Editing with `vi`

`vi` is ubiquitous: you will find it even in the smallest distributions. It is much easier to use than its original forms, `ed` and `ex`.

Modes

`vi` has two main modes: move and insert. A variety of commands in `vi` will put you into insert mode, but `ESC` is the way to get out!

Spell checking

As mentioned in on page 168 of your text, `spell` (or `ispell`, an interactive version) are usually used outside of `vi`.

Summary of `vi` commands

A summary of `vi` commands is here.

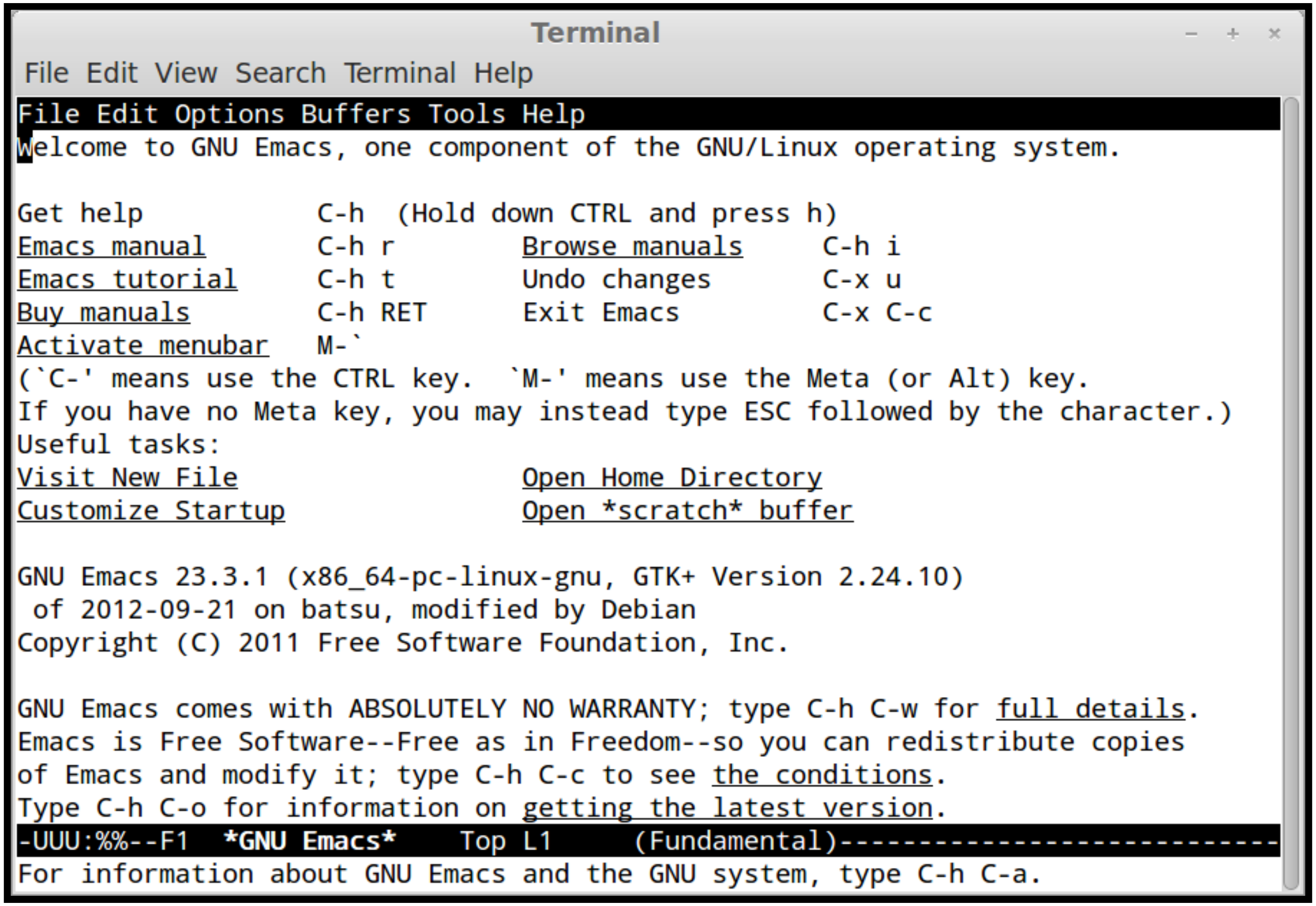
Editing with `emacs`

`emacs` is a far more ambitious program than `vi`.

`emacs` is derived from `teco`, an even more cryptic editor than `ed`; the history of `emacs` is far more detailed than that of ed.

emacs in action

emacs has a distinctive look:



```
File Edit Options Buffers Tools Help
Welcome to GNU Emacs, one component of the GNU/Linux operating system.

Get help           C-h  (Hold down CTRL and press h)
Emacs manual       C-h r        Browse manuals     C-h i
Emacs tutorial     C-h t        Undo changes       C-x u
Buy manuals        C-h RET      Exit Emacs         C-x C-c
Activate menubar   M-`
(`C-' means use the CTRL key.  `M-' means use the Meta (or Alt) key.
If you have no Meta key, you may instead type ESC followed by the character.)
Useful tasks:
Visit New File                  Open Home Directory
Customize Startup               Open *scratch* buffer

GNU Emacs 23.3.1 (x86_64-pc-linux-gnu, GTK+ Version 2.24.10)
 of 2012-09-21 on batsu, modified by Debian
Copyright (C) 2011 Free Software Foundation, Inc.

GNU Emacs comes with ABSOLUTELY NO WARRANTY; type C-h C-w for full details.
Emacs is Free Software--Free as in Freedom--so you can redistribute copies
of Emacs and modify it; type C-h C-c to see the conditions.
Type C-h C-o for information on getting the latest version.
-UUU:%%--F1  *GNU Emacs*    Top L1     (Fundamental)----------------------------
For information about GNU Emacs and the GNU system, type C-h C-a.
```

emacs summary

A summary of emacs commands is here.

`emacs` and ispell

Unlike `vi`, it is very easy and convenient to run `ispell` interactively within `emacs`.

Editing, files, and exiting `vi`

Keystroke(s)	Meaning
`ZZ`	Save and quit
`Q`	Quit `vi` and go into `ex` (bad idea!)
`:w`	Write file
`:w!`	Force write, even if "protected"
`:w somefile`	Write to a new file `somefile`
`:q`	Quit
`:q!`	Quit, losing all changes
`:e somefile`	Start editing a new file called `somefile`
`:e!`	Start editing over, forgetting all changes since last save
`:n`	Go to the next file (such as one named on the command line)

Movement

Keystroke(s)	Meaning
`h, j, k, l`	The big four: left, down, up, right (←, ↓, ↑, →)
`w, W, b, B`	Forward one word, back one word
`e, E`	end of current word
`(, )`	Beginning of previous sentence; end of next sentence (doesn't always work correctly!)
`{, }`	Beginning of previous paragraph; end of next pararaph
`0, ^`	First position, first character

$	End of line
+, -	First character of next or previous line
H, M, L	Top line of screen, middle line, last line
^F, ^B	Forward one screen, back one screen
^D, ^U	Down half a screen, up half a screen
z RET	Position current line at top of screen
/*something*	Search forward for *something*
?*something*	Reverse search for *something*
/ RET	Repeat last search forward
? RET	Position current line in middle of screen
^G	Reports status line (including the current line's number)
^G	Reports status line (including the current line's number)
G *n* RET, :*n* RET	Go to absolute line *n*
G	Go to last line
m*x*	Mark current as *x*
'*x*	Go to mark *x*

Editing

Keystroke(s)	Meaning
i	Insert text
a	Append text (very useful at the end of line!)
o, O	"Open" a new line immediately below current, "open" a new line immediately above current
r	Change one character
cw	Change a word from the current point

cc	Reinitialize the current line
C	Change from current point to end of line
R	Overwrite mode
x	Delete the current character
dw	Delete word
dd	Delete current line
D	Delete from current character to end of line
p	Put back at current
P	Put back at previous (much more useful!)
"*n*p	Go back in delete history *n* items for P
yw	Copy word (use p,P to retrieve)
yy	Copy line (use p,P to retrieve)
"*ayy*	Copy line into buffer *a*
"*aP*	Paste line from buffer *a*
:%s/*something*/*otherthing*/g	Replace *something* with *otherthing* throughout the current file
u, U	Undo; restore line (very useful!)

Note: in `emacs` terminology, "META" may refer to either "ESC" (which I tend to use), or some combination of ALT/CTRL/??? keys — if you do most of your typing on one keyboard, it may be worth figuring out which combination works for "META".

Editing, files, and exiting `emacs`

Default Keystroke(s)	Meaning
`^X^C`	Save and quit
`^X^S, ^Xs`	Write file, write all files
`^X^W somefile`	Write to a new file `somefile`
`^X^F somefile, ^X^V`	Start editing a new file called `somefile`
`ESC X revert-buffer`	Start editing over, forgetting all changes since last save

Manipulating `emacs` windows

Default Keystroke(s)	Meaning
`^X2, ^X3`	Split windows
`^Xo`	Switch windows
`^X^`	Make a window larger vertically
`^Xb`	Switch the current window's buffer

Movement

Default Keystroke(s)	Meaning
^F, ^B, ^N, ^P	The big four: left, down, up, right (←, ↓, ↑, →)
ESC F, ESC B	Forward one word, back one word
^A, ^E	Beginning of current line, end of current line
^V, ESC V	Forward one screen, back one screen
^S*something*	Incremental search forward for *something*
ESC x isearch-forward-regexp*regexp*	Incrementally search forward for *regexp* (very powerful!)
^R*something*	Incrementally reverse search for *something*
ESC >	Go to last line
ESC <	Go to first line
^SPACE	Set mark (start creating a region)
^X^X	Go to mark

Macros

Default Keystroke(s)	Meaning
^X(	Start learning a macro
^X)	End learning a macro
^Xe	Execute a macro

Editing

Default Keystroke(s)	Meaning
`^K`	"Kill" from this point up to the end of line (saved in "kill" ring)
`^O`	"Open" a new line immediately below current, "open" a new line immediately above current
`^T`	"Transpose" the current character and the next one
`^W`	Cut the current region (saved in "kill" ring)
`ESC W`	Copy the current region (saved in "kill" ring)
`^Y`	Paste from latest item in "kill" ring
`ESC x query-replace` *something* `RET` *otherthing*	Replace *something* with *otherthing*
`ESC x query-replace-regexp` *regexp* `RET` *otherthing*	Replace *regexp* with *otherthing* (powerful!)
`^Y`	Paste from latest item in "kill" ring
`ESC x undo`	Undo