toString() :

```
public class Test1
{
    String name;
    int id;

    @Override
    public String toString() {
        return name+", "+id;
    }

    public static void main(String[] args)
    {
        Test1 t1=new Test1();

        t1.name="deepak";
        t1.id=101;

        System.out.println(t1); //Test1@xxxxx(if no toString() Method)

        //-----------------------------------------

        ArrayList al=new ArrayList();

        al.add("aaa");
        al.add("bbb");
        al.add("ccc");
        al.add("ddd");

        System.out.println(al); //[aaa, bbb, ccc, ddd]
    }
}
```

=> Cursors :-

-In java, whenever we print the object reference, internally JVM will call toString() method of Object class. In case of simple object it will print ClassName@referencevalue but in case of printing Collection object it will print the elements present in Collection object.

-When we print the collection object it will retrieve all the elements at one time but we want to retrieve the elements one by one then we have to use cursors

-> Types of cursors :-

1. Enumeration
2. Iterator
3. ListIterator

=> Enumeration :-

-Enumeration is the cursor which is used to get the elements one by one from the collection object
-Enumeration was introduced in JDK 1.0 version
-Enumeration is used only for legacy class

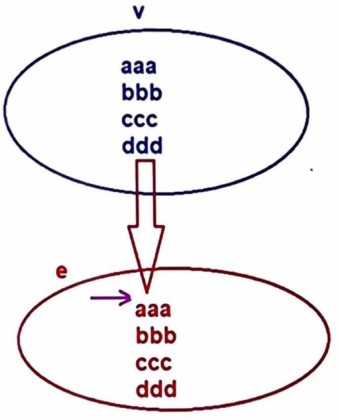
-> Steps "how to use" enumeration cursor :-

1. Create Enumeration Cursor Object
-> public Enumeration elements()
(this method is present in Vector & Stack legacy class)

2. Read one by one all the elements from Enumeration Cursor
-> public boolean hasMoreElements()
-> public Object nextElement()
(these methods are present in Enumeration interface)

```
public class Test2
{
    public static void main(String[] args)
    {
        Vector v=new Vector();

        v.addElement("aaa");
        v.addElement("bbb");
        v.addElement("ccc");
        v.addElement("ddd");

        Enumeration e=v.elements();   // elements() Method is present in vector and stack, Which retrun Enumeration

        while(e.hasMoreElements())
        {
            System.out.println(e.nextElement());
        }
    }
}
```

```
public class Test2
{
   public static void main(String[] args)
   {
      Vector v=new Vector();

      v.addElement("aaa");
      v.addElement("bbb");
      v.addElement("ccc");
      v.addElement("ddd");

      Enumeration e=v.elements();
      while(e.hasMoreElements())
      {
         System.out.println(e.nextElement());
      }
   }
}
```



-> Limitations of Enumeration :-

1. It can be used only with Legacy class and thus it is not universal cursor
2. By using enumeration cursor we can only perform read operation but not update or remove operation
3. It can be used to traverse the elements only in forward direction

=> Iterator :-

-Iterator is a cursor which is used to get the elements one by one from the collection object

-It is universal cursor which means that we can use it with all collection objects

-It can be used for read and remove operation

-It was introduced in JDK 1.2 version

-> Steps "how to use" iterator cursor :-

1. Create Iterator cursor object :-

-> public Iterator iterator()

(this method is present for every collection object)

2. Read one by one all the elements from iterator cursor :-

-> public boolean hasNext()

-> public Object next()

-> public void remove()

(these methods are present in Iterator interface)

-> Limitations of Iterator cursor :-

1. It can be used only for read and remove operation but not for replacement or addition operation
2. It can be used to iterate the elements only in forward direction

```
public class Test {
    public static void main(String[] args) {
        ArrayList<String> list = new ArrayList() ;
        list.add("Arun") ;
        list.add("Rahul") ;
        list.add("Bahvek") ;
        list.add("Divas") ;

        Iterator itr = list.iterator() ;
        while(itr.hasNext()){
            System.out.println(itr.next());
        }
    }
}
```

==========

```
public class Test {
    public static void main(String[] args) {
        ArrayList<String> list = new ArrayList() ;
        list.add("Arun") ;
        list.add("Rahul") ;
        list.add("Bhavek") ;
        list.add("Divas") ;

        Iterator itr = list.iterator() ;
        while(itr.hasNext()){
            String str = (String)itr.next() ;
            if(str.equals("Bhavek")){
                itr.remove();
            }
        }
        System.out.println(list);
    }
}
```

=> ListIterator :-

-ListIterator is a cursor which is used to get the elements one by one from collection object

-ListIterator is bi-directional cursor which means it can be used to traverse the elements in forward or backword direction

-It can be used to read, remove, insert and replace operations

-It was introduced in JDK 1.2 version

-> Steps "how to use" ListIterator cursor :-

1. Create ListIterator cursor object :-

-> public ListIterator listIterator();

(which is present in only List implementation classes)

2. Read one by one all the elements from ListIterator cursor :-

-> public boolean hasNext()

-> public Object next()

-> public int nextIndex()

(above methods are used to traverse the elements in forward direction)

-> public boolean hasPrevious()(above methods are used to traverse the elements in backword direction)

-> public void remove()

-> public void add(Object obj)

-> public void set(Object obj)

(above methods are used to remove, add and replace operations)

-> Limitations of ListIterator cursor :-

1. It can be used only with List implemented classes thus it is not universal cursor

-> public Object previous()

-> public int previousIndex()

=> What is difference between Enumeration, Iterator & ListIterator :-

1. Enumeration can be used only with legacy classes
   Iterator can be used for any collection object
   ListIterator can be used only for List implemented classes

2. Enumeration can be used to traverse the elements only in forward direction
   Iterator can be used to traverse the elements only in forward direction
   ListIterator can be used to traverse the elements in forward and backword direction

3. Enumeration is used only for read operation
   Iterator can be used for read and remove operation
   ListIterator can be used for read, remove, add and replace operations

```
public class Test4
{
    public static void main(String[] args)
    {
        ArrayList al=new ArrayList();

        al.add("aaa");
        al.add("bbb");
        al.add("ccc");
        al.add("ddd");

        //----------forward direction----------------------
        ListIterator li=al.listIterator();
        while(li.hasNext())
        {
            System.out.println(li.next());
        }

        System.out.println("------------------------------");

        //-------------backword direction-----------------
        while(li.hasPrevious())
        {
            System.out.println(li.previous());
        }
    }
}
```