Introduction to Operating Systems
Fall 2024

Assignment 2
(10 points)

Develop a C program, "mycopy", whose main routine accepts two input parameters from the user, an input file name and an output file name (both passed to your program when it was invoked from the shell). Both files are text files.

Your program shall then create an output file and print the user's full name, followed by the user's ID (an integer), followed by a newline and then copy the contents from the input file into the output file. It is okay to hardcode your name as a string literal in your code, but for the user ID, you shall use a system call that gets it for you.

Use Unix calls for opening, closing, reading and writing files, not the standard C calls.

Below is an example of how your program should be invoked from the shell:

```
mycopy input.txt output.txt
```

Where input.txt is a file that already exists (you may create an input file with a few lines to test your code).

If a path is not provided in filenames, then it's assumed that a file is located at the same directory as the working directory of your program, i.e. the directory where your program was invoked from.

After developing your program, invoke using `strace` and then answer the following questions:

1) What are the system call names for getting the process' userID, opening a file, closing a file, reading a file and writing a file?
2) How many system calls are (i.e. the count) involved with opening a file, closing a file, reading a file and writing a file? (count each individually. You may either use strace options to aid you in doing so, or you may use grep).
3) What was the value of the file descriptor of your read file? Should we expect it to change if you change the order of opening the input and output files?
4) What was the value of the file descriptor of your write file? Should we expect it to change if you change the order of opening the input and output files?

Notes and hints:
- Please include your answers, the strace log in your submitted .pdf file.
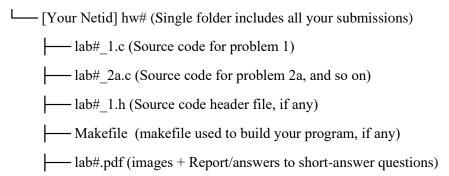- Create a text file and use it to test your program, e.g. type:
  ```
  touch input.txt
  echo "Hello world" > input.txt
  echo "This is lab 2" >> input.txt
  ```
- Use the man pages to learn how to use POSIX API library functions (and the necessary include files) and/or UNIX commands and its various optional arguments (e.g. strace, especially for counting), e.g.:

```
man strace                    // gets info from section 1, user's manual
man getuid.2                  // section 2 is programmer's manual
```

Submission file structure:

Please submit a **single .zip file** named **[Your Netid]_lab#.zip**. It shall have the following structure (replace # with the actual assignment number):

└── [Your Netid] hw# (Single folder includes all your submissions)

    ├── lab#_1.c (Source code for problem 1)

    ├── lab#_2a.c (Source code for problem 2a, and so on)

    ├── lab#_1.h (Source code header file, if any)

    ├── Makefile (makefile used to build your program, if any)

    ├── lab#.pdf (images + Report/answers to short-answer questions)

What to hand in (using Brightspace):

- Source files (.c or .h) with appropriate comments.
- Your Makefile if any.
- A .pdf file named **"lab#.pdf"** (# is replaced by the assignment number), containing:
  - Screen shot(**s**) of your terminal window showing the current directory, the command used to compile your program, the command used to run your program and the output of your program.

RULES:

- You shall **use kernel version 4.x.x or above**. You shall not use kernel version 3.x.x.
- You may consult with other students about GENERAL concepts or methods but copying code (or code fragments) or algorithms is NOT ALLOWED and is considered cheating (whether copied form other students, the internet or any other source).
- If you are having trouble, please ask your teaching assistant for help.
- You must submit your assignment prior to the deadline.