ECE 361 Fall 2023
Homework #3a

SINCE WE ARE COMING INTO MIDTERM EXAM WEEKS FOR FALL TERM, I AM RELEASING HOMEWORK #3 IN TWO PARTS WITH TWO DUE DATES. PART A (THIS PART) SHOULD BE SUBMITTED TO CANVAS BY 11:59 PM ON SUN, 29-OCT. NO LATE SUBMISSIONS FOR PART A WILL BE ACCEPTED. PART B (IN ece361f23_hw3b) WILL BE DUE TO CANVAS BY 11:59 PM ON SUN, 05-NOV. NO LATE SUBMISSIONS FOR PART B WILL BE ACCEPTED AFTER 8:00 AM ON WED, 08-NOV. I KNOW THAT MIDTERM SEASON AT PSU CAN BE STRESSFUL, HOPEFULLY THIS HELPS.

THIS ASSIGNMENT (PART A) SHOULD BE SUBMITTED TO CANVAS BY 11:59 PM ON SUN, 29-OCT-2023. NO LATE SUBMISSIONS FOR THIS ASSIGNMENT WILL BE ACCEPTED BECAUSE I WILL BE REVIEWING MY SOLUTIONS IN CLASS ON MONDAY.

THE ASSIGNMENT IS WORTH 35 POINTS. IT IS EASIEST TO GRADE, AND I BELIEVE IT IS EASIEST FOR YOU TO SUBMIT, SOURCE CODE FILES AS TEXT FILES INSTEAD OF CUT/PASTE YOUR CODE INTO A `.doc` OR `.docx` FILE. SHORT ANSWER, TRUE/FALSE AND MULTIPLE-CHOICE QUESTIONS SHOULD BE SUBMITTED IN A SINGLE TEXT OR .PDF FILE FOR THE ASSIGNMENT. CLEARLY NAME THE FILES SO THAT WE KNOW WHICH QUESTION THE ANSWER REFERS TO. SUBMIT THE PACKAGE AS A SINGLE .ZIP, .7Z,.TGZ, OR .RAR FILE (EX: `ece361f23_rkravitz_hw3a.zip`) TO YOUR CANVAS HOMEWORK #3 DROPBOX

FOR THIS ASSIGNMENT, YOU WILL BE TURNING IN THESE FILES:

- .C FILE WITH SOURCE CODE FOR PROBLEM 1
- TRANSCRIPT OF YOUR COMMAND LINE SESSION SHOWING THAT YOUR SOLUTION COMPILES AND WORKS CORRECTLY. INCLUDE AT LEAST 10 WORDS IN YOUR SORTED LIST

NOTE: THERE WAS CODE IN HOMEWORK #1, PROB3_STARTER.C THAT SHOULD DISPLAY YOUR WORKING DIRECTORY. THESE APPLICATIONS SHOULD INCLUDE THE CODE TO DISPLAY A GREETING AND YOUR WORKING DIRECTORY.

ACKNOWLEDGEMENT: THIS PROBLEM IS BASED ON EXERCISES AND PROGRAMMING PROJECTS PROVIDED BY K. N. KING IN *C PROGRAMMING: A MODERN APPROACH.*

Problem 1 (35 pts) Dynamic memory allocation (`malloc()` and `free()`)

The C standard library includes two functions that make use of a function pointer to provide flexible sort (`qsort()`) and binary search (`bsearch()`) capabilities. We are going to use those functions, along with `malloc()` and `free()`, which provide support for allocating blocks of dynamic memory. Consider a program that sorts a series of words entered by a user:

```
Enter word: foo
Enter word: bar
Enter word: baz
Enter word: quux
Enter word:
```

In sorted order: `bar baz foo quux`

a. (30 pts) Write a program that reads in words from `stdin` and prints them to `stdout` in sorted order. Your program must have the following constraints and assumptions:
   - Assume that each word is no more than 20 characters long.
   - Stop reading words when the user enters and empty word (only a \n).
   - Use `malloc()` to store each word in a dynamically allocated string.
   - Hold the string addresses returned by `malloc()` in an array of pointers to keep track of the strings.
   - After all the words have been entered, sort the array using the C library `qsort()` function. The compare function is included in the starters folder.
   - Use a loop to print the words in sorted order.
   - Don't forget to use `free()` to deallocate the strings before you exit the program.
   - *Hint: This is not required, but you may want to use the `getaline()` (and perhaps the `copy()`) function(s) from the `LongestLineHelper` ADT used in Homeworks #1 and #2.* `LongestLineHelper.o` is included in the starters folder.

     Note: The intent is that you only use the provided API which is the same as in LongestLineHelper.c but you won't be marked down if you make changes to the functions.

b. (5 pts) Compile, debug, and execute your application using the command line. Include at least 10 words to show that your application works correctly. Submit your source code and a transcript showing that your application runs successfully.