JAVA - THE LINKEDHASHMAP CLASS

This class extends HashMap and maintains a linked list of the entries in the map, in the order in which they were inserted.

This allows insertion-order iteration over the map. That is, when iterating a LinkedHashMap, the elements will be returned in the order in which they were inserted.

You can also create a LinkedHashMap that returns its elements in the order in which they were last accessed.

The LinkedHashMap class supports five constructors. The first form constructs a default LinkedHashMap:

```
LinkedHashMap( )
```

The second form initializes the LinkedHashMap with the elements from m:

```
LinkedHashMap(Map m)
```

The third form initializes the capacity:

```
LinkedHashMap(int capacity)
```

The fourth form initializes both capacity and fill ratio. The meaning of capacity and fill ratio are the same as for HashMap:

```
LinkedHashMap(int capacity, float fillRatio)
```

The last form allows you to specify whether the elements will be stored in the linked list by insertion order, or by order of last access. If Order is true, then access order is used. If Order is false, then insertion order is used.

```
LinkedHashMap(int capacity, float fillRatio, boolean Order)
```

Apart from the methods inherited from its parent classes, LinkedHashMap defines following methods:

SN	Methods with Description
1	**void clear()** Removes all mappings from this map.
2	**boolean containsKey(Object key)** Returns true if this map maps one or more keys to the specified value.
3	**Object get(Object key)** Returns the value to which this map maps the specified key.
4	**protected boolean removeEldestEntry(Map.Entry eldest)** Returns true if this map should remove its eldest entry.

Example:

The following program illustrates several of the methods supported by this collection:

```
import java.util.*;

public class LinkedHashMapDemo {

   public static void main(String args[]) {
      // Create a hash map
      LinkedHashMap lhm = new LinkedHashMap();
      // Put elements to the map
      lhm.put("Zara", new Double(3434.34));
      lhm.put("Mahnaz", new Double(123.22));
      lhm.put("Ayan", new Double(1378.00));
      lhm.put("Daisy", new Double(99.22));
      lhm.put("Qadir", new Double(-19.08));

      // Get a set of the entries
      Set set = lhm.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 Zara's account
      double balance = ((Double)lhm.get("Zara")).doubleValue();
      lhm.put("Zara", new Double(balance + 1000));
      System.out.println("Zara's new balance: " +
      lhm.get("Zara"));
   }
}
```

This would produce the following result:

```
Zara: 3434.34
Mahnaz: 123.22
Ayan: 1378.0
Daisy: 99.22
Qadir: -19.08

Zara's new balance: 4434.34
```