Nicole Bowler

CS 202 - 1001

Due: Wednesday, January 31st, 2018

C. Papachristos

Project 1

The goal of this programming assignment was to review the basics of C programming to get ready for the semester of C++. Along with that the specifics included in the goals were to use basic file I/O, console I/O, getting comfortable with strings and arrays, writing and implementing functions, and to create a program in the C++ language that compiles and runs on Linux. The program written is to ask the user for names for both and input and output file to be used in the process. Then it would take the names written on the input file (assumed to be 10 names of 8 characters in length) and copy them to a 2D char array. That array would then be printed to the console and output file before the names get sorted alphabetically. The sorted names would then also be printed to both console and output file.

My implementation of this programming assignment includes three functions aside from the main. Those additional functions include: myStringCopy(which is a function that copies the contents of a source string into another string and does not return a value), myStringLength(a function that counts the number of characters in a string and returns that value as an int), and myStringCompare(a function that moves down two strings' characters until they do not match and then returns a 0, -2 or +2 based on which strings character is greater if any). The myStringCompare() would move down the list to be sorted and look at two strings at a time and swap them using myStringCopy() so that the lesser string would be progressively moved left

until all the strings where in alphabetical order. This was established by having the compare function move down the entire array of indexes the same amount of times as there were arrays.

My bugs and issues with the code came from small mistakes such as using similar names like ‘temp’ and ‘tempInt’ which were different data types or other mistakes such as making an array go into unallocated memory addresses by accessing an index outside of the declared array. The majority of the bugs and errors were found in data types and *for* loops accessing arrays. One bug that I could find a solution for in the limited time frame is that one of the strings in the sorted array would append a string adjacent to it even though it surpassed 8 characters. This could be an issue in printing of the arrays or it may be within the array itself. The closest clue that I found while attempting to debug it was that it took place when copying the first array into a temporary array for sorting. If given more time I would spend as much time as needed until each index contains only one name as it should.