

COMP311

Linux OS Laboratory

Lab2: Vi Editor

By

Alaa' Omar



Linux command line Editors



Emacs



Sublime



Gedit



Nano



Vim

Why Vi editor?



It is available in almost all Linux distributions.

It works the same across multiple platforms and distributions.

It is user friendly.

Command mode



Vi editor opens in this mode.



Move the cursor and cut, copy, and past the text.



Saves the changes to the file.



Commands are case sensitive.





Insert mode

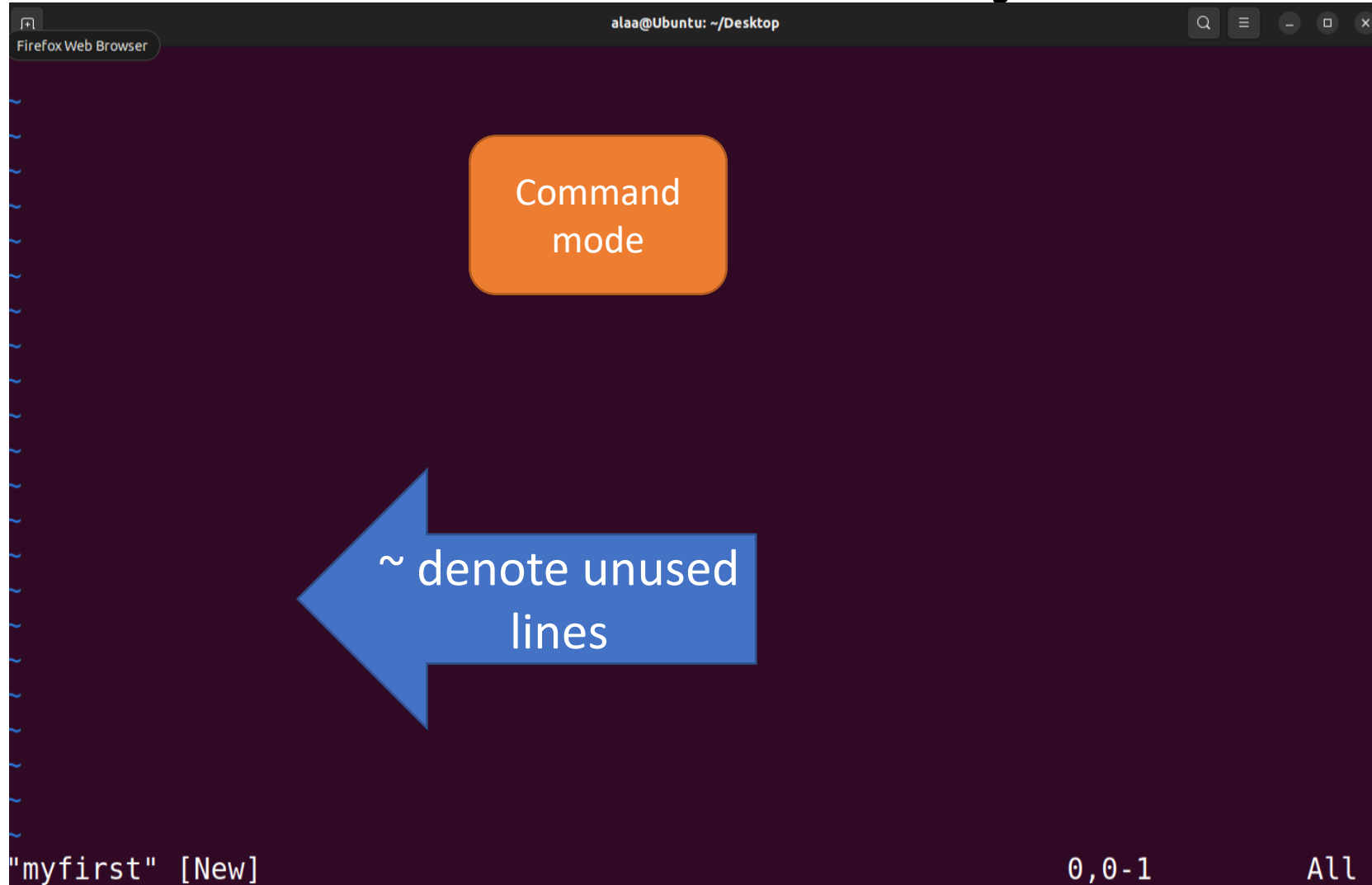
This mode is for inserting text in the file.

Press “**i**” on the keyboard for insert mode.

In the insert mode, any key would be taken as an input.

Press Esc key to save changes and return to command mode.

Start vi editor `$ vi myfirst`



Do the following:

`$mkdir lab2`

`$cd lab2`

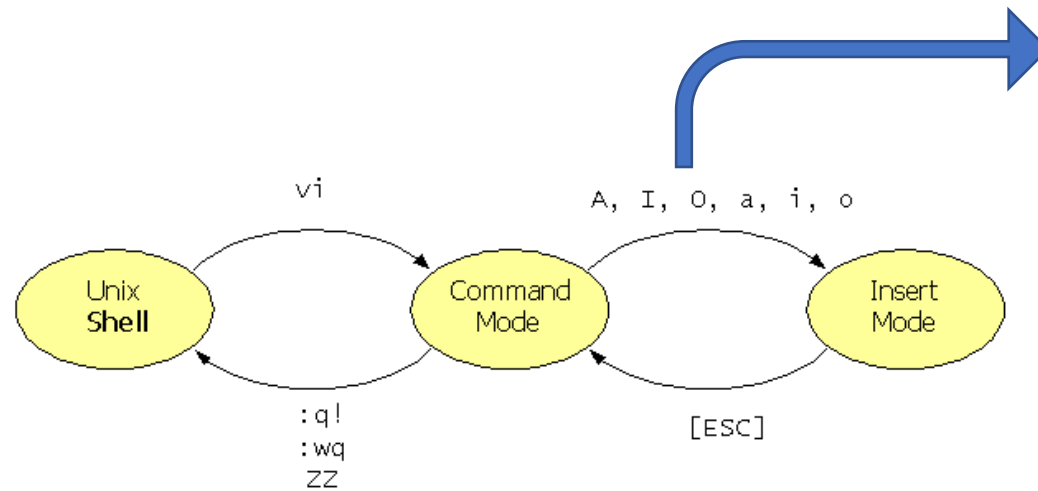
`$vi myfirst`

Your full name,

My id: 12265465

Computer science

Switch between vi modes



Editing action	Command
Insert text at current position	i
Insert text at beginning of line	I
Append text at current position	a
Append text to end of line	A
Open new line below cursor for new text	o
Open new line above cursor for new text	O

`:wq` write the file and quit

`:q!` Quit the editor (discarding edits)

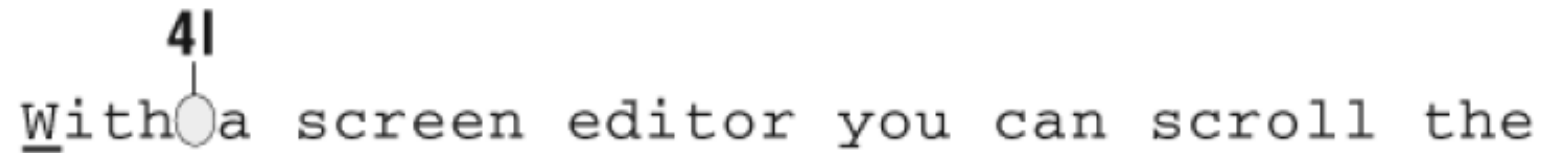
`zz` Quit vi, writing the file only if changes were made.

Single Movement in the command mode.

- *h*
Left, one space
- *j*
Down, one line
- *k*
Up, one line
- *l*
Right, one space

Movement	Commands
←, ↓, ↑, →	<i>h, j, k, l</i>
←, ↓, ↑, →	<div>BACKSPACE, CTRL-N and ENTER, CTRL-P, space bar</div>
To first character of next line	<i>+</i>
To first character of previous line	<i>-</i>
To end of word	<i>e or E</i>
Forward by word	<i>w or W</i>

Numeric Arguments



The screenshot shows a terminal window with a text editor. The text 'With a screen editor you can scroll the' is visible. Above the 'l' in 'With', the command '4l' is entered, indicating a numeric argument of 4 for the 'l' (down) command.

Figure 2-2. Multiplying commands by numbers

Try: in command mode , used 2h, 3l,3k,2j

Movement Within a Line

- *0* (*digit zero*) : Move to beginning of line.
- *\$* Move to end of line.

To end of line	\$
To beginning of line	0
To a particular line	G

Try: in command mode , used 2G, 3G\$, 1G0

Delete (cut) commands

x delete current character

#x delete number of characters (e.g., 4x deletes 4 characters starting from the current character).

dw delete current word

#dw delete # of words (e.g., 2dw or d2w deletes two words starting with the current words)

d\$ or **dd**: deletes the current line

#dd delete # of lines

Copy Command

yw : yank word (copy word)

#yw : yank number of words (2yw or y2w)

y\$ or **yy**: yank line

#yy yank number of lines

Modify content


r : replace current character (**r**e replaces character r with e)

cw : change word (e.g **cw**new) changes current word to new)

#cw: used to change # words with new words.

c\$: change line

#c\$: change # of lines with new lines.



GENERAL FORM OF VI COMMANDS

(command)(number)(text object) e.g. **d2w**,
y3y

or the equivalent form:

(number)(command)(text object) e.g. **2dw**,
2yy

Here's how this works. ***number*** and ***command*** are optional. Without them, you simply have a **movement command**. If you add a *number*, you have a multiple movement. On the other hand, combine a *command* (**c**, **d**, or **y**) with a ***text object*** to get an editing command.

Other Commands

past commands

p paste left or below

P right or above

Undo Commands

- **u** undo last change

- **U** undo all changes in current line

- **:e!** undo all changes since last save (discard changes)

Search Command

- **/pattern** Search forward for *pattern*. End with ENTER

- **?pattern** Search backward for *pattern*. End with ENTER .

- To get the next position of the pattern type **n**

Vi macros: Using the map Command

To save yourself keystrokes, or the time that it takes to remember the sequence, you can assign the sequence to an unused key by using the map command.

:map x sequence

Define character x as a sequence of editing commands.

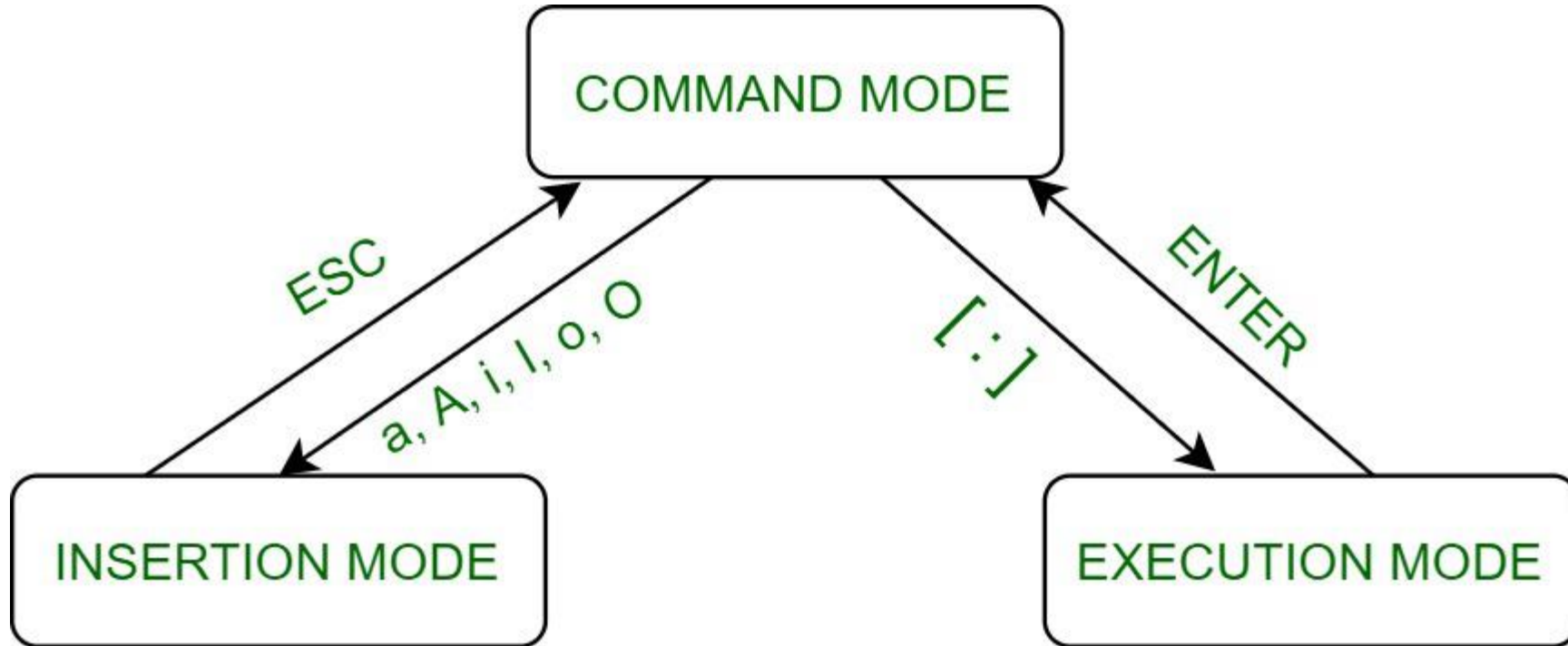
:unmap x

Disable the sequence defined for x.

:map

List the characters that are currently mapped.

Vi Modes



Vi macros

- Command mode map

```
:map S 1GddGp:wq
```

```
unmap S
```

- Insert mode map

```
:map! S ^[1GddGp:wq^M
```

Save command for permanent use in **.exrc** or **.vimrc**

