CS 2401 Spring 2023

Lab 2: Sorting and GDB

Due: Friday 2/3 at 11:59 PM

For this lab I have provided a little class that stores numbers and can sort them. The sorting algorithm this class uses is insertion sort.

The problem is that the insertion sort is going into an infinite loop, and I need you to diagnose why that is happening and to fix it. I want you to do this using *gdb.* I will also need you to write the load_from_file and write_to_file member functions, as well as a main that drives the program.

Start by downloading the files I have provided on Blackboard.

You will find a little class that stores numbers in an array. Some of these functions have been written for you, and some are left for you to write yourself.

Your jobs are to:

- Write the implementations for the load_from_file and save_to_file functions.
- Write a main that asks the user for the name of their file. Remember you can read the filename into a string and then use the .c_str() function to use that string in an open command:
    - ifs.open(filename.c_str());

The main then:

- loads the array from the file into a NumList object
- Calls the i_sort function on the NumList
- Writes the sorted numbers out to the screen, using the see_all function.

Here there is a problem. Something goes into an infinite loop. To find this you are going to use gdb (or lldb if you are on MacOS). I have provided a data file called smaller.dat, so that you will not have to walk through so much code. Here are the steps:

- Compile with the –g option (g++ -g *.cc)
- Type gdb ./a.out
- Type break main
- Type run
- Start hitting n to walk through the code
- When you get to the call of a member function you want to “step into” hit s
- Continue hitting n

Using this method, try to find the error in the code that I gave you, and then correct it. Take screenshots of running gdb showing at least you starting the program and the point at which you found the issue. When you have corrected the code, recompile everything and test it. You should see a sorted list of numbers on the screen for either the larger.dat or the smaller.dat file.

Open your main again, and have it ask the user to type in three of their own numbers, which will be inserted into the list. Using the string member function find (which returns the location of a substr within the string) and the insert function (which will insert a substring at the position sent as its first argument), modify the filename, so that the characters “sorted” are inserted before the dot. (This means that “larger.dat” becomes “largersorted.dat” and “ebenezer.txt” becomes “ebenezersorted.txt”). Then have the main write the number array out to that file.

After you have successfully compiled all of this run it on the larger.dat file and insert the numbers 107, 541 and 23,508.

Submit your source code, sorted data file, and gdb screenshots on Blackboard.