Operating Systems (PG) CSE531

Assignment # 1 – Develop Text Editor

Deadline: 26th August 17:00 PM

Objective :

Build a fully functional text editor resembling the vi editor, albeit with a restricted feature set.

Prerequisites :

1. Basic usage and architectural know-how of the vi editor. [1][6]
2. Preliminaries such as C/C++ code compilation and execution, referring to linux man pages and utilizing the GNU debugger (very helpful). [4]

Learning objectives :

1. Gaining familiarity with basic programming in linux.
2. Use of general terminal interface libraries to interact and programmatically control the terminal.
3. Know how the linux terminal works under canonical (default) mode; and understand how to get finer control over raw input processing and raw output rendering.
4. Acquaint yourself with common linux system calls to perform numerous actions, along with other low level wrapper functions located in different low level header files (e.g. termios.h, unistd.h, sys/ioctl.h etc.)
5. Use of efficient data structures and algorithmic procedures for text manipulation, storage and rendering.

Specifications :

The vi editor works in 3 modes (6 in fact, but 3 are focused upon) :

1. Insert / Edit mode - Enables the user to insert new text and edit the text of the current file.
2. Normal mode (Default mode for vi) - Primarily used for navigation of the cursor within the file. Also allows for the manipulation of text using key combinations.
3. Command-line mode - For entering editor commands, your custom commands, as well as shell commands to be run.

The functionalities to be implemented in each of the modes are listed below:

Normal Mode

1. **Open / Load** : opens a new file OR loads an already existing file based on the command line argument provided during the execution of the program.
2. **Navigation** : Allows scrolling (both horizontal and vertical) using **h**, **j**, **k**, **l** character keys (***Bonus marks* : mapping arrow keys in addition).

3. **Single character replace** : Replace a character (position the cursor on the character to be changed and then press ‘**r**’, followed by the new character).
4. **Go to first line of file** : Pressing ***‘gg’*** should take the cursor to the first line of the file.
5. **Go to last line of file** : Pressing ***‘G’*** should take the cursor to the last line of the file.

Insert/Edit Mode

1. **Insertion:** Insertion of a character (or sequence of characters) at the current cursor position if you press a key (alpha-numeric). The cursor should also move one character ahead along with the character insertion.
2. **Deletion:** Deletion of a character (or sequence of characters) using *delete (forward deletion) / backspace (backward deletion).*

Command Line Mode

1. **File Save:** Saving the current file using “**:w**” command.
2. **Quitting (normally):** Quitting the editor using ***“:q”*** command - should raise a warning if the file hasn’t been saved since last changes.
3. **Quitting (forcefully):** Quitting the editor using ***“:q!”*** command - forcefully quits without caring whether or not file is saved.
4. **Running shell/editor commands:** Integrating editor with the shell using ***“:!<any linux command>”*** - should display the output of that command on the screen, then wait there until the user inputs a key and then come back to the editor. This is to be done using fork-exec system calls.

Switching between modes

1. Normal to Insert/Edit mode - on pressing key ***‘i’***.
2. Insert/Edit to Normal mode OR Command Line to Normal mode - on pressing key ***“esc”***.
3. Normal to command-line mode - by prefixing ***“:”*** in front of your command.

Display characteristics

1. Starting from the top-left corner of the terminal window it should display the file contents line by line. You should be able to handle text rendering if the terminal window size is varied.
2. Leave the last line of the display screen as empty (Status line) - to be used in normal and command-line modes.

Guidelines:

1. Select data structures with care. Insertion / deletion / replacement of sequences of characters must be possible at any point in your text.
2. Tab space handling : Tab spaces must be rendered on the screen as 4 spaces without modifying the occurrence of the actual tab space character in the file.

3. Your application will be tested on text files only (ASCII) and avoid hardcoding / assuming any details about that file.
4. Handle error cases wherever required.
5. Languages allowed: **C/C++**
6. You are not supposed to use *'system()'* library function or fork-exec to load the system's copy of vi/vim. If you violate this, your submission will NOT be evaluated.
7. Indent and comment your code properly.
8. **ZERO tolerance towards any kind of code plagiarism. Plagiarism will fetch you a ZERO. No negotiations. PERIOD.**
9. **DO NOT COPY/SHARE code/code-snippets (even a few lines of copied code would be detected and punished) - both the parties will get ZERO.**
10. Make sure you do not upload any executables.

Upload Format:

Upload Format will be shared on the course moodle before the submission deadline.

References:

[1] Introduction to the vi editor: https://www.tutorialspoint.com/unix/unix-vi-editor.htm

[2] VIM modes: https://en.wikibooks.org/wiki/Learning_the_vi_Editor/Vim/Modes

[3] Canonical vs. Non-canonical Terminal modes: https://stackoverflow.com/questions/358342/canonical-vs-non-canonical-terminal-input

[4] Beginning Linux Programming (Please read the **first chapter,** at least. It talks about how to compile your C/C++ source files, running the compiled and linked programs, using manual pages for reference, and debugging your programs in Linux. Please read **Chapters 5 and 6** as they are pertinent to this assignment.): https://doc.lagout.org/operating%20system%20/linux/Beginning%20Linux%20Programming%2C%204%20Ed.pdf

[5] Escape sequences (How to obtain fine control over the terminal): http://ascii-table.com/ansi-escape-sequences.php

[6] If you're really interested in the VIM editor: https://en.wikibooks.org/wiki/Learning_the_vi_Editor

[7] Vim Quick Reference: https://vim.rtorr.com/

[8] Spawning processes and fork-exec: https://en.wikipedia.org/wiki/Fork_(system_call), https://en.wikipedia.org/wiki/Fork%E2%80%93exec