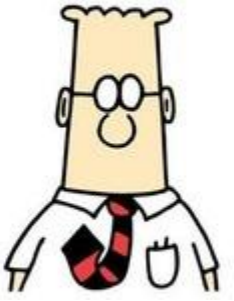
TMN4133 System Programming
GROUP PROJECT



Project Name	**:** System Programming Project
No of group members	**:** 4 students
Release Date	**:** Thursday, 17th November 2022 (Week6)
Due Date	: Friday, 6th January 2023, (Week 12)
Weight	: **15%**
Hardware & Software	: Access to terminal (in Ubuntu or any Linux distro
Requirements	: text editor (vi, nano, etc)
	: GNU glibc and gcc compiler

SCENARIO

You are currently working as a system programmer in a company in JB. One day, your boss calls you up to his office, and the following conversation ensued:

Boss: Hi <your name here>, how are you doing today?

You: I'm ok, how about you boss?

Boss: I'm fine, thank you. Let's get to business, I know you are good at system programming….

You: yes, I study that at my university last time.

Boss: Hmmm, I have a task for you. I want to do the following tasks from the command (linux) line, but no existing command is able to handle all these tasks. I want you to write a C program using the necessary system calls to accomplish the following tasks.

The new tool/command should be able to do this:

- A menu system for selecting which operation to be conducted. Options are:
  - 1 for file operations
  - 2 for directory operations
  - 3 for storing keystrokes from keyboard

- For **(1) file operations**, I want to be able to do the following tasks, which are available through another submenu (all the following must be implemented using appropriate system call):
  - Create/Open a new file and have the ability to specify the filename and the path.
  - Change the permission of a file based on user input.

  - **`read`** from a given file and print to the standard output.
  - Remove or delete a file given the file name.

- For **(2) directory operations**, I want to be able to do the following tasks, which are available through another submenu:
  - Create a directory (**`mkdir`**) using a given name and specified path (if any).
  - Remove (delete) a directory using a given name and specified path (if any).
  - Get the current working directory.
  - List current directory (and files). Able to list file(s) in the current directory (use readdir)
- For **(3) keylogger operations**, I want to spy on my boyfriend/girlfriend:
  - Run in background of the terminal
  - Record any keystroke from the keyboard to a textfile named “keylog.txt”
  - Add timestamp at the beginning of the session in the above keylog.txt file

For any of the operations above, you need to print out statement to the user whether the operation is successful or not.

Please name your program above as **`supercommand.c`**

ALL the above tasks need to be executed from command line, not from menu system. For example,

Example Task 1: File operation: to create new file, the command would be like this:

```
./supercommand -m 1 1 newFile.txt
```

Whereby,
1st parameter - 1 - refers to file operation mode
2nd parameter - 1 - refer to first operation for file operation which is to create new file
3rd parameter – newFile.txt – name of the file to be created

Example Task 2: Directory operation, list content of current directory

```
./supercommand -m 2 3 .
```

Whereby,
1st parameter - 2 - refers to directory operation mode
2nd parameter - 3 - refer to third operation for file operation which is to list directory content
3rd parameter – . refer to current directory

Example Task 3: Directory operation, list content of current directory

```
./supercommand -m 3 keylog.txt
```

Whereby,

1st parameter - 3 - refers to keylogger operation mode
2nd parameter - **`keylog.txt`** - refer to log file to store the keystrokes

By the way, don't forget to create the **man page** for the above command. Don't expect me to remember how to run the command !!!

You: Boss, you want me to do all the above!!!! How long do I have?

Boss: Since you are a good system programmer, how about 6 weeks?

You: <Noooooooooooo….> (quietly, of course). Errr Boss, am I doing this myself?

Boss: Hmmmmm, well actually it's quite a demanding task, why don't you get another 3 people and do this as a team. You can choose whoever you like.

You: Fewhhhh, ok Boss, consider this done. See you in 6 weeks time.

For the report, include the following:

1. Faculty Assignment Cover Page
2. A list of your group members, their tasks, and percentage of work done by each member.
3. How to compile and execute the program.
4. A set of test cases for each of the operations from your program.

   For example:

   Test Case 1: Running the program
   <screen shot>

   Test Case 2: Displaying the main menu.
   <screenshot>

   Make sure to include error handling such as file not found, etc.

5. Complete source code.

Marking Rubric

Part 1: **Correctness** - Compile successfully - Correct operation - Correct output - Appropriate error handling and message	60%	- (both compile time, run time and the result) and adherence to the program specification. - working program is the premise of the grading. If a program can't be successfully compiled, all 60% of part-1 will be lost, because basically the program can't be tested if the compile is not passed. So be sure to make your program compile before turning in. If some required function or operation is not implemented or implemented incorrectly, the portion of that function or operation in each of part-1, part-2 and part-3 will be deducted accordingly.
Part 2: man page	20%	- Create man page for the command created
Part 3: program design and coding style	10%	- Use proper style and indentation - Include where necessary
Part 4: Documentation	10%	- Documentation is expected to be in standard English. "Documentation" mentioned above includes comments in the code, README file and any other required analysis summary specified in the project write-ups.

What needs to be submitted?

1. Softcopy of report containing the statement of problem, how to compile and execute, testing, sample output showing all type of output, and source code.

2. Soft copy of all programs and executable, zip according to the following format and uploaded to course cube in Odyssey/Morpheus.

   a. Format for zip file name - **`<GroupNo>-Project.zip`**
   Example: **`Group01-Project.zip`**

.: **End of Project Specification** :.