public void sort(){

Node<E>[ ] tempArray = new Node[getCount()];

Or

E[] tempArray = (E[]) new Comparable[getCount()];

Or

Node<E>[] tempArray = (Node<E>[]) new Comparable[getCount()];

Node<E> mover = myHead;

Int i = 0;

while(mover != null) {

tempArray[i] = mover;

mover = mover.getLink();

i++;

}//while

UtilityMethods.bubbleSort(tempArray, getCount());

myHead = null;

for(int j = tempArray.length - 1; j > - 1; j--)[

add(tempArray[j].getData());

}//for

}//sort

public class Node < E extends Comparable<E>>///says that E has compareTo method

implements Comparable <Node<E>>{///says that the entire class has compareTo method

…

…

…

…

public int compareTo(Node<E> theOther){

return myData.compareTo(theOther.myData);

}//compareTo

}//Node

04/10/2014

Write 2 methods

1. public class method that has one parameter of class E and return type Boolean.
2. private method that is called by the public method to do the work. The private method calls itself until it finds the appropriate sublist where new Node is inserted at the head. The private method returns a reference to the new node if it has found the correct sublist or returns the head of the correct sublist