Lab 6 README

ECE 13

Sean Manger

Prof: Steve McGuire

F24

Author: <Sean Manger> (smanger@ucsc.edu)

Collaborators: For this lab, I again collaborated with Donny Tang. We just discussed general concepts about the lab. For instance, Donny asked me what results I was seeing for the sorting experiment on the Nucleo since he was getting different results on the terminal vs on the Nucleo's serial monitor. We compared our results for the timing experiment and found that they were quite similar. Besides this everything else we discussed was purely conceptual such as did you get a Segfault in LinkedListRemove()? As always, all my code is my own original work and was written entirely by me.

Results of Timing Experiment:

```
Welcome to CRUZID's Lab06_main.c,compiled on Nov  5 2024 22:54:52.

Starting stopwatch for SelectionSort()...
alleged beef crowd damage decide eggnog exuberant faithful finger frequent hair head hug juggle jump lowly maddening nest pies pig punishment ragged rake raspy reduce reign robin
rock scream selective shut side sidewalk sort step substance surround telephone tendency thought toothpaste zip
Program took 10 cycles and 0.000000 ms to complete.

Starting stopwatch for InsertionSort()...
alleged beef crowd damage decide eggnog exuberant faithful finger frequent hair head hug juggle jump lowly maddening nest pies pig punishment ragged rake raspy reduce reign robin
rock scream selective shut side sidewalk sort step substance surround telephone tendency thought toothpaste zip
Program took 6 cycles and 0.000000 ms to complete.
root@Seans:/mnt/c/Users/seanm/smanger/Lab06/src#
```

```
Welcome to CRUZID's Lab06_main.c,compiled on Nov  5 2024 20:57:18.

Starting stopwatch for SelectionSort()...
alleged beef crowd damage decide eggnog exuberant faithful finger frequent hair head hug juggle jump lowly maddening nest pies pig punishment ragged rake raspy reduce reign robin roc
k scream selective shut side sidewalk sort step substance surround telephone tendency thought toothpaste zip
Program took 4294966963 cycles and 268.000000 ms to complete.

Starting stopwatch for InsertionSort()...
alleged beef crowd damage decide eggnog exuberant faithful finger frequent hair head hug juggle jump lowly maddening nest pies pig punishment ragged rake raspy reduce reign robin roc
k scream selective shut side sidewalk sort step substance surround telephone tendency thought toothpaste zip
Program took 4294966429 cycles and 268.000000 ms to complete.
```

I observed the above results for the timing experiment in this lab. For SelectionSort() I got about 10 Cycles, 0 ms on the terminal and for InsertionSort() I got about 6 cycles 0 ms. On the Nucleo my results were both functions taking over 4 billion cycles and 268 ms to complete. These results match both Donny's results and what some of the class has been seeing according to the class Discord server. So, in terms of consistency, I think I am spot on. In terms of performance on my laptop both SelectionSort() and InsertionSort() ran pretty much the same with InsertionSort() having fewer cycles. On the Nucleo board InsertionSort() again had the slight advantage with fewer cycles required for completion. On both the Nucleo & the standard

terminal the time to complete both processes was the same albeit 268 ms slower on the Nucleo when compared to the terminal. The fact that it takes longer on the Nucleo than the Windows terminal should come as no surprise. I am running an Alienware laptop and most folks I know have similar gaming laptops. The computing power of a gaming laptop is of course many orders of magnitude greater than our Nucleo board. As such, it is no surprise that the results showed the test was faster on a laptop terminal.

Lab Summary & Process

My process when starting this lab was to initially review the material on pointers and Linked Lists provided in both lecture and the assigned class reading. I had struggled with pointers in the past in a community college C++ course and didn't want to face the same difficulties again. I then proceeded to read the lab manual thoroughly and ensured I understood what was expected in Lab 6. After this, I then began working on implementing the required Linked List functions for the lab. I thought that the most important thing that helped me personally in this lab was to have a solid conceptual understanding of how both a Linked List & pointers work going in. This made planning and brainstorming the necessary functions much easier. I also had my IPad on hand to draw pictures of how a given Linked List would work in each function. These diagrams were extremely helpful for me as I worked towards completing the lab.

I would say that the most difficult part of this lab for me was just thoroughly testing each function to ensure that it worked properly. Since the sorting algorithms require the use of our LinkedList functions, any problems with them would greatly compound any problems with the implementation of the sorting algorithms. In the end, however, each of my functions passed all their tests and I completed the lab successfully. If I could do this lab over again, I probably wouldn't change much but I'd might consider drafting the readme as I go along as opposed to at the very end. After much coding, I always feel like the readme is somewhat rushed.

I thought that this was an excellent lab that I probably spent about 10 – 12 hours total working on. As previously mentioned, I had struggled with both pointers and Linked Lists in the past and thought that any assignment related to them was a nightmare. This time around however, I found myself moving through the lab without much difficulty at all. I think the likely reason is that Steve presented the material in a much simpler and more digestible way than my previous professor at community college. I also would imagine that I have grown as both a programmer and a student since then, resulting in a much easier time. If I could offer any improvements to this lab, I would say perhaps introduce pointers earlier in the course. Pointers are tremendously useful, but I've noticed that many students including myself tend to struggle with them. I think that a more gradual build into pointers might alleviate much of the angst surrounding pointers. Other than that, I again thought that this was an excellent lab that further grew my skills both in C and as a programmer.