ECEN 5033 LAB 0 Write-Up

Name: Nitik Gupta

Code Organization:

The code is organized based on their common functionality and their usage. There are 4 different c++ files in the project: main.cpp, util.cpp, mergesort.cpp and quicksort.cpp. As the name suggests, util file contains all the miscellaneous functions used in the utility of the main function. Similarly, mergesort and quicksort contains the functions used for the sorting algorithm. And the main file contains the main function. The header files for each file contains the function declarations of the functions used in C++ file. In the next topic, the functionality of each function is given in detail.

File Description:

- main.cpp: Contains the main function where the command line arguments are captured using the getopt_long function. These arguments are then processed for their respective functionality.
- makefile: makefile for the project which contains the commands to compile the code ad create the object files of each program and then combine in the mysort object file for the execution of the whole program.
- mergesort.cpp: Contains the function definitions required to perform mergesort operation on the given numbers.
- mergesort.h: Contains the function declarations required to perform mergesort operation on the given numbers.
- quicksort.cpp: Contains the function definitions required to perform quicksort operation on the given numbers.
- quicksort.h: Contains the function declarations required to perform quicksort operation on the given numbers.
- util.cpp: Contains the function definitions of all the miscellaneous utility functions that performed inside the main function like print, write to file etc.
- util.h: Contains the function declarations of all the miscellaneous utility functions that performed inside the main function like print, write to file etc.

Compilation Instructions:

The makefile can used to help in compilation. Just call make or make all to compile the object file and use them to execute the program. To delete all the object files, you can use make clean in the command line.

Execution Instructions:

- After the compilation of the program, you can use the ./mysort instruction on the command line, to execute the program. There are 4 arguments you can give to the ./mysort instruction (you must at least give 2, the output and input file for successful execution of the program).
- These are --name, input file (without any argument variable), output file (with variable -o) and algorithm required to sort.

- The --name argument will print out the name.
- The input file doesn't require a seperate variable, but without an input file, the code will not run and will exit the program.
- The output file uses the variable –o. If Output file is provided then it will write to it, if not it will print on the console.
- Last is the algorithm with variable --alg, it can take two values, one is merge and other is quick, according to the value they will perform quicksort and mergesort. If something else is given, or nothing is given, it will perform mergesort by default. There are some bugs which will be discussed in the next section.

Extant Bugs:

1. If you provide more command line variables than the required, then the input file might not be taken by the program, and there won't be a sort program execution.
2. If values in the input file is greater than INT_MAX(+2147483647), then the program will not be able to apprehend them and will give wrong answers.

References Used for the Project:

https://www.geeksforgeeks.org/command-line-arguments-in-c-cpp/

https://stackoverflow.com/questions/30141000/c-how-to-pass-command-line-argument-to-read-txt-file/30141127

http://www.cplusplus.com/forum/beginner/185537/

https://www.geeksforgeeks.org/quick-sort/

https://www.geeksforgeeks.org/merge-sort/

https://www.geeksforgeeks.org/converting-strings-numbers-cc/

https://www.geeksforgeeks.org/getopt-function-in-c-to-parse-command-line-arguments/

https://www.geeksforgeeks.org/file-handling-c-classes/

Version Control Link:

https://github.com/IMNG7/CocurrentLab1