# DLNode 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 DLNode class, or “doubly linked node,” which supports going forward and backward in the linked list. We say the linked list made of many DLNodes is a “doubly linked list.”

head

# The DLNode class

**public class** DLNode

{

**private** Object value;

**private** DLNode prev;

**private** DLNode next;

**public** DLNode()

{

value = null;

prev = null;

next = null;

}

**public** DLNode(Object obj, DLNode p, DLNode n)

{

obj

value = obj;

prev = p;

next = n;

}

**public** Object getValue()

{

**return** value;

}

**public** DLNode getPrev()

{

**return** prev;

}

**public** DLNode getNext()

{

**return** next;

}

**public** **void** setValue(Object obj) ­

{

value = obj;

}

**public** **void** setPrev(DLNode p)

{

prev = p;

}

**public** **void** setNext(DLNode n)

{

next = n;

}

}

# Exercises

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

head

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

head

# Given a doubly linked list with head pointing to a linked list with several DLNodes, write the code to insert a new node temp after the first node.

head

obj

# temp

# In a doubly linked list referenced by head, find the first node that contains the Object obj and delete it. “To delete” means “to link around it.” Assume that the list contains the obj.