# LAB EXERCISE

## MergeList

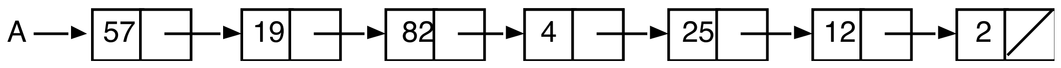
**Background:**

An unordered linked list is difficult to sort given its sequential nature, but a recursive merge sort can be developed for linked lists. The algorithms required (split and merge) will provide excellent practice in working with the `java.util.LinkedList` and `ListIterator` classes.

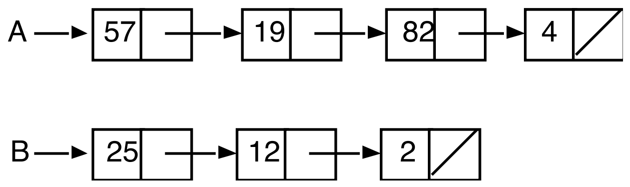
A linked list can be sorted using a recursive merge sort algorithm. Here are the steps:

1. Split the list into two smaller lists.
2. Recursively sort these two lists using merge sort.
3. Merge the two sorted lists together.

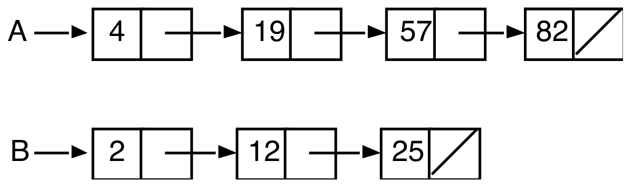
Suppose we begin with a list of seven nodes, pointed to by A.



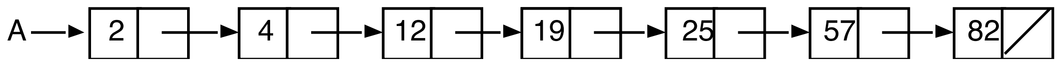
We then split this list into two smaller lists that differ in size by no more than one node. It does not matter which values end up in either list. In the example to follow, list A will take the first half of the list while list B will have the second half of the original list.



Next, we recursively sort these two lists.



Finally, we merge these two sorted lists together.



You are encouraged to look over your code for the `merge` and `mergeSort` methods from Lesson 24. Reviewing these algorithms as applied to arrays will help you to solve the same mergesort concept with linked lists.

**<u>Assignment:</u>**

1. The linked list should be of type `java.util.LinkedList`.

2. The data file to be used in this lab is (*file20.txt*).

3. Building the initial linked list from (*file20.txt*) should follow this logic. As each new piece of data (Id/Inv pair) comes off the data file, it is placed at the beginning of the list. Therefore, the first values read from the data file will end up last in the list.

4. The recursive merge sort algorithm will need the supporting algorithms of splitting and merging lists. List iterators should be used to implement these methods.

5. Your program should consist of this sequence of scripted events:

   Load the data file and build the initial list
   Print the linked list - it is unordered
   Recursively merge sort the list
   Print the linked list - it is now sorted
   Reverse the linked list
   Print the linked list - it is now in descending order

6. If your instructor chooses, you will be provided with a program shell consisting of a `MergeList` class containing a main method, and the `Item` class. **All** of the code development should appear in the `MergeList` class. With this template, you are encouraged to NOT use generics!! Here are some of the specifications of the `MergeList` class.

   a. The `reverseList` method is stubbed out as a print statement.

   b. The `split` method is stubbed out as a print statement.

   c. The `merge` method is stubbed out as a print statement.

   d. Methods to read the data file and print the list are provided.

**<u>Instructions:</u>**

1. Modify and write code as necessary to satisfy the above specifications.

2. Display the source for the `MergeList` class.

3. Display your source along with the run output for file20.txt then call your instructor to your workspace for scoring.