# Recursive Manipulation of Linked Lists

File **IntList.java** contains definitions for a linked list of integers (you worked with this file in the previous lab exercise). The class contains an inner class **IntNode**, which holds information for a single node in the list (a node has a value and a reference to the next node) and the following **IntList** methods:

> **public IntList()** -- constructor; creates an empty list of integers
>
> **public void addToFront(int val)** -- takes an integer and puts it on the front of the list
>
> **public void addToEnd(int val)** -- takes an integer and puts it on the end of the list
>
> **public void removeFirst()** -- removes the first value from the list
>
> **public void print()** -- prints the elements in the list from first to last

File **IntListTest.java** contains a driver that allows you to experiment with these methods. Save both of these files to your directory. If you have not already worked with these files in a previous exercise, compile and run **IntListTest**, and run it to see how it works. Then add the following method to the **IntList** class. Add a user option to **IntListTest.java** to test the method.

> **public void printRec()** -- prints the list from first to last using recursion. Hint: The basic idea is that you print the first item in the list then do a recursive call to print the rest of the list. This means you need to keep track of what hasn't been printed yet (the "rest" of the list). In particular, your recursive method needs to know where the first item is. Note that **printRec()** has no parameter so you need to use a helper method that does most of the work. It should have a parameter that lets it know where the part of the list to be printed starts.