DAY 4x/14 LEARNING ABOUT LINUX

4.X: TEXT EDITING WITH VI/VIM AND NANO

Within the Unix and Linux ecosystems, the capability to edit text files efficiently is crucial for software development, system administration, and data processing tasks. Two of the most prominent text editors that have stood the test of time are vi and nano. This section aims to introduce these editors, focusing on vi for its ubiquity and nano for its simplicity and ease of use, especially for beginners.

A. VI/VIM: THE UBIQUITOUS EDITOR

Originally developed by Bill Joy at the University of California, Berkeley, for the BSD UNIX system, vi (visual editor) has become a standard component across virtually all Unix systems. Its inclusion in both BSD UNIX and later in System V UNIX editions cemented its position as a universal tool for text editing within the Unix world.

a. Starting with Vi

To begin editing with vi, one can simply invoke the editor by typing vi followed by the file name. If the file doesn't exist, vi starts with a blank slate, ready to receive text. The interface, initially presenting a series of tildes (~) indicating empty lines, awaits user commands.

Vi operates primarily in two modes:

- **Command Mode**: The default state upon opening vi, where keystrokes are interpreted as commands for navigating, editing, or configuring the editor.
- **Insert Mode**: Activated by pressing keys like i (insert before cursor), I (insert at the beginning of the line), a (append after cursor), and several others as shown in next figure. In this mode, users can freely enter text until they exit back to command mode using the Esc key.

Key	Action
i	*Text is inserted in front of the cursor.*
I	*Text is inserted at the beginning of the current line.*
a	*Text is added after the cursor.*
A	*Text is added to the end of the current line.*
o	*Text is added after the current line.*
O	*Text is inserted before the current line.*
R	*Text is replaced (overwritten).*

b. Editing Features

VI is replete with commands for editing, including deleting text (x for characters, dw for words, dd for lines), replacing text (r for characters, cw for words), and pasting text. Navigation within the text is facilitated by a variety of cursor movement commands, as detailed in the table below:

Movement	Key sequence

Up one line	**<cursor up> or k**
Down one line	**<cursor down> or j**
Right one character	**<cursor right> or l (will not wrap around)**
Left one character	**<cursor left> or h (will not wrap around)**
To start of line	**^**
To end of line	**$**
Back one word	**b**
Forward one word	**w**
Forward to end of current word	**e**
To top of screen	**H**
To middle of screen	**M**
To bottom of screen	**L**
Down a half screen	**Control-D**
Forward one screen	**Control-F**
Up a half screen	**Control-U**
Back one screen	**Control-B**
To line nn	**:nn<Enter> (nnG also works)**
To end of file	**G**

Table zzz: offers aA quick reference to the cursor movement commands in vi, facilitating navigation and editing within the text editor

c. Saving and Exiting

To save changes and exit vi, the command :**wq** is used. Alternatively, to exit without saving, one can use :**q!**.

B. NANO: THE FRIENDLY EDITOR

Contrasting with vi, nano offers a more straightforward and intuitive interface, making it an excellent choice for those new to the Unix and Linux environment. It displays helpful shortcuts at the bottom of the window and does not require switching modes to insert text or execute commands.

Using Nano

Starting nano is as simple as typing nano followed by the file name. The editor opens with the file's content ready for editing. Commands for saving (**Ctrl + O**), exiting (**Ctrl + X**), cutting (**Ctrl + K**), and pasting (**Ctrl + U**) text are directly accessible and displayed at the bottom of the editor window, guiding users through their editing tasks.

CONCLUSION

Both vi and nano offer powerful capabilities suited to different user preferences and needs. vi's command-driven interface is highly efficient for those willing to climb its learning curve, while nano provides an accessible entry point for new users with its straightforward, modeless operation.

Homework TC 4x

a) Homework 1: Basic Navigation and Editing

Objective: Familiarize yourself with basic vi navigation and text insertion.

1. **Opening and Navigating**: Open sample.txt and use vi navigation keys (h, j, k, l) to move the cursor around the text.
2. **Text Insertion**: Insert a new line of text at the beginning and end of sample.txt.

Opening and Navigating:

Open sample.txt with vi sample.txt.

Use j to move down, k to move up, h to move left, and l to move right.

Text Insertion:

To insert at the beginning: Open sample.txt, press i to enter insert mode at the cursor, enter your text, and press Esc to exit insert mode.

To insert at the end: Press G to go to the last line, press o to insert a new line below, enter your text, and press Esc.

b) Homework 2: Text Manipulation

Objective: Master basic text manipulation commands in vi.

1. **Deleting Text**: Delete the third line in data.txt.
2. **Replacing Text**: Replace the first occurrence of "error" with "warning" in log.txt.

Deleting Text:

Open data.txt, navigate to the third line using j, and press dd to delete the line

Replacing Text:

Open log.txt, navigate to the line containing "error" or press :/error and then Enter to search, press cw to change the word, type "warning", and press Esc

c) Homework 3: Advanced Editing Techniques

Objective: Employ advanced vi editing techniques for efficient text processing.

1. **Copy and Paste**: Copy the first two lines of article.txt and paste them after the fifth line.
2. **Find and Replace**: Perform a global find and replace to change all instances of "http" to "https" in urls.txt.

Copy and Paste:

Open article.txt, move to the first line with gg, press yy twice to copy the first two lines, press 5G to move to the fifth line, and press p to paste below it

Find and Replace:

Open urls.txt, press : to enter command mode, type %s/http/https/g to replace all occurrences of "http" with "https", and press Enter.

d) Homework 4: Multi-file Editing and Search

Objective: Utilize vi to work with multiple files and execute search operations.

1. **Multi-file Editing**: Open file1.txt and file2.txt in vi and practice switching between them using :n and :prev.
2. **Search Within a File**: Search for the word "confidential" in report.txt and move the cursor to its first occurrence.

Multi-file Editing:

Open vi with vi file1.txt file2.txt. Use :n to switch to file2.txt and :prev to return to file1.txt.

Search Within a File:

Open report.txt, type :/confidential and press Enter to search for "confidential". Press n to navigate to subsequent matches.

e) Homework 5: Customizing and Using Buffers

Objective: Explore vi customization and the use of buffers for advanced editing.

1. **Customizing vi**: Create a .vimrc file in your home directory and set vi to display line numbers by default.
2. **Using Buffers for Text Movement**: Move a paragraph from chapter1.txt to chapter2.txt using vi buffers.

Customizing vi:

Create or edit .vimrc in your home directory with vi ~/.vimrc, add set number to always show line numbers, and save with :wq.

Using Buffers for Text Movement:

Open chapter1.txt, highlight the paragraph to move with V (visual line mode) and move down or up to select, press d to cut, open chapter2.txt with :e chapter2.txt, navigate to the insertion point, and press p to paste.

Submission Guidelines

- Ensure your scripts are well-commented to explain your logic and approach.
- Test your scripts with sample data to verify correctness.
- Submit/export history and upload it as usual.