public boolean intsertSorted(Integer item)

{

Element ptr = head;// start with second element

Element prevPtr = null;

Element newItem;

boolean inserted = false;

if (head == null)

{

append (item);

return true;

}

if (item.compareTo(head.data)<0)// first element is special case

{

prepend(item);

return true;

}

if (item.compareTo(tail.data)>=0)

{

append(item);

return true;

}

newItem= new Element(item);

ptr= head.next;// start with second element

prevPtr= head;

while (ptr!= null && !inserted)

{

if (newItem.data.compareTo(prevPtr.data)>=0 && newItem.data.compareTo(ptr.data)<=0)

{

newItem.next = ptr;

prevPtr.next = newItem;

inserted = true;

}

prevPtr=ptr;

ptr= ptr.next;

}

return inserted;

}

public class Driver

{

public static void main(String [] args)

{

SLL myList = new SLL();

System.out.println("\nInitial list: ");

myList.showAll();

System.out.println("\nAdd One item: ");// Add item to Empty list

myList.intsertSorted(new Integer(3));

myList.showAll();

System.out.println("\nAdd One smaller than head:");// Add item to Empty list

myList.intsertSorted(new Integer(2));

myList.showAll();

System.out.println("\nAdd One larger than the last one:");// Add item to Empty list

myList.intsertSorted(new Integer(9));

myList.showAll();

System.out.println("\nAdd One in the middle:");// Add item to Empty list

myList.intsertSorted(new Integer(5));

myList.showAll();

}

}