Programming Assignment 2:
Implementing Sorting Algorithms

Due date: 11:55 pm on Monday, March 5, 2018

Description:

You are to write three C++ programs.

The first program will be an implementation of insertion sort using linked lists to store the data. Make sure your implementation is correct and efficient (for insertion sort). The linked list must be implemented by you, and all movement in the lists must be done using pointers. You must read the integers from the files and build a linked list in the order that matches the file and then sort that linked list. You may not change what data is in any node after the data is initially stored into a node.

The second program is an implementation of shellsort using the best sequence described in your text.

The third program will implement **bottom up** mergesort.

Use arrays or vectors for your shellsort and mergesort programs. You are not required to use classes or objects for any of the programs, though, of course, you may. If you build a linked list class, put your sort inside that class. You are expected to use reasonable program design – programs that have no functions except main will do no better than a C. Note that your sorting functions are NOT to be templated (I do not want you just copying code out of the book).

For each program, you are to read a file of integers (longs), sort them, and write them into an output file. The input files will contain one integer per line, and the output files must be in the same format. Your program will accept the names of the input and output files as **command line** parameters. Your program must be able to handle files of up to at least 1,000,000 longs. Sample input files will be given later in ReggieNet.

Hints for figuring out whether or not your programs work – check out the sort and diff commands for UNIX.

Submit the following items to ReggieNet:

1) **Group submission**: your source files from the School's linux machines. Use a separate source file for each sorting. Name your files with clearly meaningful names (no main.cpp, please). Make a zip or a tar file that include all of your files. Include only the .h and .cpp files and a makefile if you have one, no folders, no backup files, no executables, nothing extra.
2) **Individual submission**: Peer evaluations by each member.