Thomas Gibbons

October 7, 2016

ECE 3220

LAB 4

Objective

The objective of this lab was to learn to use functions in our programs to break up different parts of our code to perform different tasks. We learned to work with arrays of different types: int, char, double, etc. We also learned to use arrays dynamically and statically allocated. We learned to handle file and file pointers.

Discussion

In my code I first get user input from 1 to 99 since in the format of files the number was up to 2 digits. Given the input, I was able to create a char string with the name of the input file. I used the function sprintf to combine the integer input with the rest of the premade input name.

Next I used the function readFile I created to take in 2 int pointers to store the count and max found in the input files and the name of the input file and return an array of integers found in the input file to a dynamically allocated array. I utilized the fscanf first to scan in the first 2 numbers and store the count and max. Then I used it repetitively to store every number in the file by scanning the amount of times specified in the count. If readFile returns NULL the file couldn't be accessed and the program will terminate.

After I took user input to determine whether to scale or offset the data. And then took in a scaling or offsetting value accordingly. I then used a function to scale/offset the data with the given value and stored it in a dynamically allocated double array.

Then I used the function outputFile I created to take in a value (count or some number), another value (offset/scale or some number), double* (array or NULL) and a char* (output file name). I wrote this function strategically to be able to use to also write the stats data output file. The function prints the first 2 values to the first line of the file and prints the data in the rest if it is not NULL. So for the exception of the stat file I can just pass the mean instead of count and max instead of an offset/scale value and put NULL in data to skip that part of the printing.

For part two, I utilized the functions I already made plus a few more. For the statistics of the data I found the mean by adding every value of the array and dividing by the count and the max value by comparing every number in the array and when it is greater than the previous high it will replace the high.

To center the data around zero, I utilized the offset function by making the value of the offset the average of the data times negative one. This makes the average of the off-setted data zero.

To normalize the data between 0 and 1, I utilized the scaling function by making the value of the scale one divided by the max value. This makes it so if the max is in the data when multiplied by the scale it would be 1 which would still be the max.

Finally I freed all the allocated memory including the input file names, output file names, and data arrays. The link to my code on my GitHub account is http://github.com/tgibbons95.