Class CDLinkedList (it has Object data instead of int) is given, along with all other classes and interface. Your task is to implement class IntegerList, which is a list that is designed to store Integer Object (Please search google to find out about class Integer, which is not the same as int but can be used in a similar way).

The following methods are to be coded by you (You can test these methods by running Junit test file, IntegerListTest.java)

1. public void removeOddValue() throws Exception {// remove all odd numbers

* remove all odd numbers from the list
* if the list is empty, this method does nothing.
* If the list has {8,1,3,6,5,2} , the resulting list after removeOddValue will contain {8,6,2}.

1. public void removeRange(DListIterator itr1, DListIterator itr2, int rangeSize)

* Given iterator itr1 and itr2, you can assume that:
  + The list is not empty.
  + itr1 and itr2 are not null.
  + itr1 and itr2 each points to a node in the list.
  + itr1 never points at header.
  + itr2 always points to the same data as itr1 or to the right of itr1.
* This method removes data from position marked by itr1 to position marked by itr2 (inclusive) from the list, where rangeSize is the number of data from itr1 to itr2 (inclusive).
  + This method **must perform in ϴ(1)**.
  + For example, if list have data {1,2,3,4,5} , itr1 marks the position of 2, itr2 marks the position of 4, removeRange(itr1,irr2,3) will leave us with the list that contains {1,5}.

1. public void appendToBack(IntegerList l)

* This method appends l to the back of our list.
* This method **must perform in ϴ(1)**.
* List l will not be used again in the future.
* If our list contains {1,2} and l contains {3,4}, then appendToBack(l) will turn our list into the list that contains {1,2,3,4}.

1. public void evenToFront() throws Exception

* This method moves even value to the front of the list.
* All even values must maintain their order from the original list.
* All odd values must maintain their order from the original list.
* If we our list contains {2,9,6,8,1,3,4,7}, calling evenToFront() on it will turn our list into the list that contains {2,6,8,4,9,1,3,7}.

1. public int sum() throws Exception

* return the sum of all integer data stored in the list.
* Empty list will give 0 as its result.
* If our list contains {1,2,3}, calling sum() will return 6.

1. public void insertList(DListIterator itr, IntegerList l)

* put all contents of list, l, at position after the position marked by itr in our list.
* List l will not be used again in the future.
* itr is assumed to mark a position in your list. It can mark header.
* If our list contains {1,2,3} and l contains {4,5,6}, and itr marks the position of value 2, then insertList(itr,l) will turn our list into the list that contains {1,2,4,5,6,3}.
* This method **must perform in ϴ(1)**.

Each test method is 0.5 point. There are 26 test methods in JUnit. The total score is 13.

**How to submit: attach your IntegerList.java to MyCourseville submission page. Make sure you really attach the file.**