

# Editing with *vi*

Tejas Parikh ([t.parikh@northeastern.edu](mailto:t.parikh@northeastern.edu))

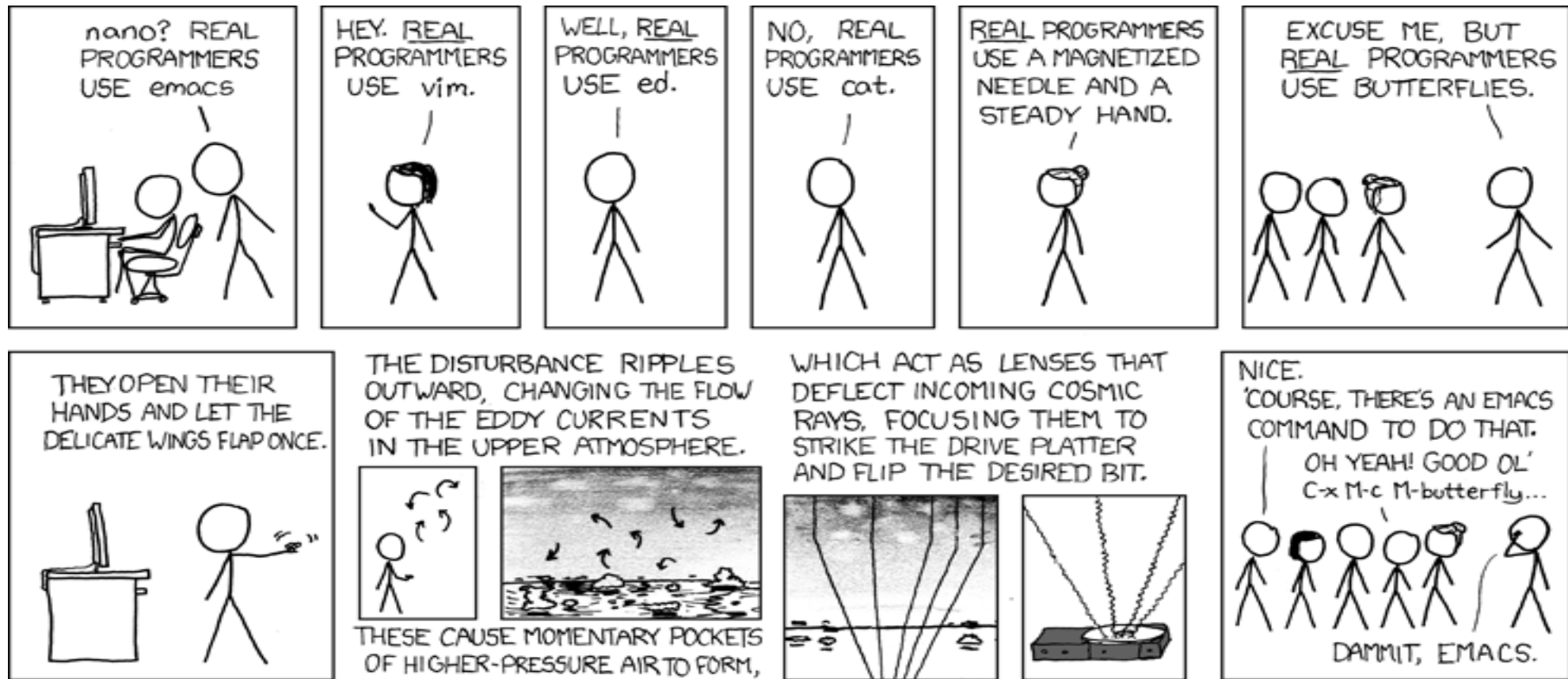
Spring 2018

CSYE 6225

Northeastern University

<https://spring2018.csye6225.com>

# The Great Text Editor War



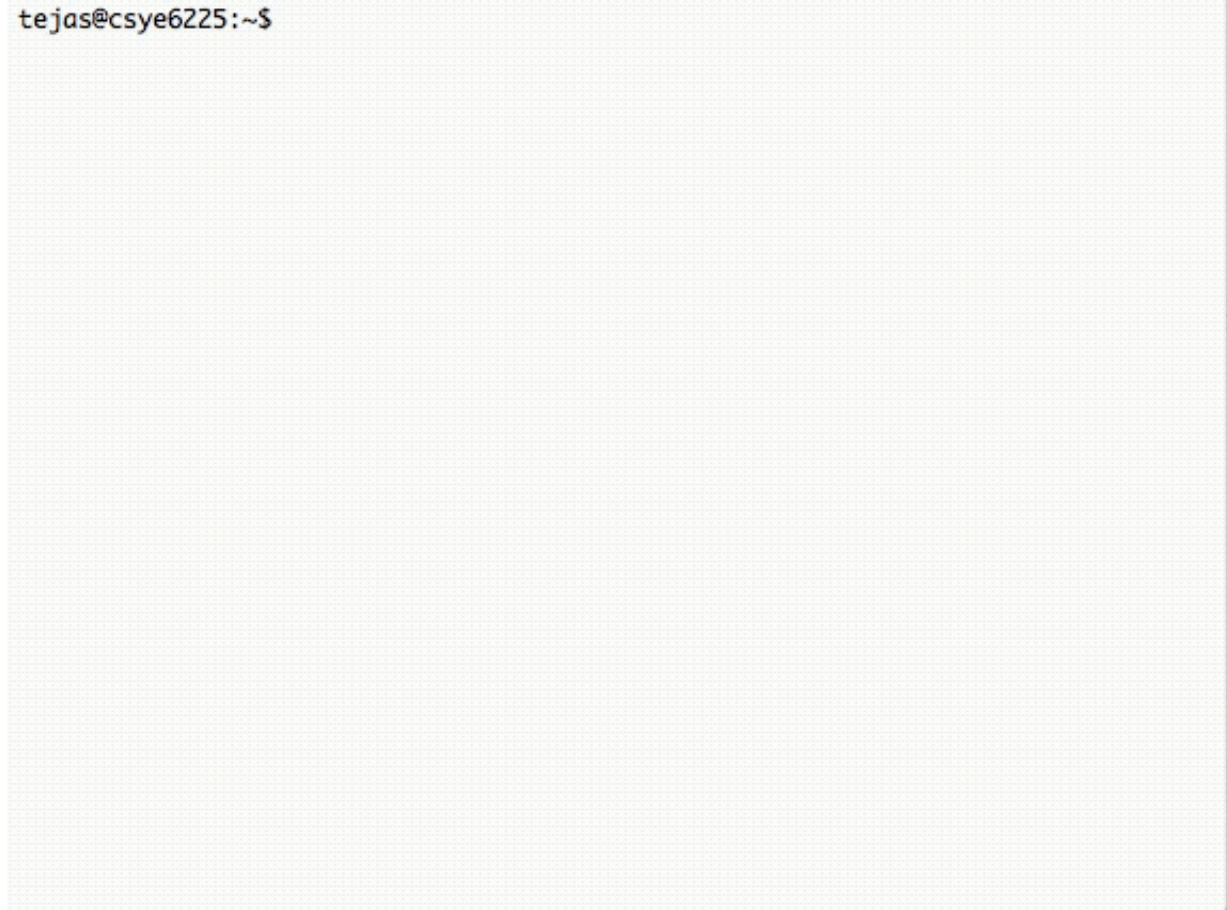
# Why We Should Learn vi

- vi is always available.
- vi is lightweight and fast.
- We don't want other Linux and Unix users to think we are sissies.

# A Little Background

- First version written by Bill Joy in 1976, a University of California at Berkley student who later went on to co-found Sun Microsystems.
- Most Linux distributions don't include real vi; rather, they ship with an enhanced replacement called vim (which is short for “vi improved”) written by Bram Moolenaar.
- vim is a substantial improvement over traditional Unix vi and is usually symbolically linked (or aliased) to the name “vi” on Linux systems.
- When we refer to “vi”, we will be actually referring to vim.”

# Starting and Stopping vi

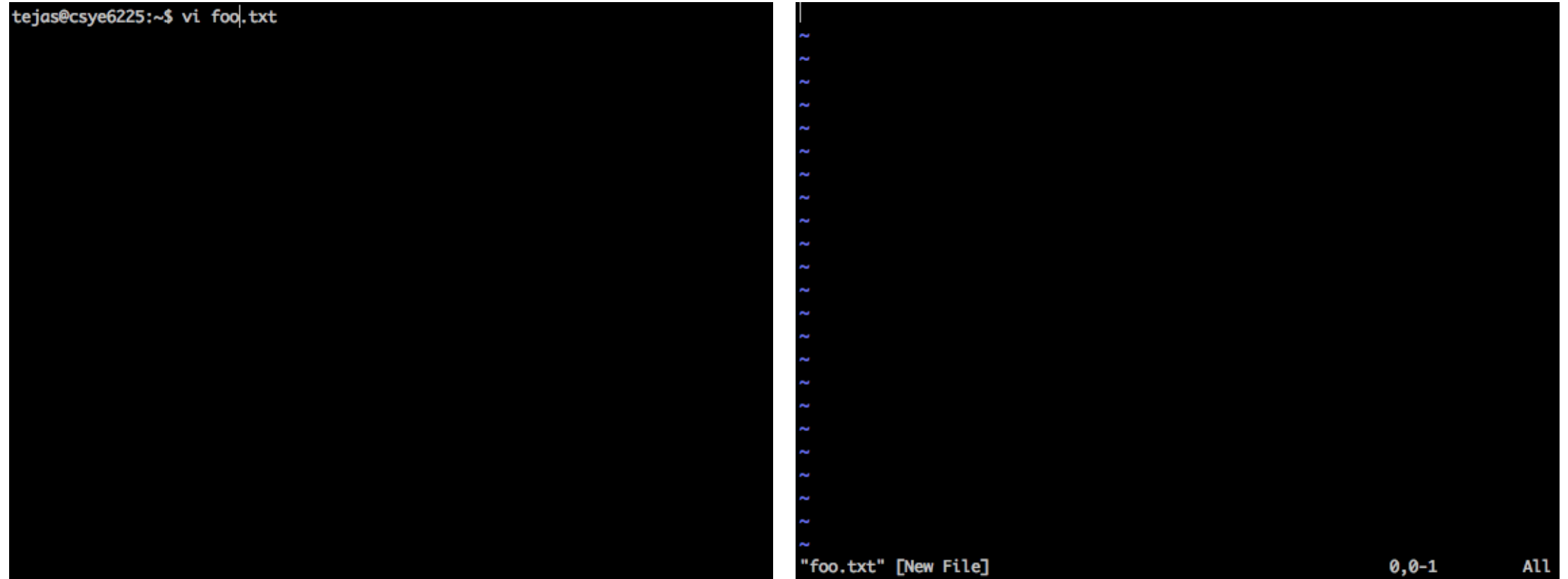


```
tejas@csye6225:~$
```

# Running in Vi compatible mode

- “Running in Vi compatible mode” means that vim will run in a mode that is closer to the normal behavior of vi rather than the enhanced behavior of vim.
- To run vim with its enhanced behavior, you can do one of the following:
  - Try running *vim* instead of *vi*.
  - Add **alias vi='vim'** to your *.bashrc* file.
  - Add **set nocp** to your *.vimrc* configuration file.
- Different Linux distributions package vim in different ways. Some distributions install a minimal version of vim by default that only supports a limited set of vim features.

# Editing Modes



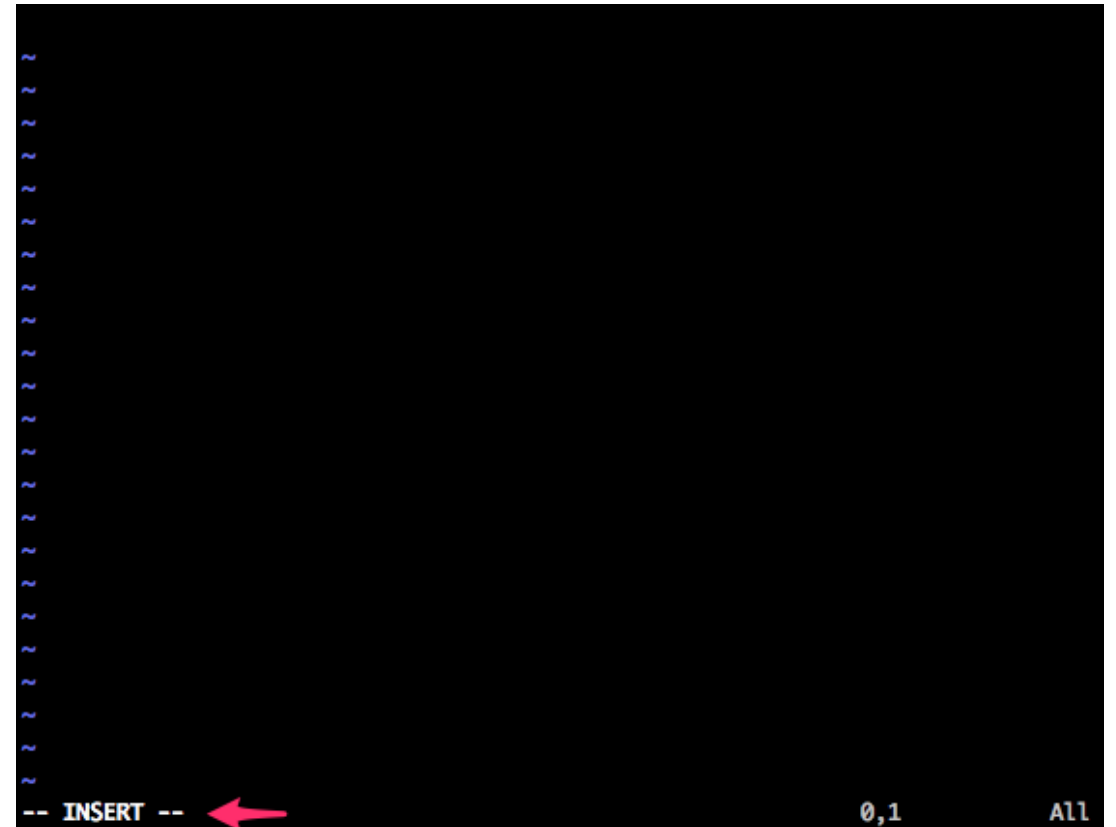
The image displays two side-by-side terminal windows. The left window shows a terminal prompt where the command `vi foo.txt` has been entered. The right window shows the vi editor's initial state: a blank file named `foo.txt` in `[New File]` mode. The left margin of the editor contains a vertical column of tilde (`~`) characters, indicating the start of each line. The bottom right corner of the editor shows the cursor position `0,0-1` and the word `All`.

```
tejas@csye6225:~$ vi foo.txt
```

```
"foo.txt" [New File] 0,0-1 All
```

# Entering & Exiting Insert Mode


- In order to add some text to our file, we must first enter *insert mode*. To do this, we press the “i” key.
- To exit insert mode and return to command mode, press the **Esc** key.





# Saving File Changes

- To save the change made to our file, we must enter command mode. This is easily done by pressing the “:” key. After doing this, a colon character should appear at the bottom of the screen.
- To write our modified file, we follow the colon with a “w” then Enter.



```
tejas@csye6225:~$
```

# Moving The Cursor Around

Key	Moves The Cursor
l or Right Arrow	Right one character.
h or Left Arrow	Left one character.
j or Down Arrow	Down one line.
k or Up Arrow	Up one line.
0 (zero)	To the beginning of the current line.
^	To the first non-whitespace character on the current line.
\$	To the end of the current line.
w	To the beginning of the next word or punctuation character.
W	To the beginning of the next word, ignoring punctuation characters.
b	To the beginning of the previous word or punctuation character.
B	To the beginning of the previous word, ignoring punctuation characters.
Ctrl-f or Page Down	Down one page.
Ctrl-b or Page Up	Up one page.
<i>number</i> G	To line <i>number</i> . For example, 1G moves to the first line of the file.
G	To the last line of the file.

# Appending Text

- Since we will almost always want to append text to the end of a line, vi offers a shortcut to move to the end of the current line and start appending. It's the “A” command. Let's try it and add some more lines to our file.
- First, we'll move the cursor to the beginning of the line using the “0” (zero) command.
- Now we type “A” and add the lines of text.
- Press the Esc key to exit insert mode. As we can see, the “A” command is more useful as it moves the cursor to the end of the line before starting insert mode.

# Deleting Text

- vi offers a variety of ways to delete text, all of which contain one of two keystrokes.

Command	Deletes
x	The current character.
3x	The current character and the next two characters.
dd	The current line.
5dd	The current line and the next four lines.
dW	From the current cursor position to the beginning of the next word.
d\$	From the current cursor location to the end of the current line.
d0	From the current cursor location to the beginning of the line.
d^	From the current cursor location to the first non-whitespace character in the line.
dG	From the current line to the end of the file.
d20G	From the current line to the twentieth line of the file.

# Cut, Copy & Paste

- The `d` command not only deletes text, it also “cuts” text. Each time we use the `d` command the deletion is copied into a paste buffer (think clipboard) that we can later recall with the `p` command to paste the contents of the buffer after the cursor or the `P` command to paste the contents before the cursor.
- The `y` command is used to “yank” (copy) text in much the same way the `d` command is used to cut text.

Command	Copies
<code>yy</code>	The current line.
<code>5yy</code>	The current line and the next four lines.
<code>yw</code>	From the current cursor position to the beginning of the next word.
<code>y\$</code>	From the current cursor location to the end of the current line.
<code>y0</code>	From the current cursor location to the beginning of the line.
<code>y^</code>	From the current cursor location to the first non-whitespace character in the line.
<code>yG</code>	From the current line to the end of the file.
<code>y20G</code>	From the current line to the twentieth line of the file.

# Search & Replace (Searching The Entire File)

- To move the cursor to the next occurrence of a word or phrase, the / command is used.
- When you type the / command a “/” will appear at the bottom of the screen. Next, type the word or phrase to be searched for, followed by the Enter key. The cursor will move to the next location containing the search string.
- A search may be repeated using the previous search string with the n command.

This graduate-level course will cover topics and technologies related to cloud computing and their practical implementations. The current industry trends have blurred the line between software developer, system administration, tester and many other roles. These days development teams are responsible from developing software to testing it to deploying it out to production and handling production issues at 2am. We will explore different models, techniques and architecture of cloud computing and prepare you to meet current market demands. You will gain hands on experience with various feature of popular cloud platforms such as Google Cloud Platform, Amazon Web Services, Microsoft Azure, etc. The lectures and assignments aim to help you build skills to develop large-scale applications using cloud platform and tools.

```
"foo.txt" 6L, 829C      1,1      All
```

# Global Search-And-Replace

**:%s/Line/line/g**

Item	Meaning
:	The colon character to enter command mode
%	Specifies the range of lines for the operation. % is a shortcut meaning from the first line to the last line.
s	Specifies the operation. In this case, substitution (search-and- replace)
/Line/line/	The search pattern and the replacement text.
g	This means “global” in the sense that the search-and-replace is performed on every instance of the search string in the line. If omitted, only the first instance of the search string on each line is replaced.

# Save a file (that requires root permissions) from within vim editor

- <https://www.cyberciti.biz/faq/vim-vi-text-editor-save-file-without-root-permission/>



# Additional Resources

<https://spring2018.csye6225.com/>