Assignment 1: Intro to System Calls

Operating Systems and Networks (Monsoon 2020)

DEADLINE: 22nd August, 11:55 PM. There will be no deadline extension.

Part 1:

Task:

Given a file, you need to reverse the contents of the file and store the result in a new file in the directory named “Assignment”. The percentage of the file written should be printed on the console during file writing. The directory created should have read, write, and execute permissions for the user who created it. The new file created should have the read and write permissions for the user who created it. The program will be tested on LARGE (> 1GB) files which could be greater than RAM size.

Example:

Input: `‘A.txt’` ⟶ `“My name is abcd”`

Output: `‘Assignment/A.txt’` ⟶ `“dcba si eman yM”`

Note:

- File Path will be passed as a command-line argument. Ex:

  ```
  $ ./a.out <path_of_file_to_be_reversed>
  ```
- The percentage of the file written should be overwritten each time (shouldn’t write multiple times).
- The output file should have the same name and extension as the input file.

Part 2:

Task:

Write a program to

1. Check the permissions for the two files and the directory.
2. Check whether the contents in the new file are the reverse of the old file.

The input to this program has paths for `newfile`, `oldfile`, and the `directory`.

Example:

```
Directory is created: Yes
Whether file contents are reversed in newfile: Yes
User has read permissions on newfile: Yes
User has write permission on newfile: Yes
User has execute permission on newfile: No
Group has read permissions on newfile: No
Group has write permission on newfile: No
Group has execute permission on newfile: No
Others has read permissions on newfile: No
Others has write permission on newfile: No
Others has execute permission on newfile: No
```

Note:

- The above 9 should be printed for the old file and the directory too.
- Path of newfile oldfile and directory will be passed as a command-line argument.

  ```
  $ ./a.out <newfile> <oldfile> <directory>
  ```

Guidelines:

- All Programs must use **system calls only**. Use of printf and scanf are restricted.
    - You may use sprintf for formatting strings.
    - Use of string.h library is permitted
- Useful commands: read, write, lseek, stat, fflush, perror.
- Use **man** pages exclusively.
- Assignment should be coded in C. Indent your codes.
- Implement basic **error handling**. For example, if a file does not exist.

  Tip: Check return value of system calls on man pages
- For any assumptions made, mention them clearly in a Readme File
- Submission_format: **<RollNo>_Assignment1.tar.gz**
- Directory Structure:

```
2019XXXXXX_Assignment1
        |__ Q1.c
        |__ Q2.c
        |__ README.md
```

- Submission by email to TAs will not be accepted.
- **Plagiarism will lead to serious consequences.**