**9.8 Tree map :-**

The **TreeMap** class implements the **Map** interface by using a tree. A **TreeMap** provides an efficient means of storing key/value pairs in sorted order, and allows rapid retrieval. You should note that, unlike a hash map, a tree map guarantees that its elements will be sorted in ascending key order.   
The following **TreeMap** constructors are defined:

TreeMap( )   
TreeMap(Comparator *comp*)   
TreeMap(Map *m*)   
TreeMap(SortedMap *sm*)

The first form constructs an empty tree map that will be sorted by using the natural order of its keys. The second form constructs an empty tree-based map that will be sorted by using the **Comparator** *comp*. (Comparators are discussed later in this chapter.) The third form initializes a tree map with the entries from *m*, which will be sorted by using the natural order of the keys. The fourth form initializes a tree map with the entries from *sm*, which will be sorted in the same order as *sm*.

**TreeMap** implements **SortedMap** and extends **AbstractMap**. It does not define any additional methods of its own. The following program reworks the preceding example so that it uses **TreeMap**:

**import java.util.\*;   
class TreeMapDemo {   
public static void main(String args[]) {   
// Create a tree map   
TreeMap tm = new TreeMap();   
// Put elements to the map   
tm.put("John Doe", new Double(3434.34));   
tm.put("Tom Smith", new Double(123.22));   
tm.put("Jane Baker", new Double(1378.00));   
tm.put("Todd Hall", new Double(99.22));   
tm.put("Ralph Smith", new Double(-19.08));   
// Get a set of the entries   
Set set = tm.entrySet();   
// Get an iterator   
Iterator i = set.iterator();   
// Display elements   
while(i.hasNext()) {   
Map.Entry me = (Map.Entry)i.next();   
System.out.print(me.getKey() + ": ");   
System.out.println(me.getValue());   
}   
System.out.println();   
// Deposit 1000 into John Doe's account   
double balance = ((Double)tm.get("John Doe")).doubleValue();   
tm.put("John Doe", new Double(balance + 1000));   
System.out.println("John Doe's new balance: " +   
tm.get("John Doe"));   
}   
}**

The following is the output from this program:

Jane Baker: 1378.0   
John Doe: 3434.34   
Ralph Smith: -19.08   
Todd Hall: 99.22   
Tom Smith: 123.22   
John Doe's current balance: 4434.34

Notice that **TreeMap** sorts the keys. However, in this case, they are sorted by first name instead of last name. You can alter this behavior by specifying a comparator when the map is created. The next section describes how.