Text Editors and Tex Processing

1. **Describe the general structure of vi and describe the vi tradeoff.**

   The vi editor is defined by its use of the same characters for commands as well as text insertion. Therefore, vi requires the concept of "mode" that the user must be aware of from moment to moment. It is assumed that the user can easily switch between "command" and "insert" modes, but even the experienced user always slips up and forgets the current mode. This results in the insertion of commands as text into the working document. Or worse, typing text in command mode leads to the execution of "random" commands as various letters of a word are quickly entered. The random commands delete or duplicate letters, words, and lines of previously entered text.

   The vi editor is also "line oriented." Commands are character or line related, but the characters that delimit lines cannot be directly manipulated. For example, there is no global find and replace command to replace the end-of-line character with a space character. (The idea of a global "join command" but only for non-empty lines.)

   To put it simply, the vi text editor is clumsy and error prone. On the other hand, vi is the common denominator of Unix-based systems. Thus, knowing how to use vi will always let you get around and wiggle out of tight spots where there is no alternative editor.

2. **Describe the history and general structure of the emacs editor and contrast the design philosophy of vi *versus* emacs.**

   The emacs editor was inspired by a "rebel" text editor and corrector in the 1970's called TECO. The TECO command line editor viewed text as a simple collection of characters that were manipulated with separate command characters. Much of TECO's strength came from its ability to do macro processing and flow control iteration based upon the value of the strings discovered in the text buffer. In fact, more that half of the TECO commands were built from TECO macros.

   Following and improving upon this design, emacs employs a series of "buffers" that hold text strings. Characters can be typed into and moved among the buffers. Text commands use the Ctrl and Alt keys to keep commands separated from entered text.

   Unlike the line orientation in vi, emacs is character oriented. It can search for and replace any character sequence. Inspired by TECO, emacs employs a powerful macro processing subsystem from which much of emacs functionality arises.

   The power of emacs is also its weakness. The commands are numerous and obscure in spite of extensive on-line documentation. Virtually any character can be "bound" to a given function. Many commands span multiple control character sequences, and simple text replacement requires macro invocation. Tracking and moving among many buffers in not obvious. And the default control character sequences are often trapped by a remote OS and not sent over a network connection to emacs.

3. **Explain the difference between an ASCII, PostScript, dvi, and PCL files and the design tradeoff.**

   These terms relate to printer file formats. ASCII formatted files contain codes for each character. The ASCII code is considered "universal" since almost all display and printing hardware is able to accept ASCII codes.

   PostScript was invented by Adobe Systems. PostScript is a superset of ASCII (PostScript files consist of ASCII codes) that implements a Page Description Language (PDL). Put another way, PostScript files contain two types of information. One type is the "copy" or primary document information, and the second type is a list of instructions on how to display the copy.

   The Device Independent (dvi) format was developed for Unix and, like PostScript, implements a page description language. The Unix document formatters TeX and LaTeX generate dvi formatted files.

   PCL is a graphic file format for HP LaserJet printers. PCL describes the printed page pixel by pixel, 300 pixels per inch.

   Although universal and compact, ASCII files only provide the "copy" with minimal formatting. PCL is at the other extreme. It is specific to HP printers and creates very large files. In between ASCII and PCL is PostScript and dvi, where they offer a compromise between description and size by just offering, if possible, the manner in which to display the document. Probably more than half of the laser printers sold today are able to interpret PostScript files.

4. **Explain the difference between nroff, groff, troff, TeX, and LaTeX.**

   Just as emacs was inspired by TECO, nroff and groff were inspired by "roff" (runoff) for the old DEC 10 Operating System. Roff, in turn, was developed on the PDP-8 for driving automated typesetters in the newsprint industry.

   In Unix, nroff is the basic ASCII text formatter. Unix also employs troff to create tables with formatted text in the table cells. In Linux, nroff calls the GNU groff ASCII text formatter which implements nroff and troff functionality.

   TeX and LaTeX are Page Description Language (PDL) generators that detail layout specifications of the document as well as the document content. TeX deals with low-level formating issues such as type face, kerning, leading, lines, and embedded graphics. LaTeX is a macro package that provides document style guides such as type of document paging, margins, columns, paragraphs, and blocking boarders.

5. **Describe the interaction between Linux formats and formatters.**

   Generally formatters and formats are arranged so that the PDL formatters (TeX and LaTex) use dvi and PostScript formats while an ASCII formatter (groff) uses just an ASCII format.

   The complication arises when the user switches output devices or instructs an ASCII formatter to direct its output to a PostScript format. Here is a selection of commands that switch the various document formats:

dvips (1)	- convert a TeX DVI file to PostScript
dvilj4 (1)	- convert dvi files to PCL, for HP LaserJet printers

dvired (1)	- print dvi-files
dvitype (1)	- translate a dvi file for humans
grops (1)	- PostScript driver for groff
grodvi (1)	- convert groff output to TeX dvi format
sgml2latex (1)	- create LaTeX, DVI or PostScript output from a SGML source file
mpage (1)	- print multiple ASCII pages per sheet on PostScript printer
nenscript (1)	- format an ASCII file and convert to PostScript
pbmtolps (1)	- convert portable bitmap to PostScript
pdftops (1)	- Portable Document Format (PDF) to PostScript converter
gv (1x)	- a PostScript and PDF previewer
xdvi (1)	- DVI Previewer for the X Window System

6. **Contrast the commands lpr file, lpr -h -Plj4 file, export PRINTER=lj4.**
7. **Contrast the commands lpq, lpq -Plj4 file, lprm 011, lprm -, lprm lj4, and lprm.**
8. **Contrast the commands man lpr | col -b | lpr -h -Plj4, man mpage | colcrt | mpage | lpr -h, man mpage | col -b | mpage -2 | lpr -h, groff -man -Tps /usr/man/man1/lpr.1 | lpr.**
9. **Contrast BSD and System V printer sub-systems, explain the confusion over lp.**
10. **Explain why the same physical printer might be given several logical names in the /etc/printcap file.**