

RHSA (Admin 1) Red Hat Enterprise Linux 8 Essentials

Prepared by Eng. Taghreed Elsayed

Lecture 3

جامعة الجميع الذكية

تسعى لتكون الكيان الأول الذي يتبنى مبدأ المجتمع التعليمي
التعاوني ويقدم العلم مقابل العمل لتنمية وبناء المجتمعات.

CREATING, VIEWING, AND EDITING TEXT FILES



جامعة الجميع الذكية
EVERYONE'S SMART UNIVERSITY

CREATING, VIEWING, AND EDITING TEXT FILES

EDITING TEXT FILES WITH A GRAPHICAL EDITOR

- ❑ **EDITING FILES WITH GEDIT** The gedit application is a full-featured text editor for the GNOME desktop environment. Launch **gedit** by selecting **Applications → Accessories → gedit** from the **GNOME menu**.
- ❑ Like other graphical applications, **gedit** can be started without navigating the menu. Press **Alt+F2** to open the Enter a Command dialog box. Type **gedit** and press **Enter**.

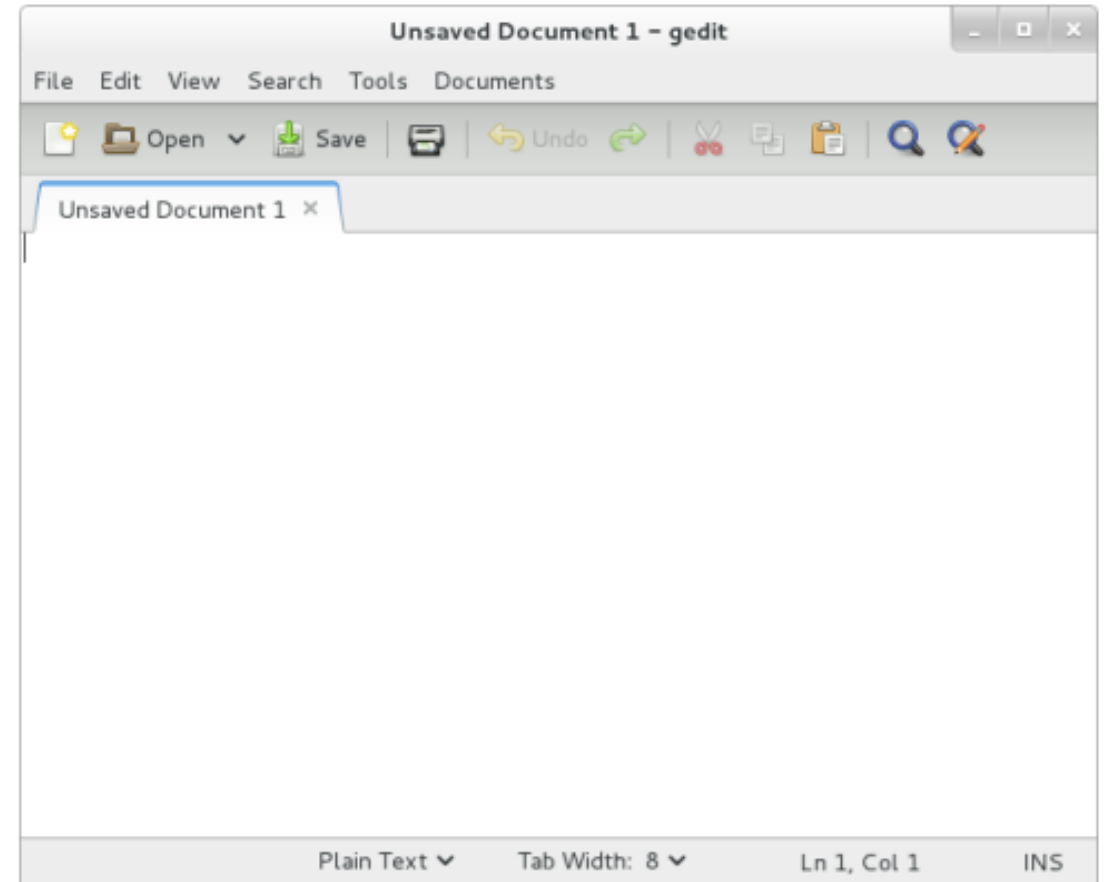


Figure 4.11: gedit text editor

- ❑ GNOME Help includes a gedit help guide, which may be found by selecting Applications → Favorites → Help from the GNOME menu.
- ❑ Then select Go → All Documents to view the list of graphical applications. Scroll down to select the gedit Text Editor hyperlink.

❖ Basic gedit Keystrokes :

- Perform many file management tasks using gedit's menu:
 - To create a new file in gedit, click the blank paper toolbar icon, or select File → New (Ctrl+n) from the menu.
 - To save a file, click the disk-drive save toolbar icon, or select File → Save (Ctrl+s) from the menu.

- To open an existing file, click the Open toolbar icon, or select File → Open (Ctrl+o) from the menu.
- **The Open Files dialog window will display from which users can locate and select the file to open.**
- Multiple files may be opened simultaneously, each with a unique tab under the menu bar. Tabs display a file name after being saved the first time.

COPYING TEXT BETWEEN GRAPHICAL WINDOWS

- Text can be copied between documents, text windows, and command windows in the graphical environment.
 - Selected text is duplicated using copy and paste or moved using cut and paste.
- Whether cut or copied, the text is held in memory for pasting into another location. To select text:



- Click and hold the left mouse button before the first character desired.
- Drag the mouse over and down until all required text is in a single highlighted selection, then release the left button. Do not click again with the left button, as that deselects the text. To paste the selection, multiple methods can accomplish the same result. In the first method:
 - **Click the right mouse button anywhere on the text area just selected.**
 - From the resulting context menu, select either cut or copy.
 - Move the mouse to the window or document where the text is to be placed, click the left mouse button to position where the text should go, and click the right mouse button again, now choosing paste. Here is a shorter mouse technique to practice:
 - First, select the text.
 - Hover the mouse over the destination window and click the center mouse button, just once, to paste the text at the cursor. This last method can only copy, not cut. The original text remains selected and can be deleted. As with other methods, the text remains in memory and can be repeatedly pasted.



Chapter 04

CREATING, VIEWING, AND EDITING TEXT FILES

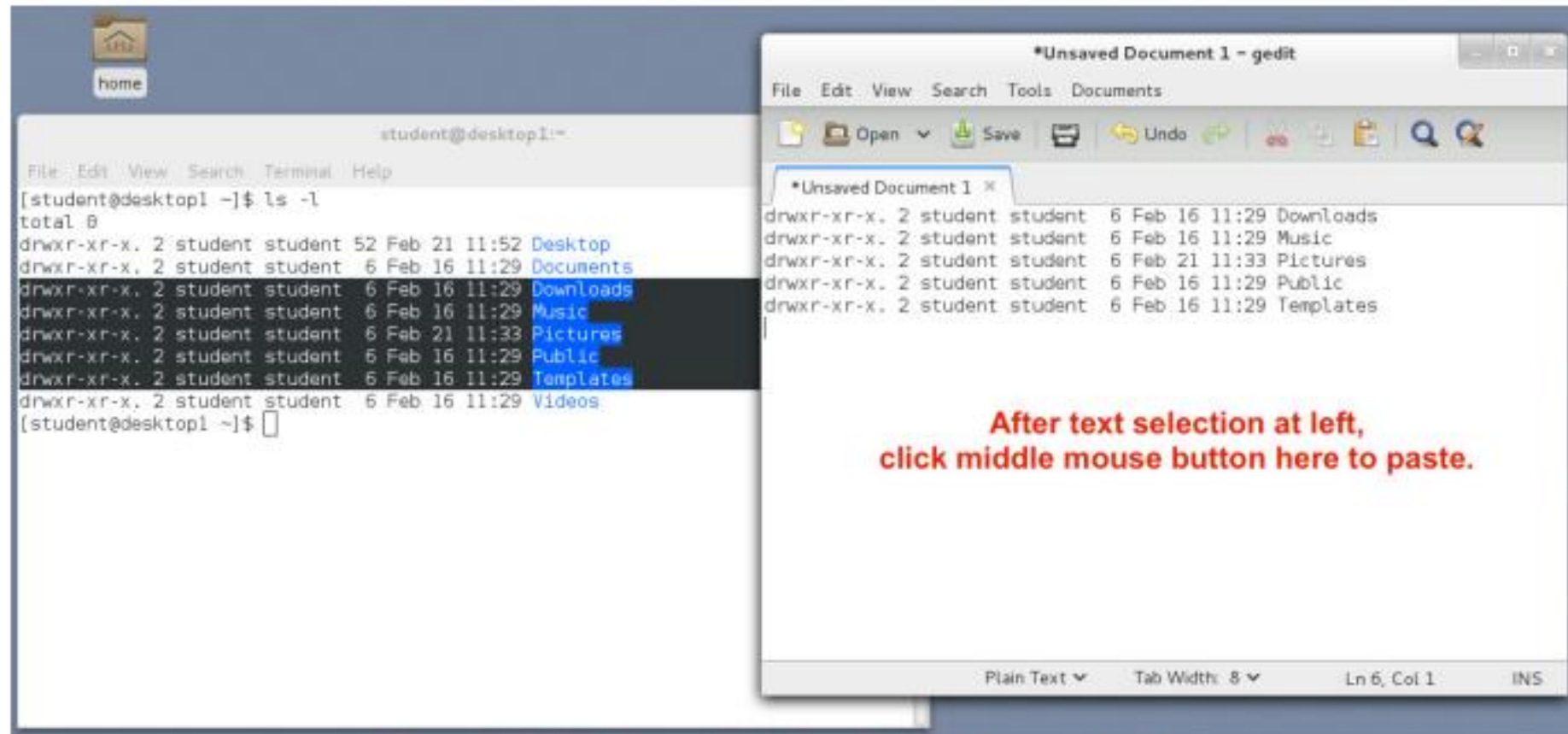


Figure 4.12: Select and paste using middle mouse button

The keyboard shortcut method can also be used in graphical applications:

- First, select the text.
- Use Ctrl+x to cut or Ctrl+c to copy the text.
- Click the location where the text is to be placed to position the cursor.
- Use Ctrl+v to paste.

EX: you will edit a file with gedit, selecting text and pasting it into the editor.

☐ **OUTCOMES** An edited list of the configuration files found in the user's home directory.

- ❑ 1. Redirect a long listing of all home directory files, including hidden, into a file named `gedit_lab.txt`. Confirm that the file contains the listing.

```
[student@serverX ~]$  
cd  
[student@serverX ~]$  
ls -al > gedit_lab.txt  
[student@serverX ~]$  
cat gedit_lab.txt
```

- 2. Open the file with the `gedit` text editor. Include the ending ampersand so that the shell prompt can return while `gedit` is running.

CREATING, VIEWING, AND EDITING TEXT FILES

```
[student@serverX ~]$  
gedit gedit_lab.txt &
```

3. Insert the date at the top of your file document. 3.1. In the shell command window, display today's date with day of the week, month, date, and year.

```
[student@serverX ~]$ date +%A", "%B" "%d", "%Y
```

```
Friday, February 21, 2014
```

3.2. Select the text using the mouse.

```
[student@desktop1 ~]$  
[student@desktop1 ~]$ date +%A", "%B" "%d", "%Y  
Friday, February 21, 2014  
[student@desktop1 ~]$
```

- 3.3. Insert the text at the top of the file document. Switch to the **gedit** window. Using arrow keys, place the cursor at the upper-left corner of the document. Press the middle mouse button to paste the text.
- 3.4. Press **Enter** one or more times at the end of the inserted text to open blank lines above the file listing.

CREATING, VIEWING, AND EDITING TEXT FILES

4. Insert a description for this document, including your username and host name, on line 2.
- 4.1. In the shell command window, create descriptive text using shell expansion concepts to include the username and host name where the file list was generated.

```
[student@serverX ~]$ echo "$USER's configuration files on" $(hostname)
student's configuration files on serverX.example.com
```

- 4.2. Select the text using the mouse.

```
[student@desktop1 ~]$
[student@desktop1 ~]$ echo "$USER's configuration files on" $(hostname)
student's configuration files on desktop1.example.com
[student@desktop1 ~]$
```

- 4.3. Insert the text on the second line of the file document. Switch to the **gedit** window. Using the arrow keys, place the cursor at the second line's leftmost character. Press the middle mouse button to paste the text.
- 4.4. Press **Enter** or **Delete**, as necessary, to maintain blank lines above the file listing.

5.Remove file lines that are not hidden configuration files or directories.

5.1. Remove the “total” line at the beginning of the listing.

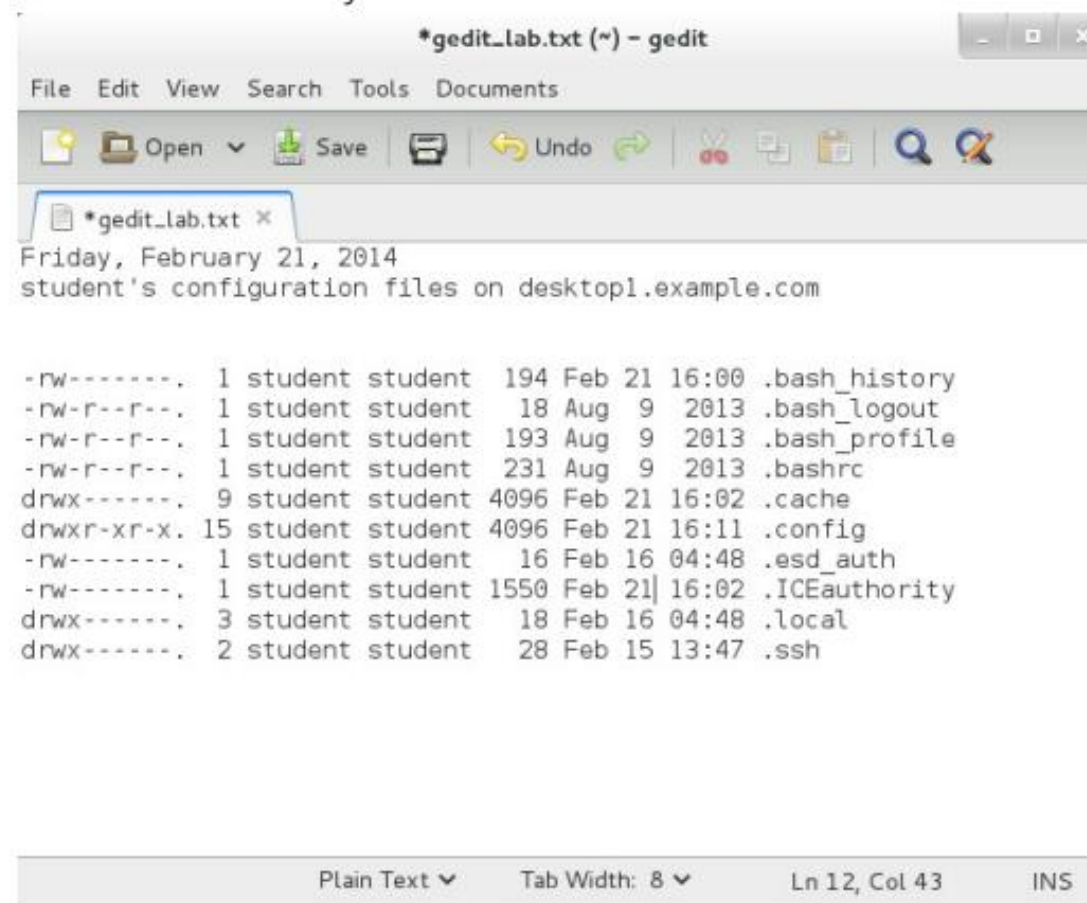
5.2. Remove the two lines representing the current directory and the parent directory.

5.3. Remove lines for file names that do not start with a dot. Do not edit or remove lines for hidden files or directories that begin with a dot.

6.The final file document should be similar to the following image. Manually edit the file to make corrections. The asterisk in the document tab or window header is a reminder of edits unsaved. Save the file and exit gedit.

Chapter 04

CREATING, VIEWING, AND EDITING TEXT FILES



```
*gedit_lab.txt (~) - gedit
File Edit View Search Tools Documents
Open Save Undo
*gedit_lab.txt x
Friday, February 21, 2014
student's configuration files on desktop1.example.com

-rw-----. 1 student student 194 Feb 21 16:00 .bash_history
-rw-r--r--. 1 student student 18 Aug 9 2013 .bash_logout
-rw-r--r--. 1 student student 193 Aug 9 2013 .bash_profile
-rw-r--r--. 1 student student 231 Aug 9 2013 .bashrc
drwx-----. 9 student student 4096 Feb 21 16:02 .cache
drwxr-xr-x. 15 student student 4096 Feb 21 16:11 .config
-rw-----. 1 student student 16 Feb 16 04:48 .esd_auth
-rw-----. 1 student student 1550 Feb 21 16:02 .ICEauthority
drwx-----. 3 student student 18 Feb 16 04:48 .local
drwx-----. 2 student student 28 Feb 15 13:47 .ssh

Plain Text Tab Width: 8 Ln 12, Col 43 INS
```

➤ EDITING FILES WITH VIM

- ❑ A key design principle of Linux is that information is stored in text-based files.
- ❑ Text files include both flat files with rows of similar information, such as configuration files in /etc, and Extensible Markup Language (XML) files, which define data structure through text tags, seen in application configuration files throughout both /etc and /usr.
- ❑ The advantage of text files is that they can be moved or shared between systems without requiring conversion, and can be viewed and edited using any simple text editor.
- ❑ Vim is an improved version of the vi editor distributed with Linux and UNIX systems. Vim is highly configurable and efficient for practiced users, including such features as split screen editing, color formatting, and highlighting for editing text.

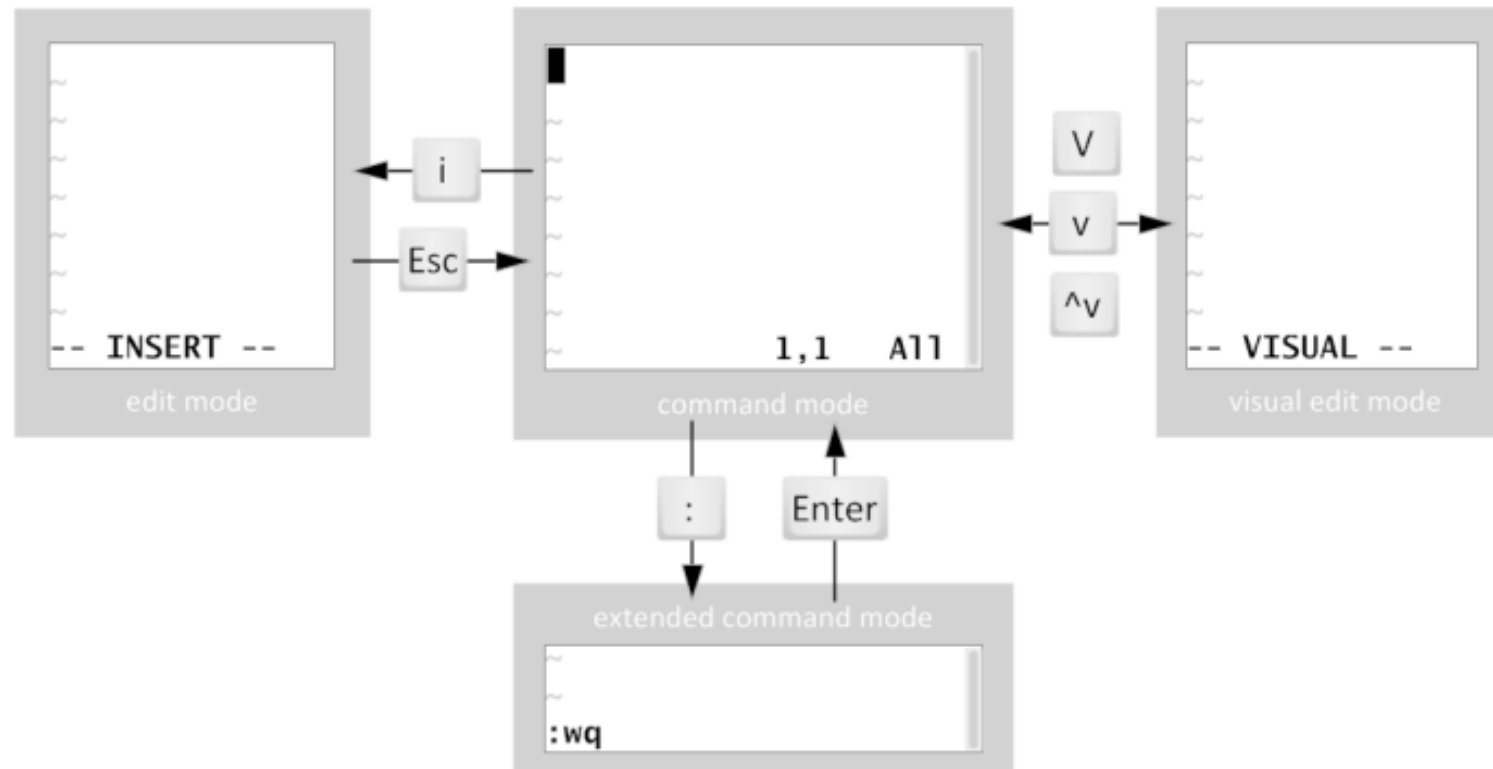


Figure 4.10: Moving between Vim modes

When first opened, Vim starts in command mode, used for navigation, cut and paste, and other text manipulation. Enter each of the other modes with single character keystrokes to access specific editing functionality:

- An **i** keystroke enters insert mode, where all text typed becomes file content. Pressing **Esc** returns to command mode.
- A **v** keystroke enters visual mode, where multiple characters may be selected for text manipulation. Use **V** for multi-line and **Ctrl+v** for block selection. The same keystroke used to enter visual mode (**v**, **V** or **Ctrl+v**) is used to exit.
- The **:** keystroke begins extended command mode for tasks like writing the file to save it, and quitting the Vim editor.

The minimum, basic Vim workflow

- ❑ **Vim has efficient, coordinated keystrokes for advanced editing tasks.** Although considered useful with practice, Vim's capabilities can overwhelm new users. The following workflow presents the minimum keystrokes every Vim user must learn to accomplish any editing task.
- ❑ **The instructor will demonstrate a typical file editing session using only basic Vim keystrokes**
- ❑ **1. Open a file with vim filename.**
- ❑ **2. Repeat this text entry cycle, as many times as the task requires:**
 - Use arrow keys to position the cursor.
 - Press i to enter insert mode.
 - Enter text. • Press Esc to return to command mode.
 - If necessary, press u to undo mistaken edits on the current line.

3. Repeat this text deletion cycle, as many times as the task requires:

- Use arrow keys to position the cursor.
- Press x to delete a selection of text.
- If necessary, use u to undo mistaken edits on the current line.

4. **To save or exit, choose one of the following to write or discard file edits:**

- Enter :w to write (save) the file and remain in command mode for more editing.
- Enter :wq to write the file and quit Vim.
- Enter :q! to quit Vim, but discard all file changes since the last write.

Rearranging existing text

- ☐ In Vim, copy and paste is known as yank and put, using command characters **y** and **p**. Begin by positioning the cursor on the first character to be selected, then enter visual mode. Use arrow keys to expand the visual selection.
- ☐ When ready, press **y** to yank the selection into memory. Position the cursor at the new location, then press **p** to put the selection at the cursor.
- ☐ The instructor will demonstrate “yank and put” using visual mode.
- ☐ 1. Open a file with **vim filename**.
- ☐ 2. Repeat this text selection cycle, as many times as the task requires:
 - Use arrow keys to position the cursor to the first character.
 - Press **v** to enter visual mode.
 - Use arrow keys to position the cursor to the last character.
 - Press **y** to yank (copy) the selection.
 - Use arrow keys to position the cursor at the insert location.



- Press p to put (paste) the selection.
- 3. **To save or exit, choose one of the following to write or discard file edits:**
 - Enter :w to write (save) the file and remain in command mode for more editing.
 - Enter :wq to write the file and quit Vim.
 - Enter :q! to quit Vim, but discard all file changes since the last write.

EDITING FILES WITH VIM

- ❖ In this exercise, use the existing Vim tutorial bundled with the Vim editor.
- ❖ The installed vim enhanced package provides vim tutor.
- ❖ For each exercise step, use the corresponding lesson in vim tutor to practice.
Return here when the lesson step is complete.
- ❖ Perform the following steps on server X unless directed otherwise.



1. Open vimtutor.

```
[student@serverX ~]$  
vimtutor
```

- ☐ Therefore, vi was designed with commands using only standard character keys, such as the conveniently grouped h, j, k, and l. Here is one way to remember them: hang back, jump down, kick up, leap forward.
- ☐ 2. Return to the vimtutor
- ☐ 3. Return to the vimtutor window.
- ☐ Vim has faster, more efficient keystrokes to delete an exact amount of words, lines, sentences, and paragraphs. However, any editing job can be accomplished using only x for single-character deletion.

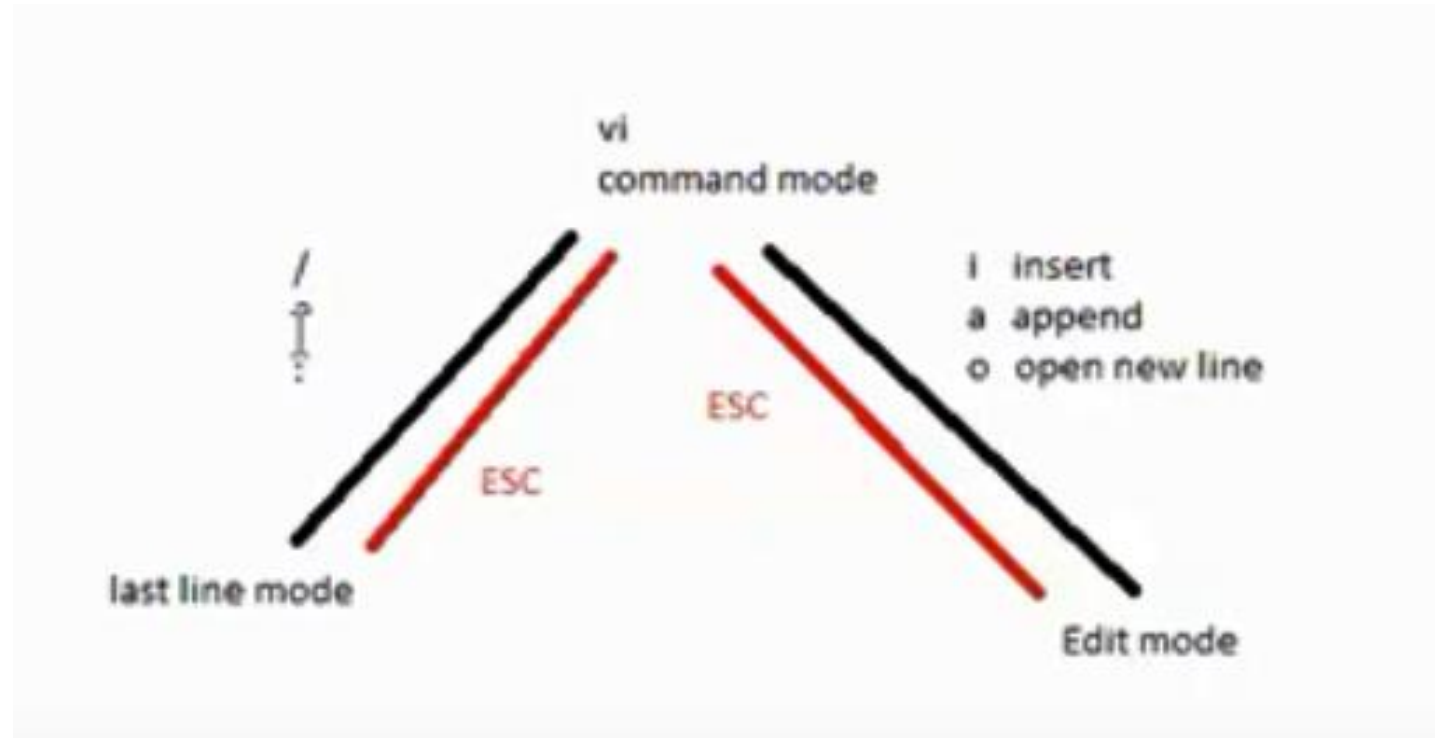
❑ 4. Return to the vim tutor window.

➤ The minimum required keystrokes are for entering and leaving edit mode, arrow keys, and deleting.

For most edit tasks, the first key pressed is i. 5. (Optional) Return to the vimtutor window.

➤ only the i (insert) command was taught as the keystroke to enter edit mode.

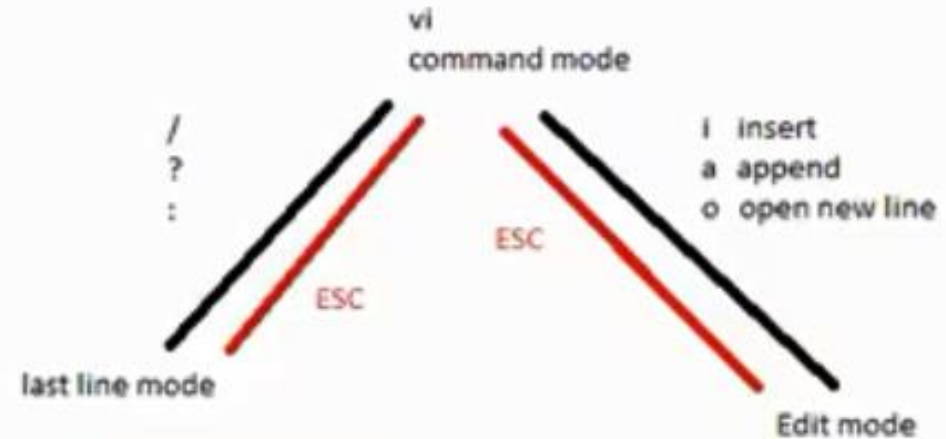
➤ This vim tutor demonstrates that other keystrokes are available to change the cursor placement when insert mode is entered. However, once in insert mode, all text typed is still file content.



CREATING, VIEWING, AND EDITING TEXT FILES

Navigation :

character : hjkl
word : w e
line : 1G 9G G
\$ 0



delete:

character : x
word : dw
line : dd D
undo : u
:1,9d

copy and paste:

line yy
p paste after
P paste Before
:4,12co15

search and replace:

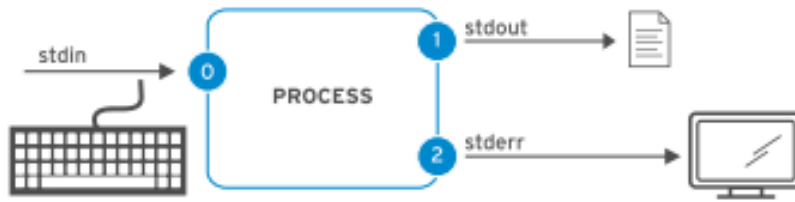
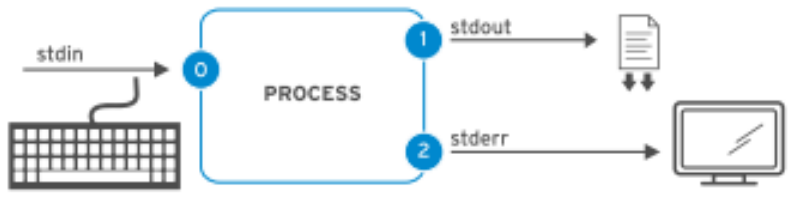
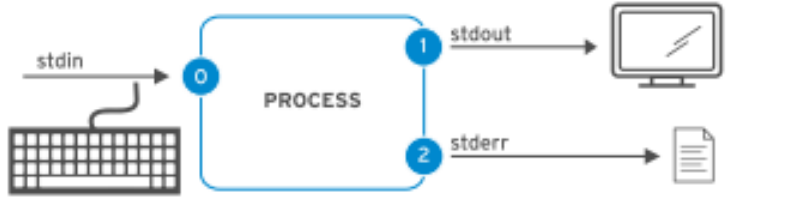
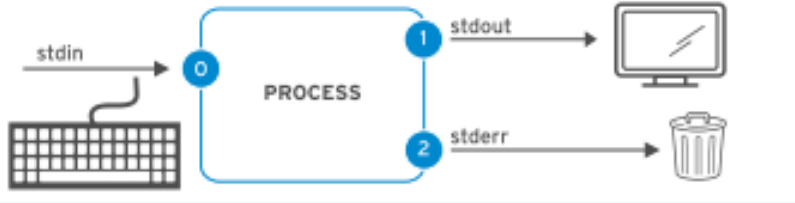
/word
?word
n find next
:%s/old/new/g

I

Chapter 04

CREATING, VIEWING, AND EDITING TEXT FILES

Output Redirection Operators

USAGE	EXPLANATION	VISUAL AID
> file	redirect stdout to overwrite a file	
>> file	redirect stdout to append to a file	
2> file	redirect stderr to overwrite a file	
2>/dev/null	discard stderr error messages by redirecting to /dev/null	



CREATING, VIEWING, AND EDITING TEXT FILES

>file 2>&1

&>file

redirect
stdout and
stderr to
overwrite
the same file



>>file 2>&1

&>>file

redirect
stdout and
stderr to
append to
the same file





جامعة الجميع الذكية
EVERYONE'S SMART UNIVERSITY

