Systems Software/Programming – Lab Manual

Lab 5 – File I/O using System Calls

Write C program for each of the problem below. After executing each of the C program, you will need to capture the output in text file format.
C program file will be named as StudentID_Lab5_x.c
Text file with output captured will be names as StudentID_Lab5_x.txt

Problem 1: Write a program (StudentID_Lab5_1.c) to implement following 3 commands as different functions

- cat <filename> - it takes parameter as file name including path and displays the content.
- ls <directory> – it takes parameter as directory including path and lists all the files in that directory (single dot "." Indicates current directory and double dots ".." indicates parent directory
- mv <source> <destination> – it take 2 parameters: source and destination. source is the filename including path which you need to move, destination is either a directory or a new filename including path. If destination is a directory files is moved to this directory but if destination is filename then you move the file with a new name (i.e. move and rename operations combined), for example
    - mv ~/mydir/abc.txt ~/mydir/def.txt → only rename file abc.txt to def.txt in the same directory ~/mydir (~ indicates home folder) but don't perform move.
    - mv ~/mydir/abc.txt ~/mydir_new/def.txt → rename abc.txt to def.txt but also move the file from ~/mydir to ~/mydir_new
    - mv ~/mydir/abc.txt ~/mydir_new/ → only move the file abc.txt from ~/mydir to ~/mydir_new but don't rename
- Test each of the function by calling them from main() function.

Problem 2: Implement your own shell in StudentID_Lab5_2.c

- Give Prompt myshell@<user>@<hostname>$ for user to type in command. E.g. user who has logged in is "student" on hostname "lab004_15" then the prompt should be displayed as myshell@student@lab004_15$

- Prompt waits for user to enter the command. It will accept commands listed below. For the first 3 commands, you will be calling the functions implemented in Problem1. For the forth command exit, you should exit from your own shell and return back to Parent Shell prompt.
    - cat <filename>
    - ls <directory>
    - mv <source> <destination>
    - exit

Problem 3: Write a program StudentID_Lab5_3.c to see the difference between hard links and symbolic links

- Using link() system call create hard link to a given text file as a command line argument
- Using symlink() system call create symbolic link to the same text file as above.
- Using ls command display the value of inodes for original file, hardlink and symbolic link and understand the output.

Submission:
StudentID_Lab5.zip with total 6 files (3 C program files + 3 captured output text files)

Optional Assignment:

Write a program to implement ls –r command which recursively goes through all the subdirectories to list the directory content.