkedList(collecshan);
        LL19.remove(new Integer(7));
        LL19.addLast(new Integer(4));
        assertTrue(LL19.indexOf(new Integer(4))==2);
```

```
}

public void testobj_D19U7() {
      LL19= new LinkedList(collecshan);
      LL19.remove(new Integer(7));
      LL19.clear();
      assertTrue(LL19.isEmpty());
}

public void testobj_D19U8() {
      LL19= new LinkedList(collecshan);
      LL19.remove(new Integer(7));
      LinkedList obj=(LinkedList)LL19.clone();
      assertTrue(obj.contains(new Integer(3)));
}

public void testobj_D19U9() {
      LL19= new LinkedList(collecshan);
      LL19.remove(new Integer(7));
      assertTrue(!LL19.contains(new Integer(7)));
}

public void testobj_D19U10() {
      LL19= new LinkedList(collecshan);
      LL19.remove(new Integer(7));
      assertTrue(LL19.get(0).equals(new Integer(5)));
}

public void testobj_D19U11() {
      LL19= new LinkedList(collecshan);
      LL19.remove(new Integer(7));
      assertTrue(LL19.getFirst().equals(new
Integer(5)));
}

public void testobj_D19U12() {
      LL19= new LinkedList(collecshan);
      LL19.remove(new Integer(7));
      assertTrue(LL19.getLast().equals(new
Integer(3)));
}

public void testobj_D19U13() {
      LL19= new LinkedList(collecshan);
      LL19.remove(new Integer(7));
      assertTrue(LL19.indexOf(new Integer(7))==-1);
}

public void testobj_D19U14() {
      LL19= new LinkedList(collecshan);
      LL19.remove(new Integer(7));
      assertTrue(LL19.lastIndexOf(new Integer(7))==-1);
```

```
}

public void testobj_D19U15() {
      LL19= new LinkedList(collecshan);
      LL19.remove(new Integer(7));
      LL19.listIterator(0);
}

public void testobj_D19U16() {
      LL19= new LinkedList(collecshan);
      LL19.remove(new Integer(7));
      LL19.remove(0);
      assertTrue(LL19.size()==1);
}

public void testobj_D19U17() {
      LL19= new LinkedList(collecshan);
      LL19.remove(new Integer(7));
      assertTrue(LL19.remove(new Integer(7))==false);
      assertTrue(LL19.indexOf(new Integer(5))==0);
}


public void testobj_D19U18() {
      LL19= new LinkedList(collecshan);
      LL19.remove(new Integer(7));
      LL19.removeFirst();
      assertTrue(LL19.indexOf(new Integer(3))==0);
}

public void testobj_D19U19() {
      LL19= new LinkedList(collecshan);
      LL19.remove(new Integer(7));
      LL19.removeLast();
      assertTrue(!LL19.contains(new Integer(3)));
}

public void testobj_D19U20() {
      LL19= new LinkedList(collecshan);
      LL19.remove(new Integer(7));
      LL19.set(0, new Integer(7));
      assertTrue(LL19.contains(new Integer(7)));
}

public void testobj_D19U21() {
      LL19= new LinkedList(collecshan);
      LL19.remove(new Integer(7));
      assertTrue(LL19.size()==2);
}

public void testobj_D19U22() {
      LL19= new LinkedList(collecshan);
```

```
      LL19.remove(new Integer(7));
      LL19.toArray();
}

public void testobj_D19U23() {
      Integer a[]=new Integer[10];
      LinkedList arr[]=new LinkedList[10];
      LL19= new LinkedList(collecshan);
      LL19.remove(new Integer(7));
      LL19.toArray(a);
}


//##############20NTH OBJ DEF USES START#########

public void testobj_D20U1() {
      LL20= new LinkedList(collecshan);
      LL20.removeFirst();
      LL20.add(2, new Integer(1));
      assertTrue(LL20.indexOf(new Integer(1))==2);
      assertTrue(LL20.indexOf(new Integer(3))==1);
}

public void testobj_D20U2() {
      LL20= new LinkedList(collecshan);
      LL20.removeFirst();
      LL20.add(new String("abc"));
      assertTrue(LL20.indexOf(new String("abc"))==2);
}

public void testobj_D20U3() {
      LL20= new LinkedList(collecshan);
      LL20.removeFirst();
      LL20.addAll(collecshan);
      assertTrue(LL20.lastIndexOf(new Integer(3))==4);
      assertTrue(LL20.indexOf(new Integer(7))==2);
}

public void testobj_D20U4() {
      LL20= new LinkedList(collecshan);
      LL20.removeFirst();
      LL20.addAll(2, collecshan);
      assertTrue(LL20.indexOf(new Integer(7))==2);
}

public void testobj_D20U5() {
      LL20= new LinkedList(collecshan);
      LL20.removeFirst();
      LL20.addFirst(new Integer(4));
      assertTrue(LL20.indexOf(new Integer(4))==0);
      assertTrue(LL20.indexOf(new Integer(5))==1);
}
```

```
public void testobj_D20U6() {
      LL20= new LinkedList(collecshan);
      LL20.removeFirst();
      LL20.addLast(new Integer(4));
      assertTrue(LL20.indexOf(new Integer(4))==2);
}

public void testobj_D20U7() {
      LL20= new LinkedList(collecshan);
      LL20.removeFirst();
      LL20.clear();
      assertTrue(LL20.isEmpty());
}

public void testobj_D20U8() {
      LL20= new LinkedList(collecshan);
      LL20.removeFirst();
      LinkedList obj=(LinkedList)LL20.clone();
      assertTrue(obj.contains(new Integer(3)));
}

public void testobj_D20U9() {
      LL20= new LinkedList(collecshan);
      LL20.removeFirst();
      assertTrue(!LL20.contains(new Integer(7)));
}

public void testobj_D20U10() {
      LL20= new LinkedList(collecshan);
      LL20.removeFirst();
      assertTrue(LL20.get(0).equals(new Integer(5)));
}

public void testobj_D20U11() {
      LL20= new LinkedList(collecshan);
      LL20.removeFirst();
      assertTrue(LL20.getFirst().equals(new
Integer(5)));
}

public void testobj_D20U12() {
      LL20= new LinkedList(collecshan);
      LL20.removeFirst();
      assertTrue(LL20.getLast().equals(new
Integer(3)));
}

public void testobj_D20U13() {
      LL20= new LinkedList(collecshan);
      LL20.removeFirst();
      assertTrue(LL20.indexOf(new Integer(7))==-1);
```

```
}

public void testobj_D20U14() {
      LL20= new LinkedList(collecshan);
      LL20.removeFirst();
      assertTrue(LL20.lastIndexOf(new Integer(7))==-1);
}

public void testobj_D20U15() {
      LL20= new LinkedList(collecshan);
      LL20.removeFirst();
      LL20.listIterator(0);
}

public void testobj_D20U16() {
      LL20= new LinkedList(collecshan);
      LL20.removeFirst();
      LL20.remove(0);
      assertTrue(LL20.size()==1);
}

public void testobj_D20U17() {
      LL20= new LinkedList(collecshan);
      LL20.removeFirst();
      assertTrue(LL20.remove(new Integer(7))==false);
      assertTrue(LL20.indexOf(new Integer(5))==0);
}


public void testobj_D20U18() {
      LL20= new LinkedList(collecshan);
      LL20.removeFirst();
      LL20.removeFirst();
      assertTrue(LL20.indexOf(new Integer(3))==0);
}

public void testobj_D20U19() {
      LL20= new LinkedList(collecshan);
      LL20.removeFirst();
      LL20.removeLast();
      assertTrue(!LL20.contains(new Integer(3)));
}

public void testobj_D20U20() {
      LL20= new LinkedList(collecshan);
      LL20.removeFirst();
      LL20.set(0, new Integer(7));
      assertTrue(LL20.contains(new Integer(7)));
}

public void testobj_D20U21() {
      LL20= new LinkedList(collecshan);
```

```
      LL20.removeFirst();
      assertTrue(LL20.size()==2);
}

public void testobj_D20U22() {
      LL20= new LinkedList(collecshan);
      LL20.removeFirst();
      LL20.toArray();
}

public void testobj_D20U23() {
      Integer a[]=new Integer[10];
      LinkedList arr[]=new LinkedList[10];
      LL20= new LinkedList(collecshan);
      LL20.removeFirst();
      LL20.toArray(a);
}


//##############21st OBJ DEF USES START#########

public void testobj_D21U1() {
      LL21= new LinkedList(collecshan);
      LL21.removeLast();
      LL21.add(2, new Integer(1));
      assertTrue(LL21.indexOf(new Integer(1))==2);
      assertTrue(LL21.indexOf(new Integer(3))==-1);
}

public void testobj_D21U2() {
      LL21= new LinkedList(collecshan);
      LL21.removeLast();
      LL21.add(new String("abc"));
      assertTrue(LL21.indexOf(new String("abc"))==2);
}

public void testobj_D21U3() {
      LL21= new LinkedList(collecshan);
      LL21.removeLast();
      LL21.addAll(collecshan);
      assertTrue(LL21.indexOf(new Integer(3))==4);
      assertTrue(LL21.lastIndexOf(new Integer(7))==2);
}

public void testobj_D21U4() {
      LL21= new LinkedList(collecshan);
      LL21.removeLast();
      LL21.addAll(2, collecshan);
      assertTrue(LL21.indexOf(new Integer(3))==4);
}

public void testobj_D21U5() {
```

```
        LL21= new LinkedList(collecshan);
        LL21.removeLast();
        LL21.addFirst(new Integer(4));
        assertTrue(LL21.indexOf(new Integer(4))==0);
        assertTrue(LL21.size()==3);
    }

    public void testobj_D21U6() {
        LL21= new LinkedList(collecshan);
        LL21.removeLast();
        LL21.addLast(new Integer(4));
        assertTrue(LL21.indexOf(new Integer(4))==2);
    }

    public void testobj_D21U7() {
        LL21= new LinkedList(collecshan);
        LL21.removeLast();
        LL21.clear();
        assertTrue(LL21.isEmpty());
    }

    public void testobj_D21U8() {
        LL21= new LinkedList(collecshan);
        LL21.removeLast();
        LinkedList obj=(LinkedList)LL21.clone();
        assertTrue(obj.contains(new Integer(5)));
    }

    public void testobj_D21U9() {
        LL21= new LinkedList(collecshan);
        LL21.removeLast();
        assertTrue(!LL21.contains(new Integer(3)));
    }

    public void testobj_D21U10() {
        LL21= new LinkedList(collecshan);
        LL21.removeLast();
        assertTrue(LL21.get(0).equals(new Integer(7)));
    }

    public void testobj_D21U11() {
        LL21= new LinkedList(collecshan);
        LL21.removeLast();
        assertTrue(LL21.getFirst().equals(new
    Integer(7)));
    }

    public void testobj_D21U12() {
        LL21= new LinkedList(collecshan);
        LL21.removeLast();
        assertTrue(LL21.getLast().equals(new
    Integer(5)));
```

```
}

public void testobj_D21U13() {
      LL21= new LinkedList(collecshan);
      LL21.removeLast();
      assertTrue(LL21.indexOf(new Integer(3))==-1);
}

public void testobj_D21U14() {
      LL21= new LinkedList(collecshan);
      LL21.removeLast();
      assertTrue(LL21.lastIndexOf(new Integer(3))==-1);
}

public void testobj_D21U15() {
      LL21= new LinkedList(collecshan);
      LL21.removeLast();
      LL21.listIterator(0);
}

public void testobj_D21U16() {
      LL21= new LinkedList(collecshan);
      LL21.removeLast();
      LL21.remove(0);
      assertTrue(LL21.size()==1);
}

public void testobj_D21U17() {
      LL21= new LinkedList(collecshan);
      LL21.removeLast();
      assertTrue(LL21.remove(new Integer(3))==false);
      assertTrue(LL21.indexOf(new Integer(5))==1);
}


public void testobj_D21U18() {
      LL21= new LinkedList(collecshan);
      LL21.removeLast();
      LL21.removeFirst();
      assertTrue(LL21.indexOf(new Integer(3))==-1);
}

public void testobj_D21U19() {
      LL21= new LinkedList(collecshan);
      LL21.removeLast();
      LL21.removeLast();
      assertTrue(!LL21.contains(new Integer(5)));
}

public void testobj_D21U20() {
      LL21= new LinkedList(collecshan);
      LL21.removeLast();
```

```
        LL21.set(0, new Integer(3));
        assertTrue(LL21.contains(new Integer(3)));
    }

    public void testobj_D21U21() {
        LL21= new LinkedList(collecshan);
        LL21.removeLast();
        assertTrue(LL21.size()==2);
    }

    public void testobj_D21U22() {
        LL21= new LinkedList(collecshan);
        LL21.removeLast();
        LL21.toArray();
    }

    public void testobj_D21U23() {
        Integer a[]=new Integer[10];
        LinkedList arr[]=new LinkedList[10];
        LL21= new LinkedList(collecshan);
        LL21.removeLast();
        LL21.toArray(a);
    }


    //##############22st OBJ DEF USES START#########

    public void testobj_D22U1() {
        LL22= new LinkedList(collecshan);
        LL22.set(0, new Integer(1));
        LL22.add(3, new Integer(1));
        assertTrue(LL22.indexOf(new Integer(1))==0);
        assertTrue(LL22.lastIndexOf(new Integer(1))==3);
    }

    public void testobj_D22U2() {
        LL22= new LinkedList(collecshan);
        LL22.set(0, new Integer(1));
        LL22.add(new String("abc"));
        assertTrue(LL22.indexOf(new String("abc"))==3);
    }

    public void testobj_D22U3() {
        LL22= new LinkedList(collecshan);
        LL22.set(0, new Integer(1));
        LL22.addAll(collecshan);
        assertTrue(LL22.indexOf(new Integer(1))==0);
        assertTrue(LL22.lastIndexOf(new Integer(7))==3);
    }

    public void testobj_D22U4() {
        LL22= new LinkedList(collecshan);
```

```
        LL22.set(0, new Integer(1));
        LL22.addAll(3, collecshan);
        assertTrue(LL22.indexOf(new Integer(3))==2);
    }

    public void testobj_D22U5() {
        LL22= new LinkedList(collecshan);
        LL22.set(0, new Integer(1));
        LL22.addFirst(new Integer(4));
        assertTrue(LL22.indexOf(new Integer(1))==1);
        assertTrue(LL22.size()==4);
    }

    public void testobj_D22U6() {
        LL22= new LinkedList(collecshan);
        LL22.set(0, new Integer(1));
        LL22.addLast(new Integer(4));
        assertTrue(LL22.indexOf(new Integer(4))==3);
    }

    public void testobj_D22U7() {
        LL22= new LinkedList(collecshan);
        LL22.set(0, new Integer(1));
        LL22.clear();
        assertTrue(LL22.isEmpty());
    }

    public void testobj_D22U8() {
        LL22= new LinkedList(collecshan);
        LL22.set(0, new Integer(1));
        LinkedList obj=(LinkedList)LL22.clone();
        assertTrue(obj.contains(new Integer(1)));
    }

    public void testobj_D22U9() {
        LL22= new LinkedList(collecshan);
        LL22.set(0, new Integer(1));
        assertTrue(!LL22.contains(new Integer(7)));
    }

    public void testobj_D22U10() {
        LL22= new LinkedList(collecshan);
        LL22.set(0, new Integer(1));
        assertTrue(LL22.get(0).equals(new Integer(1)));
    }

    public void testobj_D22U11() {
        LL22= new LinkedList(collecshan);
        LL22.set(0, new Integer(1));
        assertTrue(LL22.getFirst().equals(new
    Integer(1)));
    }
```

```
public void testobj_D22U12() {
      LL22= new LinkedList(collecshan);
      LL22.set(0, new Integer(1));
      assertTrue(LL22.getLast().equals(new
Integer(3)));
}

public void testobj_D22U13() {
      LL22= new LinkedList(collecshan);
      LL22.set(0, new Integer(1));
      assertTrue(LL22.indexOf(new Integer(7))==-1);
}

public void testobj_D22U14() {
      LL22= new LinkedList(collecshan);
      LL22.set(0, new Integer(1));
      assertTrue(LL22.lastIndexOf(new Integer(7))==-1);
}

public void testobj_D22U15() {
      LL22= new LinkedList(collecshan);
      LL22.set(0, new Integer(1));
      LL22.listIterator(0);
}

public void testobj_D22U16() {
      LL22= new LinkedList(collecshan);
      LL22.set(0, new Integer(1));
      LL22.remove(0);
      assertTrue(LL22.size()==2);
}

public void testobj_D22U17() {
      LL22= new LinkedList(collecshan);
      LL22.set(0, new Integer(1));
      assertTrue(LL22.remove(new Integer(7))==false);
      assertTrue(LL22.indexOf(new Integer(1))==0);
}


public void testobj_D22U18() {
      LL22= new LinkedList(collecshan);
      LL22.set(0, new Integer(1));
      LL22.removeFirst();
      assertTrue(LL22.indexOf(new Integer(1))==-1);
}

public void testobj_D22U19() {
      LL22= new LinkedList(collecshan);
      LL22.set(0, new Integer(1));
      LL22.removeLast();
```

```
      assertTrue(!LL22.contains(new Integer(3)));
}

public void testobj_D22U20() {
      LL22= new LinkedList(collecshan);
      LL22.set(0, new Integer(1));
      LL22.set(0, new Integer(7));
      assertTrue(LL22.contains(new Integer(7)));
}

public void testobj_D22U21() {
      LL22= new LinkedList(collecshan);
      LL22.set(0, new Integer(1));
      assertTrue(LL22.size()==3);
}

public void testobj_D22U22() {
      LL22= new LinkedList(collecshan);
      LL22.set(0, new Integer(1));
      LL22.toArray();
}

public void testobj_D22U23() {
      Integer a[]=new Integer[10];
      LinkedList arr[]=new LinkedList[10];
      LL22= new LinkedList(collecshan);
      LL22.set(0, new Integer(1));
      LL22.toArray(a);
}


//##############23RD OBJ DEF USES START#########

public void testobj_D23U1() {
      LL23= new LinkedList(collecshan);
      LL23.size();
      LL23.add(3, new Integer(1));
      assertTrue(LL23.indexOf(new Integer(1))==3);
}

public void testobj_D23U2() {
      LL23= new LinkedList(collecshan);
      LL23.size();
      LL23.add(new String("abc"));
      assertTrue(LL23.indexOf(new String("abc"))==3);
}

public void testobj_D23U3() {
      LL23= new LinkedList(collecshan);
      LL23.size();
      LL23.addAll(collecshan);
      assertTrue(LL23.lastIndexOf(new Integer(3))==5);
```

```
}

public void testobj_D23U4() {
      LL23= new LinkedList(collecshan);
      LL23.size();
      LL23.addAll(3, collecshan);
      assertTrue(LL23.indexOf(new Integer(3))==2);
}

public void testobj_D23U5() {
      LL23= new LinkedList(collecshan);
      LL23.size();
      LL23.addFirst(new Integer(4));
      assertTrue(LL23.indexOf(new Integer(4))==0);
      assertTrue(LL23.indexOf(new Integer(7))==1);
}

public void testobj_D23U6() {
      LL23= new LinkedList(collecshan);
      LL23.size();
      LL23.addLast(new Integer(4));
      assertTrue(LL23.indexOf(new Integer(4))==3);
}

public void testobj_D23U7() {
      LL23= new LinkedList(collecshan);
      LL23.size();
      LL23.clear();
      assertTrue(LL23.isEmpty());
}

public void testobj_D23U8() {
      LL23= new LinkedList(collecshan);
      LL23.size();
      LinkedList obj=(LinkedList)LL23.clone();
      assertTrue(obj.contains(new Integer(7)));
}

public void testobj_D23U9() {
      LL23= new LinkedList(collecshan);
      LL23.size();
      assertTrue(LL23.contains(new Integer(7)));
}

public void testobj_D23U10() {
      LL23= new LinkedList(collecshan);
      LL23.size();
      assertTrue(LL23.get(0).equals(new Integer(7)));
}

public void testobj_D23U11() {
      LL23= new LinkedList(collecshan);
```

```
      LL23.size();
      assertTrue(LL23.getFirst().equals(new
Integer(7)));
}

public void testobj_D23U12() {
      LL23= new LinkedList(collecshan);
      LL23.size();
      assertTrue(LL23.getLast().equals(new
Integer(3)));
}

public void testobj_D23U13() {
      LL23= new LinkedList(collecshan);
      LL23.size();
      assertTrue(LL23.indexOf(new Integer(3))==2);
}

public void testobj_D23U14() {
      LL23= new LinkedList(collecshan);
      LL23.size();
      assertTrue(LL23.lastIndexOf(new Integer(5))==1);
}

public void testobj_D23U15() {
      LL23= new LinkedList(collecshan);
      LL23.size();
      LL23.listIterator(0);
}

public void testobj_D23U16() {
      LL23= new LinkedList(collecshan);
      LL23.size();
      LL23.remove(0);
      assertTrue(!LL23.contains(new Integer(7)));
}

public void testobj_D23U17() {
      LL23= new LinkedList(collecshan);
      LL23.size();
      assertTrue(LL23.remove(new Integer(5))==true);
      assertTrue(LL23.indexOf(new Integer(3))==1);
}


public void testobj_D23U18() {
      LL23= new LinkedList(collecshan);
      LL23.size();
      LL23.removeFirst();
      assertTrue(LL23.indexOf(new Integer(3))==1);
}
```

```
public void testobj_D23U19() {
      LL23= new LinkedList(collecshan);
      LL23.size();
      LL23.removeLast();
      assertTrue(!LL23.contains(new Integer(3)));
}

public void testobj_D23U20() {
      LL23= new LinkedList(collecshan);
      LL23.size();
      LL23.set(0, new Integer(2));
      assertTrue(!LL23.contains(new Integer(7)));
}

public void testobj_D23U21() {
      LL23= new LinkedList(collecshan);
      LL23.size();
      assertTrue(LL23.size()==3);
}

public void testobj_D23U22() {
      LL23= new LinkedList(collecshan);
      LL23.size();
      LL23.toArray();
}

public void testobj_D23U23() {
      Integer a[]=new Integer[10];
      LinkedList arr[]=new LinkedList[10];
      LL23= new LinkedList(collecshan);
      LL23.size();
      LL23.toArray(a);
}


//##############24TH OBJ DEF USES START#########

public void testobj_D24U1() {
      LL24= new LinkedList(collecshan);
      LL24.toArray();
      LL24.add(3, new Integer(1));
      assertTrue(LL24.indexOf(new Integer(1))==3);
}

public void testobj_D24U2() {
      LL24= new LinkedList(collecshan);
      LL24.toArray();
      LL24.add(new String("abc"));
      assertTrue(LL24.indexOf(new String("abc"))==3);
}

public void testobj_D24U3() {
```

```
        LL24= new LinkedList(collecshan);
        LL24.toArray();
        LL24.addAll(collecshan);
        assertTrue(LL24.lastIndexOf(new Integer(3))==5);
    }

    public void testobj_D24U4() {
        LL24= new LinkedList(collecshan);
        LL24.toArray();
        LL24.addAll(3, collecshan);
        assertTrue(LL24.indexOf(new Integer(3))==2);
    }

    public void testobj_D24U5() {
        LL24= new LinkedList(collecshan);
        LL24.toArray();
        LL24.addFirst(new Integer(4));
        assertTrue(LL24.indexOf(new Integer(4))==0);
        assertTrue(LL24.indexOf(new Integer(7))==1);
    }

    public void testobj_D24U6() {
        LL24= new LinkedList(collecshan);
        LL24.toArray();
        LL24.addLast(new Integer(4));
        assertTrue(LL24.indexOf(new Integer(4))==3);
    }

    public void testobj_D24U7() {
        LL24= new LinkedList(collecshan);
        LL24.toArray();
        LL24.clear();
        assertTrue(LL24.isEmpty());
    }

    public void testobj_D24U8() {
        LL24= new LinkedList(collecshan);
        LL24.toArray();
        LinkedList obj=(LinkedList)LL24.clone();
        assertTrue(obj.contains(new Integer(7)));
    }

    public void testobj_D24U9() {
        LL24= new LinkedList(collecshan);
        LL24.toArray();
        assertTrue(LL24.contains(new Integer(7)));
    }

    public void testobj_D24U10() {
        LL24= new LinkedList(collecshan);
        LL24.toArray();
        assertTrue(LL24.get(0).equals(new Integer(7)));
```

```
}

public void testobj_D24U11() {
      LL24= new LinkedList(collecshan);
      LL24.toArray();
      assertTrue(LL24.getFirst().equals(new
Integer(7)));
}

public void testobj_D24U12() {
      LL24= new LinkedList(collecshan);
      LL24.toArray();
      assertTrue(LL24.getLast().equals(new
Integer(3)));
}

public void testobj_D24U13() {
      LL24= new LinkedList(collecshan);
      LL24.toArray();
      assertTrue(LL24.indexOf(new Integer(3))==2);
}

public void testobj_D24U14() {
      LL24= new LinkedList(collecshan);
      LL24.toArray();
      assertTrue(LL24.lastIndexOf(new Integer(5))==1);
}

public void testobj_D24U15() {
      LL24= new LinkedList(collecshan);
      LL24.toArray();
      LL24.listIterator(0);
}

public void testobj_D24U16() {
      LL24= new LinkedList(collecshan);
      LL24.toArray();
      LL24.remove(0);
      assertTrue(!LL24.contains(new Integer(7)));
}

public void testobj_D24U17() {
      LL24= new LinkedList(collecshan);
      LL24.toArray();
      assertTrue(LL24.remove(new Integer(5))==true);
      assertTrue(LL24.indexOf(new Integer(3))==1);
}


public void testobj_D24U18() {
      LL24= new LinkedList(collecshan);
      LL24.toArray();
```

```
      LL24.removeFirst();
      assertTrue(LL24.indexOf(new Integer(3))==1);
}

public void testobj_D24U19() {
      LL24= new LinkedList(collecshan);
      LL24.toArray();
      LL24.removeLast();
      assertTrue(!LL24.contains(new Integer(3)));
}

public void testobj_D24U20() {
      LL24= new LinkedList(collecshan);
      LL24.toArray();
      LL24.set(0, new Integer(2));
      assertTrue(!LL24.contains(new Integer(7)));
}

public void testobj_D24U21() {
      LL24= new LinkedList(collecshan);
      LL24.toArray();
      assertTrue(LL24.size()==3);
}

public void testobj_D24U22() {
      LL24= new LinkedList(collecshan);
      LL24.toArray();
      LL24.toArray();
}

public void testobj_D24U23() {
      Integer a[]=new Integer[10];
      LinkedList arr[]=new LinkedList[10];
      LL24= new LinkedList(collecshan);
      LL24.toArray();
      LL24.toArray(a);
}


//##############25TH OBJ DEF USES START#########

public void testobj_D25U1() {
      LL25= new LinkedList(collecshan);
      LL25.toArray(new Integer[10]);
      LL25.add(3, new Integer(1));
      assertTrue(LL25.indexOf(new Integer(1))==3);
}

public void testobj_D25U2() {
      LL25= new LinkedList(collecshan);
      LL25.toArray(new Integer[10]);
      LL25.add(new String("abc"));
```

```
        assertTrue(LL25.indexOf(new String("abc"))==3);
    }

    public void testobj_D25U3() {
        LL25= new LinkedList(collecshan);
        LL25.toArray(new Integer[10]);
        LL25.addAll(collecshan);
        assertTrue(LL25.lastIndexOf(new Integer(3))==5);
    }

    public void testobj_D25U4() {
        LL25= new LinkedList(collecshan);
        LL25.toArray(new Integer[10]);
        LL25.addAll(3, collecshan);
        assertTrue(LL25.indexOf(new Integer(3))==2);
    }

    public void testobj_D25U5() {
        LL25= new LinkedList(collecshan);
        LL25.toArray(new Integer[10]);
        LL25.addFirst(new Integer(4));
        assertTrue(LL25.indexOf(new Integer(4))==0);
        assertTrue(LL25.indexOf(new Integer(7))==1);
    }

    public void testobj_D25U6() {
        LL25= new LinkedList(collecshan);
        LL25.toArray(new Integer[10]);
        LL25.addLast(new Integer(4));
        assertTrue(LL25.indexOf(new Integer(4))==3);
    }

    public void testobj_D25U7() {
        LL25= new LinkedList(collecshan);
        LL25.toArray(new Integer[10]);
        LL25.clear();
        assertTrue(LL25.isEmpty());
    }

    public void testobj_D25U8() {
        LL25= new LinkedList(collecshan);
        LL25.toArray(new Integer[10]);
        LinkedList obj=(LinkedList)LL25.clone();
        assertTrue(obj.contains(new Integer(7)));
    }

    public void testobj_D25U9() {
        LL25= new LinkedList(collecshan);
        LL25.toArray(new Integer[10]);
        assertTrue(LL25.contains(new Integer(7)));
    }
```

```
public void testobj_D25U10() {
      LL25= new LinkedList(collecshan);
      LL25.toArray(new Integer[10]);
      assertTrue(LL25.get(0).equals(new Integer(7)));
}

public void testobj_D25U11() {
      LL25= new LinkedList(collecshan);
      LL25.toArray(new Integer[10]);
      assertTrue(LL25.getFirst().equals(new
Integer(7)));
}

public void testobj_D25U12() {
      LL25= new LinkedList(collecshan);
      LL25.toArray(new Integer[10]);
      assertTrue(LL25.getLast().equals(new
Integer(3)));
}

public void testobj_D25U13() {
      LL25= new LinkedList(collecshan);
      LL25.toArray(new Integer[10]);
      assertTrue(LL25.indexOf(new Integer(3))==2);
}

public void testobj_D25U14() {
      LL25= new LinkedList(collecshan);
      LL25.toArray(new Integer[10]);
      assertTrue(LL25.lastIndexOf(new Integer(5))==1);
}

public void testobj_D25U15() {
      LL25= new LinkedList(collecshan);
      LL25.toArray(new Integer[10]);
      LL25.listIterator(0);
}

public void testobj_D25U16() {
      LL25= new LinkedList(collecshan);
      LL25.toArray(new Integer[10]);
      LL25.remove(0);
      assertTrue(!LL25.contains(new Integer(7)));
}

public void testobj_D25U17() {
      LL25= new LinkedList(collecshan);
      LL25.toArray(new Integer[10]);
      assertTrue(LL25.remove(new Integer(5))==true);
      assertTrue(LL25.indexOf(new Integer(3))==1);
}
```

```
public void testobj_D25U18() {
      LL25= new LinkedList(collecshan);
      LL25.toArray(new Integer[10]);
      LL25.removeFirst();
      assertTrue(LL25.indexOf(new Integer(3))==1);
}

public void testobj_D25U19() {
      LL25= new LinkedList(collecshan);
      LL25.toArray(new Integer[10]);
      LL25.removeLast();
      assertTrue(!LL25.contains(new Integer(3)));
}

public void testobj_D25U20() {
      LL25= new LinkedList(collecshan);
      LL25.toArray(new Integer[10]);
      LL25.set(0, new Integer(2));
      assertTrue(!LL25.contains(new Integer(7)));
}

public void testobj_D25U21() {
      LL25= new LinkedList(collecshan);
      LL25.toArray(new Integer[10]);
      assertTrue(LL25.size()==3);
}

public void testobj_D25U22() {
      LL25= new LinkedList(collecshan);
      LL25.toArray(new Integer[10]);
      LL25.toArray();
}

public void testobj_D25U23() {
      Integer a[]=new Integer[10];
      LinkedList arr[]=new LinkedList[10];
      LL25= new LinkedList(collecshan);
      LL25.toArray(new Integer[10]);
      LL25.toArray(a);
}



//############ALL OBJt DEF USES START##############

//##########1ST OBJt DEF USE STARTS##############


public void testobjt_D1U1() {
      lobjt1= new LinkedList();
      objt1= (LinkedList)lobjt1.clone();
```

```
        lobjt1.add(0, objt1);
        assertTrue(lobjt1.size()==1);
    }

    public void testobjt_D1U2() {
        lobjt1= new LinkedList();
        objt1= (LinkedList)lobjt1.clone();
        lobjt1.add(objt1);
        assertTrue(lobjt1.contains(new LinkedList()));
    }

    public void testobjt_D1U3() {
        lobjt1= new LinkedList();
        objt1= (LinkedList)lobjt1.clone();
        lobjt1.addFirst(objt1);
        assertTrue(lobjt1.indexOf(new LinkedList())==0);
    }

    public void testobjt_D1U4() {
        lobjt1= new LinkedList();
        objt1= (LinkedList)lobjt1.clone();
        lobjt1.addLast(objt1);
        assertTrue(lobjt1.indexOf(new LinkedList())==0);
    }

    public void testobjt_D1U5() {
        lobjt1= new LinkedList();
        objt1= (LinkedList)lobjt1.clone();
        assertTrue(!lobjt1.contains(objt1));
    }

    public void testobjt_D1U6() {
        lobjt1= new LinkedList();
        objt1= (LinkedList)lobjt1.clone();
        assertTrue(lobjt1.indexOf(objt1)==-1);
    }

    public void testobjt_D1U7() {
        lobjt1= new LinkedList();
        objt1= (LinkedList)lobjt1.clone();
        assertTrue(lobjt1.lastIndexOf(objt1)==-1);
    }

    public void testobjt_D1U8() {
        lobjt1= new LinkedList();
        objt1= (LinkedList)lobjt1.clone();
        assertTrue(!lobjt1.remove(objt1));
    }

    public void testobjt_D1U9() {
        int jk=0;
        lobjt1= new LinkedList();
```

```
      objt1= (LinkedList)lobjt1.clone();
      try {
      lobjt1.set(0, objt1);
      } catch(IndexOutOfBoundsException iobe) {
      jk=1;
      }
      if (jk==0) assertTrue(false);
}


//##########2ND OBJt DEF USE STARTS##############


public void testobjt_D2U1() {
      lobjt2= new LinkedList(collecshan);
      objt2= (Integer)lobjt2.get(0);
      lobjt2.add(0, objt2);
      assertTrue(lobjt2.size()==4);
}

public void testobjt_D2U2() {
      lobjt2= new LinkedList(collecshan);
      objt2= (Integer)lobjt2.get(0);
      lobjt2.add(objt2);
      assertTrue(lobjt2.lastIndexOf(new
Integer(7))==3);
}

public void testobjt_D2U3() {
      lobjt2= new LinkedList(collecshan);
      objt2= (Integer)lobjt2.get(0);
      lobjt2.addFirst(objt2);
      assertTrue(lobjt2.lastIndexOf(new
Integer(7))==1);
}

public void testobjt_D2U4() {
      lobjt2= new LinkedList(collecshan);
      objt2= (Integer)lobjt2.get(0);
      lobjt2.addLast(objt2);
      assertTrue(lobjt2.lastIndexOf(new
Integer(7))==3);
}

public void testobjt_D2U5() {
      lobjt2= new LinkedList(collecshan);
      objt2= (Integer)lobjt2.get(0);
      assertTrue(lobjt2.contains(objt2));
}

public void testobjt_D2U6() {
      lobjt2= new LinkedList(collecshan);
```

```
      objt2= (Integer)lobjt2.get(0);
      assertTrue(lobjt2.indexOf(objt2)==0);
}

public void testobjt_D2U7() {
      lobjt2= new LinkedList(collecshan);
      objt2= (Integer)lobjt2.get(0);
      assertTrue(lobjt2.lastIndexOf(objt2)==0);
}

public void testobjt_D2U8() {
      lobjt2= new LinkedList(collecshan);
      objt2= (Integer)lobjt2.get(0);
      assertTrue(lobjt2.remove(objt2));
}

public void testobjt_D2U9() {
      lobjt2= new LinkedList(collecshan);
      objt2= (Integer)lobjt2.get(0);
      lobjt2.set(1, objt2);
      assertTrue(lobjt2.lastIndexOf(objt2)==1);
}


//##########3ND OBJt DEF USE STARTS##############


public void testobjt_D3U1() {
      lobjt3= new LinkedList(collecshan);
      objt3= (Integer)lobjt3.getFirst();
      lobjt3.add(0, objt3);
      assertTrue(lobjt3.size()==4);
}

public void testobjt_D3U2() {
      lobjt3= new LinkedList(collecshan);
      objt3= (Integer)lobjt3.getFirst();
      lobjt3.add(objt3);
      assertTrue(lobjt3.lastIndexOf(new
Integer(7))==3);
}

public void testobjt_D3U3() {
      lobjt3= new LinkedList(collecshan);
      objt3= (Integer)lobjt3.getFirst();
      lobjt3.addFirst(objt3);
      assertTrue(lobjt3.lastIndexOf(new
Integer(7))==1);
}

public void testobjt_D3U4() {
      lobjt3= new LinkedList(collecshan);
```

```
      objt3= (Integer)lobjt3.getFirst();
      lobjt3.addLast(objt3);
      assertTrue(lobjt3.lastIndexOf(new
Integer(7))==3);
}

public void testobjt_D3U5() {
      lobjt3= new LinkedList(collecshan);
      objt3= (Integer)lobjt3.getFirst();
      assertTrue(lobjt3.contains(objt3));
}

public void testobjt_D3U6() {
      lobjt3= new LinkedList(collecshan);
      objt3= (Integer)lobjt3.getFirst();
      assertTrue(lobjt3.indexOf(objt3)==0);
}

public void testobjt_D3U7() {
      lobjt3= new LinkedList(collecshan);
      objt3= (Integer)lobjt3.getFirst();
      assertTrue(lobjt3.lastIndexOf(objt3)==0);
}

public void testobjt_D3U8() {
      lobjt3= new LinkedList(collecshan);
      objt3= (Integer)lobjt3.getFirst();
      assertTrue(lobjt3.remove(objt3));
}

public void testobjt_D3U9() {
      lobjt3= new LinkedList(collecshan);
      objt3= (Integer)lobjt3.getFirst();
      lobjt3.set(1, objt3);
      assertTrue(lobjt3.lastIndexOf(objt3)==1);
}


//###########4th OBJt DEF USE STARTS###############


public void testobjt_D4U1() {
      lobjt4= new LinkedList(collecshan);
      objt4= (Integer)lobjt4.getLast();
      lobjt4.add(0, objt4);
      assertTrue(lobjt4.size()==4);
}

public void testobjt_D4U2() {
      lobjt4= new LinkedList(collecshan);
      objt4= (Integer)lobjt4.getLast();
      lobjt4.add(objt4);
```

```
      assertTrue(lobjt4.lastIndexOf(new
Integer(3))==3);
}

public void testobjt_D4U3() {
      lobjt4= new LinkedList(collecshan);
      objt4= (Integer)lobjt4.getLast();
      lobjt4.addFirst(objt4);
      assertTrue(lobjt4.lastIndexOf(new
Integer(7))==1);
}

public void testobjt_D4U4() {
      lobjt4= new LinkedList(collecshan);
      objt4= (Integer)lobjt4.getLast();
      lobjt4.addLast(objt4);
      assertTrue(lobjt4.lastIndexOf(new
Integer(3))==3);
}

public void testobjt_D4U5() {
      lobjt4= new LinkedList(collecshan);
      objt4= (Integer)lobjt4.getLast();
      assertTrue(lobjt4.contains(objt4));
}

public void testobjt_D4U6() {
      lobjt4= new LinkedList(collecshan);
      objt4= (Integer)lobjt4.getLast();
      assertTrue(lobjt4.indexOf(objt4)==2);
}

public void testobjt_D4U7() {
      lobjt4= new LinkedList(collecshan);
      objt4= (Integer)lobjt4.getLast();
      assertTrue(lobjt4.lastIndexOf(objt4)==2);
}

public void testobjt_D4U8() {
      lobjt4= new LinkedList(collecshan);
      objt4= (Integer)lobjt4.getLast();
      assertTrue(lobjt4.remove(objt4));
}

public void testobjt_D4U9() {
      lobjt4= new LinkedList(collecshan);
      objt4= (Integer)lobjt4.getLast();
      lobjt4.set(1, objt4);
      assertTrue(lobjt4.indexOf(new Integer(3))==1);
}
```

```
//##########5th OBJt DEF USE STARTS##############


public void testobjt_D5U1() {
      lobjt5= new LinkedList(collecshan);
      objt5= (Integer)lobjt5.remove(0);
      lobjt5.add(0, objt5);
      assertTrue(lobjt5.size()==3);
}

public void testobjt_D5U2() {
      lobjt5= new LinkedList(collecshan);
      objt5= (Integer)lobjt5.remove(0);
      lobjt5.add(objt5);
      assertTrue(lobjt5.indexOf(new Integer(7))==2);
}

public void testobjt_D5U3() {
      lobjt5= new LinkedList(collecshan);
      objt5= (Integer)lobjt5.remove(0);
      lobjt5.addFirst(objt5);
      assertTrue(lobjt5.lastIndexOf(new
Integer(7))==0);
}

public void testobjt_D5U4() {
      lobjt5= new LinkedList(collecshan);
      objt5= (Integer)lobjt5.remove(0);
      lobjt5.addLast(objt5);
      assertTrue(lobjt5.lastIndexOf(new
Integer(7))==2);
}

public void testobjt_D5U5() {
      lobjt5= new LinkedList(collecshan);
      objt5= (Integer)lobjt5.remove(0);
      assertTrue(!lobjt5.contains(objt5));
}

public void testobjt_D5U6() {
      lobjt5= new LinkedList(collecshan);
      objt5= (Integer)lobjt5.remove(0);
      assertTrue(lobjt5.indexOf(objt5)==-1);
}

public void testobjt_D5U7() {
      lobjt5= new LinkedList(collecshan);
      objt5= (Integer)lobjt5.remove(0);
      assertTrue(lobjt5.lastIndexOf(objt5)==-1);
}

public void testobjt_D5U8() {
```

```
      lobjt5= new LinkedList(collecshan);
      objt5= (Integer)lobjt5.remove(0);
      assertTrue(!lobjt5.remove(objt5));
}

public void testobjt_D5U9() {
      lobjt5= new LinkedList(collecshan);
      objt5= (Integer)lobjt5.remove(0);
      lobjt5.set(1, objt5);
      assertTrue(lobjt5.indexOf(objt5)==1);
}


//##########6th OBJt DEF USE STARTS##############


public void testobjt_D6U1() {
      lobjt6= new LinkedList(collecshan);
      objt6= (Integer)lobjt6.removeFirst();
      lobjt6.add(0, objt6);
      assertTrue(lobjt6.size()==3);
}

public void testobjt_D6U2() {
      lobjt6= new LinkedList(collecshan);
      objt6= (Integer)lobjt6.removeFirst();
      lobjt6.add(objt6);
      assertTrue(lobjt6.indexOf(new Integer(7))==2);
}

public void testobjt_D6U3() {
      lobjt6= new LinkedList(collecshan);
      objt6= (Integer)lobjt6.removeFirst();
      lobjt6.addFirst(objt6);
      assertTrue(lobjt6.lastIndexOf(new
Integer(7))==0);
}

public void testobjt_D6U4() {
      lobjt6= new LinkedList(collecshan);
      objt6= (Integer)lobjt6.removeFirst();
      lobjt6.addLast(objt6);
      assertTrue(lobjt6.lastIndexOf(new
Integer(7))==2);
}

public void testobjt_D6U5() {
      lobjt6= new LinkedList(collecshan);
      objt6= (Integer)lobjt6.removeFirst();
      assertTrue(!lobjt6.contains(objt6));
}
```

```
public void testobjt_D6U6() {
      lobjt6= new LinkedList(collecshan);
      objt6= (Integer)lobjt6.removeFirst();
      assertTrue(lobjt6.indexOf(objt6)==-1);
}

public void testobjt_D6U7() {
      lobjt6= new LinkedList(collecshan);
      objt6= (Integer)lobjt6.removeFirst();
      assertTrue(lobjt6.lastIndexOf(objt6)==-1);
}

public void testobjt_D6U8() {
      lobjt6= new LinkedList(collecshan);
      objt6= (Integer)lobjt6.removeFirst();
      assertTrue(!lobjt6.remove(objt6));
}

public void testobjt_D6U9() {
      lobjt6= new LinkedList(collecshan);
      objt6= (Integer)lobjt6.removeFirst();
      lobjt6.set(1, objt6);
      assertTrue(lobjt6.indexOf(objt6)==1);
}

//##########7th OBJt DEF USE STARTS##############


public void testobjt_D7U1() {
      lobjt7= new LinkedList(collecshan);
      objt7= (Integer)lobjt7.removeLast();
      lobjt7.add(0, objt7);
      assertTrue(lobjt7.size()==3);
}

public void testobjt_D7U2() {
      lobjt7= new LinkedList(collecshan);
      objt7= (Integer)lobjt7.removeLast();
      lobjt7.add(objt7);
      assertTrue(lobjt7.indexOf(new Integer(3))==2);
}

public void testobjt_D7U3() {
      lobjt7= new LinkedList(collecshan);
      objt7= (Integer)lobjt7.removeLast();
      lobjt7.addFirst(objt7);
      assertTrue(lobjt7.lastIndexOf(new
Integer(3))==0);
}

public void testobjt_D7U4() {
      lobjt7= new LinkedList(collecshan);
```

```
      objt7= (Integer)lobjt7.removeLast();
      lobjt7.addLast(objt7);
      assertTrue(lobjt7.lastIndexOf(new
Integer(3))==2);
}

public void testobjt_D7U5() {
      lobjt7= new LinkedList(collecshan);
      objt7= (Integer)lobjt7.removeLast();
      assertTrue(!lobjt7.contains(objt7));
}

public void testobjt_D7U6() {
      lobjt7= new LinkedList(collecshan);
      objt7= (Integer)lobjt7.removeLast();
      assertTrue(lobjt7.indexOf(objt7)==-1);
}

public void testobjt_D7U7() {
      lobjt7= new LinkedList(collecshan);
      objt7= (Integer)lobjt7.removeLast();
      assertTrue(lobjt7.lastIndexOf(objt7)==-1);
}

public void testobjt_D7U8() {
      lobjt7= new LinkedList(collecshan);
      objt7= (Integer)lobjt7.removeLast();
      assertTrue(!lobjt7.remove(objt7));
}

public void testobjt_D7U9() {
      lobjt7= new LinkedList(collecshan);
      objt7= (Integer)lobjt7.removeLast();
      lobjt7.set(1, objt7);
      assertTrue(lobjt7.indexOf(objt7)==1);
}


//##########8th OBJt DEF USE STARTS##############


public void testobjt_D8U1() {
      lobjt8= new LinkedList(collecshan);
      objt8= (Integer)lobjt8.set(0, new Integer(2));
      lobjt8.add(0, objt8);
      assertTrue(lobjt8.size()==4);
}

public void testobjt_D8U2() {
      lobjt8= new LinkedList(collecshan);
      objt8= (Integer)lobjt8.set(0, new Integer(2));
      lobjt8.add(objt8);
```

```
      assertTrue(lobjt8.lastIndexOf(new
Integer(7))==3);
}

public void testobjt_D8U3() {
      lobjt8= new LinkedList(collecshan);
      objt8= (Integer)lobjt8.set(0, new Integer(2));
      lobjt8.addFirst(objt8);
      assertTrue(lobjt8.lastIndexOf(new
Integer(2))==1);
}

public void testobjt_D8U4() {
      lobjt8= new LinkedList(collecshan);
      objt8= (Integer)lobjt8.set(0, new Integer(2));
      lobjt8.addLast(objt8);
      assertTrue(lobjt8.lastIndexOf(new
Integer(2))==0);
}

public void testobjt_D8U5() {
      lobjt8= new LinkedList(collecshan);
      objt8= (Integer)lobjt8.set(0, new Integer(2));
      assertTrue(!lobjt8.contains(objt8));
}

public void testobjt_D8U6() {
      lobjt8= new LinkedList(collecshan);
      objt8= (Integer)lobjt8.set(0, new Integer(2));
      assertTrue(lobjt8.indexOf(objt8)==-1);
}

public void testobjt_D8U7() {
      lobjt8= new LinkedList(collecshan);
      objt8= (Integer)lobjt8.set(0, new Integer(2));
      assertTrue(lobjt8.lastIndexOf(objt8)==-1);
}

public void testobjt_D8U8() {
      lobjt8= new LinkedList(collecshan);
      objt8= (Integer)lobjt8.set(0, new Integer(2));
      assertTrue(!lobjt8.remove(objt8));
}

public void testobjt_D8U9() {
      lobjt8= new LinkedList(collecshan);
      objt8= (Integer)lobjt8.set(0, new Integer(2));
      lobjt8.set(1, objt8);
      assertTrue(lobjt8.indexOf(objt8)==1);
}
```

```
//########All int def uses start here###########

//#####1st int def use###########


public void testint_D1U1() {
      lint1= new LinkedList(collecshan);
      int1= lint1.indexOf(new Integer(7));
      lint1.add(int1, new Integer(2));
      assertTrue(lint1.indexOf(new Integer(3))==3);
}

public void testint_D1U2() {
      lint1= new LinkedList(collecshan);
      int1= lint1.indexOf(new Integer(7));
      lint1.addAll(int1, collecshan);
      assertTrue(lint1.indexOf(new Integer(7))==0);
}

public void testint_D1U3() {
      lint1= new LinkedList(collecshan);
      int1= lint1.indexOf(new Integer(7));
      assertTrue(lint1.get(int1).equals(new
Integer(7)));
}

public void testint_D1U4() {
      lint1= new LinkedList(collecshan);
      int1= lint1.indexOf(new Integer(7));
      assertTrue(lint1.remove(int1).equals(new
Integer(7)));
      assertTrue(!lint1.contains(new Integer(7)));
}

public void testint_D1U5() {
      lint1= new LinkedList(collecshan);
      int1= lint1.indexOf(new Integer(7));
      lint1.listIterator(int1);
}

public void testint_D1U6() {
      lint1= new LinkedList(collecshan);
      int1= lint1.indexOf(new Integer(7));
      assertTrue(lint1.set(int1, new
Integer(2)).equals(new Integer(7)));
}


//#####2nd int def use###########


public void testint_D2U1() {
```

```
      lint2= new LinkedList(collecshan);
      int2= lint2.lastIndexOf(new Integer(3));
      lint2.add(int2, new Integer(2));
      assertTrue(lint2.indexOf(new Integer(2))==2);
}

public void testint_D2U2() {
      lint2= new LinkedList(collecshan);
      int2= lint2.lastIndexOf(new Integer(3));
      lint2.addAll(int2, collecshan);
      assertTrue(lint2.indexOf(new Integer(3))==4);
}

public void testint_D2U3() {
      lint2= new LinkedList(collecshan);
      int2= lint2.lastIndexOf(new Integer(3));
      assertTrue(lint2.get(int2).equals(new
Integer(3)));
}

public void testint_D2U4() {
      lint2= new LinkedList(collecshan);
      int2= lint2.lastIndexOf(new Integer(3));
      assertTrue(lint2.remove(int2).equals(new
Integer(3)));
      assertTrue(!lint2.contains(new Integer(3)));
}

public void testint_D2U5() {
      lint2= new LinkedList(collecshan);
      int2= lint2.lastIndexOf(new Integer(3));
      lint2.listIterator(int2);
}

public void testint_D2U6() {
      lint2= new LinkedList(collecshan);
      int2= lint2.lastIndexOf(new Integer(3));
      assertTrue(lint2.set(int2, new
Integer(2)).equals(new Integer(3)));
}

//#####3rd int def use###########


public void testint_D3U1() {
      lint3= new LinkedList(collecshan);
      int3= lint3.size();
      lint3.add(int3, new Integer(2));
      assertTrue(lint3.indexOf(new Integer(2))==3);
}

public void testint_D3U2() {
```

```
      lint3= new LinkedList(collecshan);
      int3= lint3.size();
      lint3.addAll(int3, collecshan);
      assertTrue(lint3.lastIndexOf(new Integer(3))==5);
}

public void testint_D3U3() {
      lint3= new LinkedList(collecshan);
      int3= lint3.size();
      assertTrue(lint3.get(int3-1).equals(new
Integer(3)));
}

public void testint_D3U4() {
      lint3= new LinkedList(collecshan);
      int3= lint3.size();
      assertTrue(lint3.remove(int3-1).equals(new
Integer(3)));
      assertTrue(!lint3.contains(new Integer(3)));
}

public void testint_D3U5() {
      lint3= new LinkedList(collecshan);
      int3= lint3.size();
      lint3.listIterator(int3-1);
}

public void testint_D3U6() {
      lint3= new LinkedList(collecshan);
      int3= lint3.size();
      assertTrue(lint3.set(int3-1, new
Integer(2)).equals(new Integer(3)));
}

}
```

CURRICULUM VITA

Mahesh Mungara was born in AP, India on February 19, 1980. He completed his B.Tech in Computer Science from Andhra University, Visakhapatnam, India in May, 2001.

He joined Virginia Polytechnic Institute and State University in Fall 2001 and will be graduating with a Master of Science degree in Computer Science in December 2003.