CSC 415 Operating Systems Principles
Spring 2019, Anthony J Souza

Make sure to click the link on ilearn to create your repository

Make sure to clone your repo before beginning your work.

Homework Assignment #2. Basic Programming Skills Review

Section 2.3 (Figure 2.4) of our textbook describes an algorithm for implementing a COPY program. This program prompts the user for the input and output file names, then copies the contents of the input file to the output file. You will be implementing this pseudocode. The standard tee command copies data from standard input to standard output (like an ordinary shell pipe), however it also writes a copy of standard input to a file named on the command line. See the man page for details. Your program's user interface should appear ***exactly*** as follows:

> Welcome to the File Copy Program by <yourname>!
> Enter the name of the file to copy from:
> <type in file name & hit return>
> Enter the name of the file to copy to:
> <type in the file name & hit return>
> File Copy Successful, <number> bytes copied

You will implement this program in the C language (gcc compiler) using the POSIX (i.e. Linux) API/gcc compiler. Your code should use a buffer of size 21 bytes. Following the pseudocode logic, the file read will repeatedly read from the input file into the buffer, and the file write will repeatedly write from the buffer to the output file.

Use the low-level file I/O API for reading/writing the files (this API is also referred to as the OPEN family of Linux system calls); use the FORMATTED I/O API for prompting the user and reading user input (this API is also referred to as the FOPEN family of system calls). DO NOT USE THE FOPEN family calls for reading and writing to and from files. Most of these functions begin with the letter f. You may use printf for prompts. Be sure to include all necessary error checking. Use the man page entries for function prototypes, details on how to use each system call as well as all of the necessary #include files.
Note that most system calls are in section 2 or 3 of the man pages (i.e. use 'man 2 write' not 'man write' to get information about the write system call).

Make sure the source file (the one copying from) exists, if it does not you may error out. When opening the destination file (the file copying to), the file does not need to already exist. If it does, simply overwrite it. File Extensions do not need to be maintain. Even if it breaks the file. Linux's cp command doesn't enforce this either.

1. Implement your program and test it for correct implementation using three (small) test files. Use the Linux ***diff*** utility to compare your input and output files. Also instrument your program with debugging code to display the buffer contents during each file read/write iteration. Include debugging output for a small test file; and also test your program using 3 larger files (without debugging output). Once you have correctly designed and tested your program, run the program using the ***strace*** utility that traces (lists) all of the system calls used during execution. Explain (annotate) this output. Do your best to explain this output. This explanation should be writing in your readme.txt file. ***Set the strace flags to display the number of times each syscall is used in your program.***

Homework Submission

1. Fill in the README.md to contain the following:
    - The command to build your project.
    - The command to run your project.
    - What your code does.
    - Annotated output of strace of your program.
2. Push all files to your repository with the following commands
    - git add .
    - git commit -m “message”
    - git push