

COLLECTION EXAMPLES

```
1)
import java.util.*;
class arraylist
{
    public static void main(String args[])
    {
        ArrayList<String> arraylist = new ArrayList<String>();
        arraylist.add("Item 0");
        arraylist.add("Item 2");
        arraylist.add("Item 3");
        arraylist.add("Item 4");
        arraylist.add("Item 5");
        arraylist.add("Item 6");
        arraylist.add(1, "Item 1");

        System.out.println("\nUsing the add method");
        System.out.println(arraylist);
        arraylist.remove("Item 5");
        System.out.println(arraylist);

        System.out.println("\nUsing the Iterator interface");
        String s;
        Iterator e = arraylist.iterator();
        while (e.hasNext())
        {
            s = (String)e.next();
            System.out.println(s);
        }
    }
}
```

```
Compile      : javac arraylist.java
Run          : java arraylist
```

2)

```
import java.util.*;
class arrays
{
    public static void main(String args[])
    {
        int array[] = new int[10];
        for(int loop_index = 9; loop_index > 0; loop_index--)
            array[loop_index] = -loop_index;
        for(int loop_index = 0; loop_index < array.length; loop_index++)
            System.out.print(array[loop_index] + " ");
        System.out.println();
        Arrays.sort(array);
        for(int loop_index = 0; loop_index < array.length; loop_index++)
            System.out.print(array[loop_index] + " ");
        System.out.println();
        System.out.print("Found -5 at position " +
            Arrays.binarySearch(array, -5));
    }
}
```

Compile : javac array.java

Run : java array

3)

```
import java.util.*;
class comparator
{
    public static void main(String args[])
    {
        TreeSet<String> treeset = new TreeSet<String>(new NewComparator());
        treeset.add("Item 0");
        treeset.add("Item 1");
        treeset.add("Item 2");
        treeset.add("Item 3");
        treeset.add("Item 4");
        treeset.add("Item 5");
        treeset.add("Item 6");
        Iterator iterator = treeset.iterator();
        while(iterator.hasNext()) {
            System.out.println(iterator.next());
        }
    }
}
```

```

class NewComparator implements Comparator
{
public int compare(Object obj1, Object obj2)
{
if (((String) obj1).equals("Item 3")) return -1;
return ((String) obj1).compareTo((String) obj2);
}
}

```

```

Compile      : javac comparator.java
Run          : java comparator

```

```

4)
import java.util.*;
class hashmap
{
public static void main(String args[])
{
HashMap<String, String> hashmap1 = new HashMap<String, String>();
hashmap1.put("Item 0", "Value 0");
hashmap1.put("Item 1", "Value 1");
hashmap1.put("Item 2", "Value 2");
hashmap1.put("Item 3", "Value 3");
hashmap1.put("Item 4", "Value 4");
hashmap1.put("Item 5", "Value 5");
hashmap1.put("Item 6", "Value 6");
Set set = hashmap1.entrySet();
Iterator iterator = set.iterator();
while(iterator.hasNext()) {
Map.Entry mapentry = (Map.Entry) iterator.next();
System.out.println(mapentry.getKey() + "/" +
mapentry.getValue());
}
}
}

```

```

Compile      : javac hashmap.java
Run          : java hashmap

```

5)

```
import java.util.*;
class hashtable
{
    public static void main(String args[])
    {
        Hashtable<String, String> hashtable1 = new Hashtable<String, String>();
        hashtable1.put("Item 0", "Value 0");
        hashtable1.put("Item 1", "Value 1");
        hashtable1.put("Item 2", "Value 2");
        hashtable1.put("Item 3", "Value 3");
        hashtable1.put("Item 4", "Value 4");
        hashtable1.put("Item 5", "Value 5");
        hashtable1.put("Item 6", "Value 6");
        Enumeration keys = hashtable1.keys();
        while(keys.hasMoreElements()) {
            String key = (String) keys.nextElement();
            System.out.println(key + "/" + hashtable1.get(key));
        }
    }
}
```

```
Compile      : javac hashtable.java
Run          : java Hashtable
```

6)

```
import java.util.*;
class iterator
{
    public static void main(String args[])
    {
        LinkedList<String> linkedlist = new LinkedList<String>();
        linkedlist.add("Item 0");
        linkedlist.add("Item 1");
        linkedlist.add("Item 2");
        linkedlist.add("Item 3");
        linkedlist.add("Item 4");
        linkedlist.add("Item 5");
        linkedlist.add("Item 6");
        Iterator<String> iterator = linkedlist.iterator();
        while(iterator.hasNext()) {
            System.out.println(iterator.next());
        }
    }
}
```

Compile : javac iterator.java
Run : java iterator

```
7) import java.util.*;
class linkedlist
{
public static void main(String args[])
{
LinkedList<String> linkedlist1 = new LinkedList<String>();
linkedlist1.add("Item 2");
linkedlist1.add("Item 3");
linkedlist1.add("Item 4");
linkedlist1.add("Item 5");
linkedlist1.addFirst("Item 0");
linkedlist1.addLast("Item 6");
linkedlist1.add(1, "Item 1");
System.out.println(linkedlist1);
linkedlist1.remove("Item 6");
System.out.println(linkedlist1);
linkedlist1.removeLast();
System.out.println(linkedlist1);

System.out.println("\nUpdating linked list items");
linkedlist1.set(0, "Red");
linkedlist1.set(1, "Blue");
linkedlist1.set(2, "Green");
linkedlist1.set(3, "Yellow");
linkedlist1.set(4, "Purple");
System.out.println(linkedlist1);

}
}
```

Compile : javac linkedlist.java
Run : java linkedlist

8)

```
import java.util.*;
class listiterator
{
public static void main(String args[])
{
LinkedList<String> linkedlist = new LinkedList<String>();
linkedlist.add("Item 0");
linkedlist.add("Item 1");
linkedlist.add("Item 2");
linkedlist.add("Item 3");
linkedlist.add("Item 4");
linkedlist.add("Item 5");
linkedlist.add("Item 6");
ListIterator<String> listiterator = linkedlist.listIterator();
while(listiterator.hasNext()) {
listiterator.set("This is " + listiterator.next());
}
while(listiterator.hasPrevious()) {
System.out.println(listiterator.previous());
}
}
}
```

Compile : javac listiterator.java

Run : java listiterator

9)

```
enum Mango {
Brooks, Manilla, Alphanso, Kesar, Maya
}
class MangoC {
public static void main (String args[])
{ Mango ap;
System.out.println("Here are all Mango constant:");
Mango allmangos[] = Mango.values();
for(Mango a : allmangos)
System.out.println(a);
System.out.println();
ap = Mango.valueOf(" Kesar");
System.out.println("ap contains"+ap);
}
}
```

Compile : javac MangoC.java
Run : java MangoC

```
10)
import java.util.*;
class properties
{
    public static void main(String args[])
    {
        Properties properties1 = new Properties();
        Set states;
        String outString;
        properties1.setProperty("Property 0", "Value 0");
        properties1.setProperty("Property 1", "Value 1");
        properties1.setProperty("Property 2", "Value 2");
        properties1.setProperty("Property 3", "Value 3");
        properties1.setProperty("Property 4", "Value 4");
        properties1.setProperty("Property 5", "Value 5");
        properties1.setProperty("Property 6", "Value 6");
        outString = properties1.getProperty("Property 3", "Missing");
        System.out.println(outString);
        outString = properties1.getProperty("Property 7", "Missing");
        System.out.println(outString);
    }
}
```

Compile : javac properties.java
Run : java properties

```
11)
import java.util.Map;
import java.util.Properties;
import java.util.Set;

public class setentry
{
    public static void main(String args[])
    {
        Properties props = System.getProperties();
        Set<Map.Entry<Object, Object>> entrySet = props.entrySet();
        for (Map.Entry entry: entrySet)
        {
            System.out.println(entry.getKey() + " : " +
```

```
entry.getValue());  
}  
}  
}
```

Compile : javac setentry.java
Run : java setentry

```
12)  
import java.util.*;  
class stack  
{  
public static void main(String args[])  
{  
Stack<Integer> stack1 = new Stack<Integer>();  
try {  
stack1.push(new Integer(0));  
stack1.push(new Integer(1));  
stack1.push(new Integer(2));  
stack1.push(new Integer(3));  
stack1.push(new Integer(4));  
stack1.push(new Integer(5));  
stack1.push(new Integer(6));  
System.out.println((Integer) stack1.pop());  
System.out.println((Integer) stack1.pop());  
System.out.println((Integer) stack1.pop());  
System.out.println((Integer) stack1.pop());  
System.out.println((Integer) stack1.pop());  
System.out.println((Integer) stack1.pop());  
System.out.println((Integer) stack1.pop());  
}  
catch (EmptyStackException e) {}  
}  
}
```

Compile : javac stack.java
Run : java stack

```
13)  
import java.util.*;  
public class StackTest{  
    public static void main(String[] args){  
        GenStack<String> gs = new GenStack<String>();
```



```

        System.out.println("Pushing Apple, Orange, Guava and Pineapple
into the stack...");
        gs.push("Apple");
        gs.push("Orange");
        gs.push("Guava");
        gs.push("Pineapple");
        System.out.println("...Done...\n");
        System.out.println("There are now " + gs.size() + " items in the
stack and the top item is " + gs.peek() + ".\n");
        System.out.println("When the " + gs.size() + " items in the stack are
Popped, they are displayed as: ");
        while (gs.hasItems())
            System.out.println(gs.pop());
        System.out.println("\nNow there are " + gs.size() + " item(s) in the
stack and the top item is " + gs.peek() + ".\n");
    }
}

```

Compile : javac StackTest.java
 Run : java SackTst

14)

```

import java.util.*;
class treeset
{
    public static void main(String args[])
    {
        TreeSet<String> treeset1 = new TreeSet<String>();
        treeset1.add("Item 0");
        treeset1.add("Item 1");
        treeset1.add("Item 2");
        treeset1.add("Item 3");
        treeset1.add("Item 4");
        treeset1.add("Item 6");
        treeset1.add("Item 5");
        System.out.println(treeset1);
    }
}

```

Compile : javac treeset.java
 Run : java treeset

15)

```

import java.util.*;

```

```
class vector
{
public static void main(String args[])
{
Vector vector = new Vector(5);
System.out.println("Capacity: " + vector.capacity());
vector.addElement(new Integer(0));
vector.addElement(new Integer(1));
vector.addElement(new Integer(2));
vector.addElement(new Integer(3));
vector.addElement(new Integer(4));
vector.addElement(new Integer(5));
vector.addElement(new Integer(6));
vector.addElement(new Double(3.14159));
vector.addElement(new Float(3.14159));
System.out.println("Capacity: " + vector.capacity());
System.out.println("Size: " + vector.size());
System.out.println("First item: " + (Integer)
vector.firstElement());
System.out.println("Last item: " + (Float) vector.lastElement());
if(vector.contains(new Integer(3)))
System.out.println("Found a 3.");
}
}
```

```
Compile      : javac vector.java
Run          : java vector
```