Operating Systems(PG)

Monsoon 2015

Assignment 1

Deadline: Wednesday 26th August 2015 7:00 PM

Date : 19th August 2015

PART 1: Word count

Write a program to simulate “wc” command.

Options to be implemented:
-l : Number of lines
-c : Number of characters
-m : Number of bytes
-L : Length of longest line
-w : Number of words

Input format :
ubuntu:~$ ./a.out -lc abc.txt
ubuntu:~$ ./a.out -lmc /home/user/Desktop/Assignment/abc.txt
ubuntu:~$ ./a.out -c -L ../../Assignment/abc.txt
ubuntu:~$ ./a.out -w Assignment/abc.txt
ubuntu:~$ ./a.out abc.txt

All options and file name will be given through command line arguments only. You should handle all combinations of the options.

Output format :
Output should be similar to the output provided by “wc” command in shell.

PART 2: More command

Write a program to simulate “more” command.

Input format:

ubuntu:~$ ./a.out <#lines> <filename> [file ...]

More command displays a fixed number of lines(given by argument #lines) from the given file(s) to the output console. After initially displaying <#lines> number of lines, your program should accept any of the user inputs - Enter↲, space or ‘q’. On pressing

- Enter↲ : Your program should display the next line from file or terminate if EOF occurs.
- Space : Your program should again display <#lines> number of lines.
- q : Your program should terminate.

Your program should run indefinitely until either EOF occurs or ‘q’ is pressed!
Refer more command man page to understand the working.

Examples:

ubuntu:~$ ./a.out 4 abc.txt

Here #lines = 4. So initially display 4 lines from abc.txt. After that for any of the above mentioned 3 user inputs, display output accordingly.

ubuntu:~$ ./a.out 5 abc.txt abcd.txt

Here #lines = 5. In this, after displaying the contents of abc.txt, you need to display the lines from abcd.txt as per the user inputs.

Output format:

Should be similar to more command output.

PART 3: split + tac command

Write a program to split all the files in the given source directory, based on number of lines specified similar to shell command “split -l <n>” and
Copy them to the destination directory (create the path if it does not exist) by reversing the lines in each of the splitted files similar to shell command “tac <filename>“.

Input format:
ubuntu:~$ ./a.out <source dir> <path to destination dir> <#lines>
source dir, path and #lines will be given through command line arguments only!

Examples:
ubuntu:~$ ./a.out Assignment/ dir1/dir2/dest_dir 20
Here Assignment is the source directory. All files in this directory has to be split into chunks containing 20 lines each. dest_dir will now contain these chunk files with their lines in reverse order. The path dir1/dir2 need not exist. Check accordingly and create if it does not exist.

a. Input structure:
Consider the following structure for source directory ‘Assignment’.
Assignment
|_ abc
|_ jkl

b. Output structure:
Your dest_dir structure should contain the chunks as follows:
dir1
|_dir2
|_dest_dir
|_ abc_1
|_ abc_2
|_ ….
|_ abc_n
|_ jkl_1
|_ jkl_2
|_ ….
|_ jkl_m
Now, abc_1 will contain the lines 1-20 from abc in reverse order(20,19...1). Similarly for other chunks.
Note: All the files in source directory would be text files only.

Useful System Calls :

- read
- write
- open
- close
- mkdir
- access
- scandir
- readdir
- closedir

General Guidelines :

- CSE/CSIS/MS/PGSSP students have to implement all three parts. VLSI students can choose between part1 and part2, but part3 is mandatory.
- You are not supposed to use STLs or ‘system’ library function of linux. If found violated, your submission will not be evaluated.
- Indent and comment the code properly.
- Your name and roll number should be included as comments at the beginning of code.
- Due credit will be given to modularity of code.
- ZERO tolerance towards any kind of code plagiarism.
- Strictly follow the upload format and deadlines. All invalid submissions will not be considered for evaluation.
- Make sure you do not upload any executables.

Upload format:

Create a folder named your roll number(20XXXXXXX).
Inside that create three folders named Part1, Part2 and Part3
In each folder, place your '.c' or '.cpp' files.
Create a tar.gz of the above folder(20XXXXXXX) named “Assignment1.tar.gz” and upload it.