CS 2401 Fall 2021

Lab 7: Doubly Linked Lists

Due: Friday 10/15 at 11:59 PM

Begin this lab by copying the files, list.h, list.cc and lab9main.cc into your working directory from Blackboard.

This is a class that holds a doubly linked list of integers, and a short main that drives it. The list has an unusual feature wherein it treats numbers that are divisible by five in a different manner and puts them in the middle of the list, while putting all the other numbers at the end of the list. You should begin the lab by opening the files and reading through them until you have an idea of what is happening here. Then you should begin:

1. In the llist.cc file write the implementation of the `rearview` function which will print the list backwards. This should be pretty easy since you have a function that prints the list frontwards right above that. Compile this and the lab9main.cc file and run it.
2. Clearly there's a problem – something is not working right. Now it's time to debug:
   gdb – This time you're going to be using the commands to find your way on your own. Compile with the *–g* option
   break function_name or *break filename:line-number* will stop the program at that spot
   run – executes the program up until the break point, or until it crashes or until it ends
   n – executes the line in front of you when you're single-stepping through a section
   s – if the line in front of you is a function call this will "step into" the function
   p variable-name – lets you see the contents of a variable, if you "peek" at a pointer you will see an address, if you "peek" at *pointer you will see the contents of the thing pointed at
   c – continue the program to the next breakpoint, often used to cycle to the next iteration of a loop
   where – if a program has crashed or been stopped with a Ctrl-c this will let you see the line where it was stopped, as well as all the other function calls that got you to this spot, a good first step in diagnosing a seg fault or infinite loop
   q – quit
3. Use these tools and some scratch paper (remember that it helps to draw the boxes) to figure out what's wrong with the program. Then fix it, while keeping the unusual way that it's putting items divisible by five in the middle of the list.
4. Test the code again.
5. Now write the implementation of a copy constructor, with a note that using the add function will not preserve the order of the items because of its unusual way of treating %5 == 0 items. Uncomment the section of the main that tests this.
6. Run your program and make a script file of the running (or copy the terminal output if you don't have the script command available).
7. Submit on Blackboard your finished code along with the output file.