CS61B, Fall 2009 **HW #6** **P. N. Hilfinger**

Due: Mon., 19 October 2009

Create a directory to hold your answers. There is a skeleton for your solutions in the repository under `staff/hw6`, and also in the directory `~cs61b/code/hw6`. Put non-program answers in a file hw6.txt. Use the usual command sequence to copy your final solution to a `hw6-`N entry in your tags repository directory.

1. Consider the program `Sorter3.java` in the files for this homework set. It's the same as `Sorter2.java` from Lab #7, except that it uses a LinkedList rather than an ArrayList. What effect do you expect this to have on the running time of `Sorter3` and why? (Try to answer this question without empirical measurement, giving a $\Theta(\cdot)$ estimate.)

2. You can speed the `Sorter3.java` program up considerably by using `ListIterators` and its operations, rather than the `.get` and `.set` methods on `LinkedLists`. Indeed, you should be able to make its speed comparable to that of `Sorter2`. Modify `Sorter3.java` to accomplish this.

3. [Goodrich & Tamassia] Suppose that A is an $n \times n$ array of 1's and 0's with the property that all the 1's in a row come before all the 0's in that row. The array is huge ($n > 500000$), but instead of actually being stored as a Java array, it is represented by a `BitMatrix` object with a method `.get(`i, j`)`, which returns A_{ij} (i is the row, j the column). Fill in the method `mostOnes(A)` in the template file `BigMat.java` so that it returns the index of the row of A that contains the most 1's. When several rows contain the largest number of 1's, return the smaller index. Your method must operate in $O(n)$ time (*not* $O(n^2)$ time). Your program will be given a time limit that requires it to operate in better than $O(n^2)$ time.