```java
/** This class represents an interator on a ConList as required by the interface
  * Iterable<E> of the List interface.
  *
  * @see  List
  * @see  Keyed
  *
  * @author  D. Hughes
  *
  * @version  1.0 (Mar. 2011)
  *
  * new concepts: implementing Iterator.                              */

/*Name: Long Nguyen: Student # 5427059
 * I modify this file from the conListIterator to make it works with linkedlists
 * I  added comments where I made changes
 * */

package MULTISET;

import java.util.*;

class LinkIterator < E extends Keyed > implements Iterator<E> {


    private int       cursor;  // the cursor that iterates through the list
    private MySet<E>   pointer;  // the cursor that iterates through the list
    private MySet<E>  list;    // the list being iterated over


    /** This constructor constructs an iterator on the specified ConList.
      *
      * @param  l  the list to be iterated over.                         */

```

```
LinkIterator ( MySet<E> l ) {


    list = l;
    pointer = l;
    cursor = 0;

}; // constructor


/** This method returns true if there are more items in the list.
  *
  * @return  boolean  more items on the list.                    */

/*This is the method to check that it has next with the LinkIterator
 * if list.top not equal then return true else return false
 * */
public boolean hasNext ( ) {  // from Iterator

    if(list.getTop() != null){
        return true;
    }else{
        list.setTop(pointer.getTop());
        return false;
    }


}; // hasNext


/** This method returns the next item in the list.
  *
  * @retuen  E  the next item on the list.                    */

```

```
    public E next ( ) {  // from Iterator

        E  i;

        if ( cursor >= list.getLength() ) {
            throw new NoSuchElementException();
        }
        else {
                //incrementing the pointer to the next value
            i = list.getTop().item;
             list.setTop(list.getTop().next);

            return i;
        }

    };  // next


    /** Removal is not supported so this method throws an
      * UnsupportedOperationException.
      *
      * @exception  UnsupportedOperationException  remove is not
supported.      */

    public void remove ( ) {  // from Iterator

        throw new UnsupportedOperationException();

    };  // remove


}  // LnkListIterator
```