# DoublyLinkedNode Class

All the linked list labs so far have used the ListNode class as the building block to make linked lists.

|  |  |
| --- | --- |
| Object |  |

# 

# We now introduce the DoublyLinkedNode class, which supports going forward and backward in the linked list. We say the linked list is a “doubly linked list.”

head

# The DoublyLinkedNode class

**public class** DoublyLinkedNode

{

**private** Object value;

**private** DoublyLinkedNode prev;

**private** DoublyLinkedNode next;

**public** DoublyLinkedNode()

{

value = null;

prev = null;

next = null;

}

**public** DoublyLinkedNode(Object v, DoublyLinkedNode p, DoublyLinkedNode n)

{

v

value = v;

prev = p;

next = n;

}

**public** Object getValue()

{

**return** value;

}

**public** DoublyLinkedNode getPrev()

{

**return** prev;

}

**public** DoublyLinkedNode getNext()

{

**return** next;

}

**public** **void** setValue(Object arg) ­

{

value = arg;

}

**public** **void** setPrev(DoublyLinkedNode p)

{

prev = p;

}

**public** **void** setNext(DoublyLinkedNode n)

{

next = n;

}

}

# Exercises

# Given a doubly linked list with one DoublyLinkedNode, write the code to insert a new node after the first node.

head

**head.setNext(new DoublyLinkedNode(null, head, null);**

# Given a doubly linked list with one DoublyLinkedNode, write the code to insert a new node before the first node.

head.setPrev(new DoublyLinkedNode(null, null, head);

head = head.getPrev;

# Given a doubly linked list with several DoublyLinkedNodes, write the code to insert a new node after the first node.

head

# DoublyLinkedNode insert = new DoublyLinkedNode(null, head, head.getNext());

# head.getNext().setPrev(insert);

# head.setNext(insert);

# Findthe node that contains the object obj and delete it. to delete means to link around it

# Int n = count;

# Int pointer = head;

# While(pointer.getNext().getValue() != object)

# {

# n++;

# pointer = pointer.getNext();

# }

# Pointer.setNext(pointer.getNext().getNext());

# ANSWER:

# While(! Temp.getValue().equals(obj))

# Pointer = pointer.getNext();

# pointer.getPrev().setNext(pointer.getNext()); // changes pointer of previous to the node after obj

# Pointer.getNext().setPrev(pointer.getPrev()); //changes previous pointer of the node after obj to the node before obj.