Frankauski & Kotula 1
Kristen Frankauski & Brooklyn Kotula
COSC 362

1 Text Editors: Emacs and Vi

1.1 What are Text Editors?

Text editors are a type of program used for editing plain text files. Text editors are programs on your computer that allows you to create and edit a range of programming language files. They are basically just a place to write code. A majority of text editors were originally created to be very general and to write anything with it. It was not until later that handy tools and functions for developing C/C++ and other programming languages were added.

1.2 What Text Editors we looked at:

When it comes to text editors, there are a few things that need to be taken into consideration when picking one to use. Finding one that is easy to use and navigate is important, especially for those who are not computer programmers and just computer users that need a text editor that is not a Microsoft product. Editors that have features such as replace/search, cut, copy, and paste are very beneficial and can make working through code or a document easier. Another feature, syntax highlighting, makes it much easier to read code and pick up errors made along the way. Text editor users also look for a customizable appearance where they are able to modify things such as font size and color schemes within the editor to optimize their work zone.

1.3 Emacs:

Emacs is short for Editor Macros. Emacs is a visual editor which is a representation of an entire document where the user has the ability to move around freely and edit any part of the document. This text editor uses keyboard commands, making it much faster to use than mouse commands. The use of the Ctrl and Esc characters distinguish editor commands from text to be inserted into the buffer being used. A buffer is a copy of a file that Emacs creates and stores the copy in part of RAM memory. So when a user is editing using Emacs, they are not actually editing the file itself, but they are editing the buffer. Once the buffer has been saved, Emacs writes the contents of the buffer to the original file. It is important to save work often to flush the buffer to the disk so that data is not lost if the user loses network connection or there is a power outage while editing.

1.4 Vi:

Vi is built into any UNIX/Linux installation like nano is. It is very similar to nano but gives you a full 24 line view. Vi is also a full screen text editor where you can edit several lines at once. It also has two modes, command and insert. You have to switch back and forth between each mode depending on what you want to do. Command mode is for doing saving and quitting while insert mode is to insert your text. You can also do commands such as delete word in insert mode.

1.5 Similarities and Difference between Emacs and Vi:

Most text editors are basically looking for the same output in the end, and that is to be able to edit some type of text document. Emacs and Vi have some similarities and it seems many more differences in the way that they operate. One similarity is the command format that is used to create or edit an existing file is the same. It uses the name of the editor and then the filename. An example of this is: emacs myFile.tex or vi myFile.tex, which this format is used if myFile.tex is already an existing file being edited or if myFile.tex is a new file being created. Another similarity is that they both will basically run on all operating systems that a user may come across. As for differences, Vi has historically been known to run faster than Emacs starting up in less time and is generally more responsive. Vi also takes up much less memory than Emacs and is already built into a linux system while Emacs requires being installed. Another difference more on the user operating side is that in Emacs the user is easily able to cut, copy, and paste while in Vi that is not command. Lastly, Emacs is extensively customizable by letting users choose from a variety of macros to integrate workflow while Vi relies on its simplistic and straightforward process. A macro records a sequence of user actions that carry out Emacs commands. Although they are both text editors and were created essentially to do the same thing, they are significantly different.

1.6 Basic Commands (Demonstration of commands during presentation):

Emacs:

1. *Emacs* *checks to see if emacs is already downloaded and running
2. *Emacs <filename>* : used to create or edit a file
3. While working in the new file
   (a) Ctrl-x, Ctrl-c : quit
   (b) Press “y” to save

   (c) Press "n" to not save

4. Ctrl-space : used for marking text to do actions such as copy, cut, and paste

   (a) Esc-w : copy
   (b) Ctrl-w : cut
   (c) Ctrl-y : "yank" or paste

5. Alt-d : delete an entire word

6. Ctrl-s : search for words

7. Alt-shift-% : replace a word

   (a) Type word you'd like to replace and press enter
   (b) Type the replacing word and press enter

      i. Press y to replace the current match
      ii. Press n to skip to the next match
      iii. Press q to exit without any replacements
      iv. Press ! to do a global replacement (emacs will show the replaced occurrences)

Vi:

1. To delete words or sentences you are able to use the **'delete'** or **'backspace'** button on your keyboard. Just remember this will delete EVERYTHING that is in the sentence. It can not be used to delete just one word such as the first word in a sentence.

   (a) To delete just one word, or multiple words after you use the command **dw**.
   (b) To delete multiple words you can use the **number of words + dw** (Ex. 3dw). This command must be placed on the first letter of the word/s that you want to delete.

2. To go into insert mode use the letter **i**.

3. To go into command mode use the letter **c**.

4. To save and leave vi running use the command **<Esc>, :w, <return>**.

5. To save and exit use the command **<Esc>, zz**.

1.7 Strengths and Weaknesses of Vi and Emacs

Emacs:
There are many perks to using Emacs as a text editor, one reason being that it allows the user to have several files open at once, and the user has the ability to switch among them. Although mentioned before that saving is very important in Emacs incase of a power outage, Emacs actually writes checkpoint files to help avoid losing work. Emacs also has modes for several programming languages such as C/C++, Python, java, shell script, and html to name a few. The advantage to these modes is that when a user types code, it will automatically indent it for them. Emacs also uses keyboard commands which are much faster than mouse commands. There are multiple clipboards that are used when copying in Emacs which means the user does not lose the text that was formally copied. Instead, the formally copied text is pushed into a "kill ring" where it can be accessed later. Lastly, Emacs is able to bind any command to any key, basically meaning that any or all commands could essentially be changed. The main weakness that I had found with Emacs is that there are 15 keyboard commands that need to be learned to be used effectively and 30 keyboard commands to be really proficient. Generally though, people who take the time to learn Emacs tend to be dissatisfied with other text editors.

Vi:
Vi is extremely convenient for one big reason. Vi is already built into most, if not all Linux systems. This makes it a quick way to get into a text editor instead of having to install one. Another strength would be the ability to see 24 lines and that it is full screen. You can make it into a different window in the terminal and essentially have one screen for the editor to be open and one for the terminal. The weakness in this editor would be the limited ability that it has. You can not format in this editor which makes it very inconvenient if you need formatting. Some commands also vary per system such as the delete/ backspace command so you may spend some time having to learn which keyboard commands work for your system. The last weakness would be that you can not see some of the commands that you do on the screen. This editor may not be good for new users.

1.8 What Text Editor do we prefer?

Kristen: I personally would prefer to use emacs. It is easier to know the commands especially for someone who isnt as familiar with the system and more familiar with GUI based editors.
Brooklyn: I personally would prefer to use emacs as well. I have a bit of experience with both now, and I found emacs much easier to manipulate. I really enjoyed digging into a few different text editors.

1.9 Improvements needed

Emacs:
Emacs needs to try to find a way to have a quicker startup time and take up less memory. Another improvement to Emacs is to make the user interface less esoteric and make it more similar and familiar to computer users. The reason most users do not use Emacs is because of its user interface. Although this may be an improvement, I personally think Emacs esoteric user interface is what makes it so unique and so effective to the users who take the time to learn it.

Vi:
Vi needs to have the ability to cut/delete using the cursor for a word instead of having to use dw. Having so many lettered commands like dw can also get confusing and difficult for users since there are so many different lettered commands. If some of these lettered commands were to be turned into cursor or keyboard commands it may be easier for the user. The last improvement I would make is the ability to see the commands that you are entering.