CHAPTER 4

Editors Under UNIX

4.1 VI EDITOR

The vi editor is a popular screen editor under UNIX editor since 1970's which is based on an earlier editor known as ***elvis*** [Richard L Peterson]. The editor has three modes known as: Input Mode, Command Mode, and Colon Mode. In the **Input Mode,** whatever user enters will be written into the document. **Command Mode** allows the user to enter commands. We can go to this mode from the Input mode by pressing the ESC key and hence it is also referred to as ESC mode. If the ESC key is pressed while in the command mode, one will get a beep sound. In the third mode, called the **colon mode**, users can run commands and also do some document editing. Thus it is not considered a separate mode.

In a nutshell, the following is the summary of useful commands to immediately work under UNIX :

1. **vi *filename*** — opens the vi editor to work with the given filename.
2. Initially, a screen will be opened with the command mode.
3. To enter text, press **i**. The input mode will be displayed at bottom right part of the screen.
4. On pressing **Esc** key, the command mode reemerges. One could press
   :w to save the matter and resume editing.
   :wq to save the matter and quit the vi editor.
   :q! to quit the editor without saving.
5. The three modes present in vi editor are: i) Command mode ii) input mode iii) Colon mode
6. In command mode, commands can be entered.
   press **i** to insert text before the current cursor position.
   press **I** to insert text at the beginning of the line.
   press **a** to insert text after the cursor position.
   press **A** to insert text at the end of the current line.
   press **o** to open a new line below the current line.
   press **O** to open a new line above the current line.

press **r** to replace the present character with a character.
press **R** to replace a group of characters from current cursor position.
press **x** to delete present character.
press **J** to join the next line to the end of the current line.
press **dd** to delete the current line.
press **4dd** to delete 4 lines from the current line.
press **dw** to delete the current word.
press **7dw** to delete 7 words from the current word onwards.
press **30i*Esc** (invisible command) to insert 30 *'s at the cursor position.
press **u** to undo the effect of the previous command on the document.
press **.** to repeat the previous command.
press **yy** to copy the entire line in to the buffer.
press **yw** to copy the entire word in to buffer.
press **p** to place the copied or deleted information below the cursor.
press **P** to place the copied or deleted information above the cursor.

7. **Colon mode commands**

Search and substitute commands

:/raja searches for the string "raja" in the forward direction. Press **n** to repeat the search.

:?raja search for the string in the backward direction. Press **n** to repeat the above search.

:s/raja/rama replaces the first occurrence of "raja" with "rama".

:s/raja/rama/g replaces all "raja"'s with rama in the present line.

:g/raja/s/raja/rama/g replaces all "raja"'s by "rama" in the entire file.

Block delete commands

:1d delete the line 1.

:1,5d deletes the lines from 1 to 5.

$ Means last line of the file.

. Means present line (i.e.) present line.

:10,$d deletes lines from 10th line to the last line of the file.

:1,$d deletes lines from 1 to last line of the file.

:.,$d deletes lines from present line to the last line.

:.-3,.d deletes the lines from present line and above 2 lines
(deletes 3 lines including the cursor line).

:.,.+4d deletes the lines from the present cursor line followed 3lines(total 4 lines).

:-1,. +3d deletes the lines one above the cursor line followed by it 3 lines.

:18 cursor goes to 18 th line of the file.

Block copy commands

:1,5 co 10 copies the lines from 1 to 5 after 10th line

:1,$ co $ copies the lines from 1 to last line after last line

:.,.+5 co 8 copies lines from present line to 5 lines after 8th line

:-3,. co 10 copies the lines from present cursor line and above 3 lines after 10 th line.

Block moving commands

:1,5 mo 9 moves lines from 1 to 5 after 9th line.

:1,$ mo $ moves lines from 1 to $ after last line.

:.,.+5 mo 10 moves lines from present line and next 5 lines after 10 th line onwards.

:.-3,. mo 10 moves present line and above 3 lines after 10 th line.

Importing & **Exporting the files**

:1,5 w filename writes lines 1 to 5 in the specified filename.

:1,5 w! **filename** overwrites lines 1 to 5 in the specified filename.

:r filename Adds the content of filename after the current line.

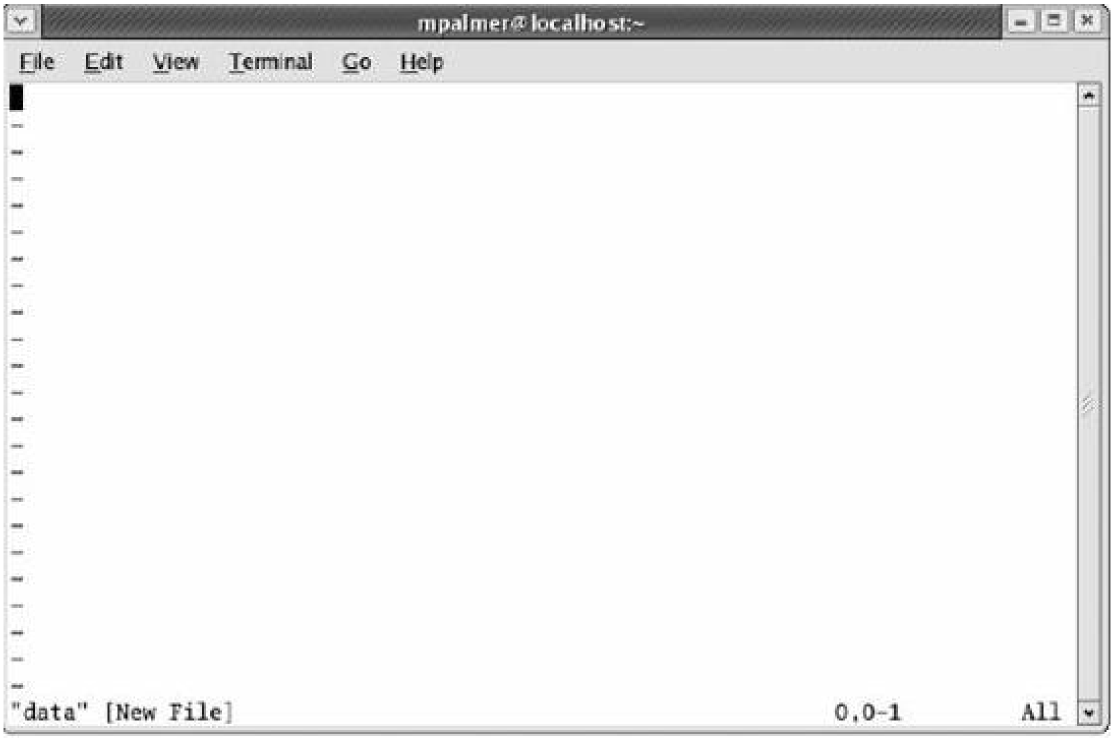
8. **Book mark command**

   Bookmarks (markers) are not visible and are useful to jump from one line to another quickly. Markers should be in lower case. To have the marker on a specified line press **m** followed by a lower case alphabet (say a) then marker for that line is set as a. To go to the marked line press **'a** ('back quote)followed a. e.g.: go to 500 th line, press mb (b is the marker). To go to the 500 th line from anywhere in the document press **'b**. Then the cursor goes to the 500 th line.

View Command

View is a readonly version of vi command. That is equivalent to vi –R filename. With the help of this command, we can view the content of a file. We can exit by type :q.

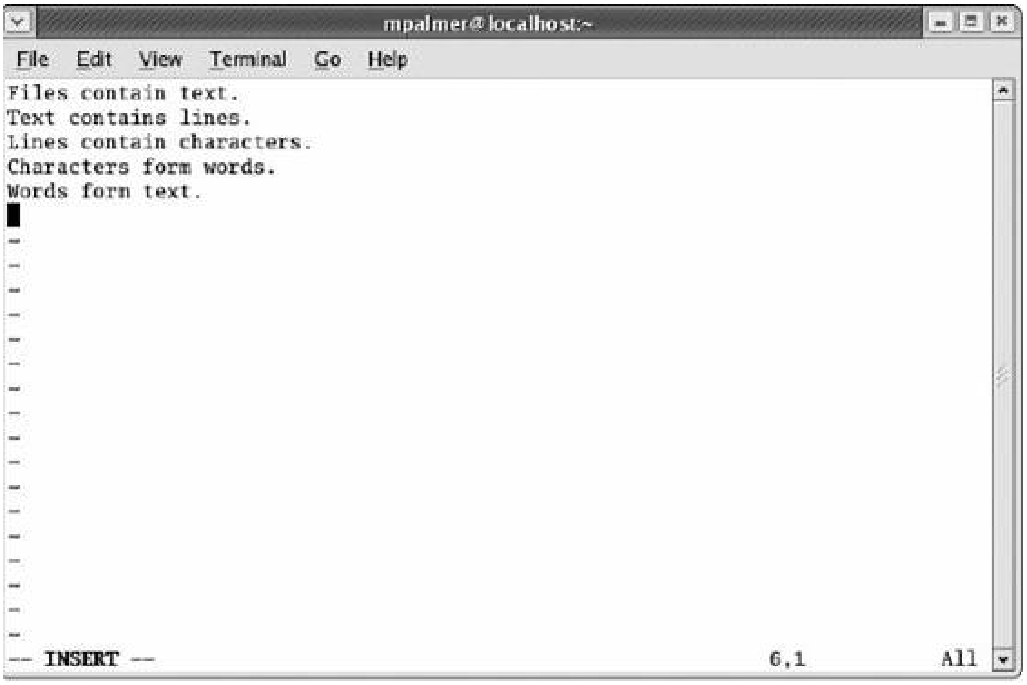
For example, **vi data** command gives vi screen as shown in Figure 4.1. Those lines which starts with ~ indicates that they are lines yet to be opened.



```
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"data" [New File]                                    0,0-1          All
```

Figure 4.1. First screen of vi editor.

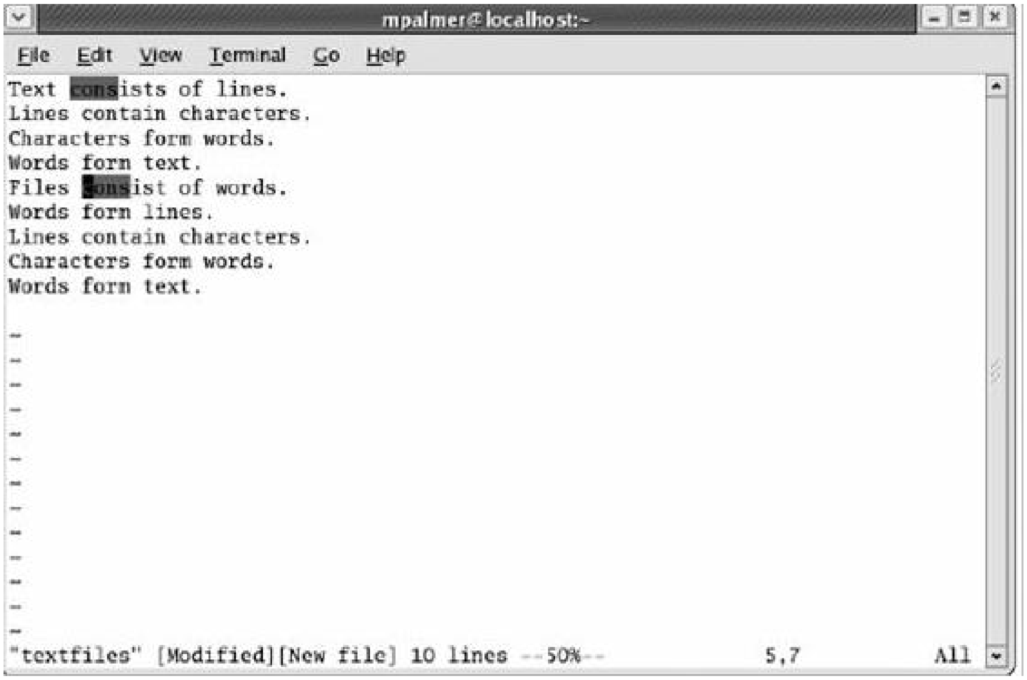
Once, we are in the vi editor we can press keys such as i,I,a,A etc., to go to input mode (see Figure 4.2) in which what ever we type will be saved in the document.



```
Files contain text.
Text contains lines.
Lines contain characters.
Characters form words.
Words forn text.
█
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
-- INSERT --                                              6,1          All
```

Figure 4.2. Insert mode of vi editor

If the file is already existing, then it displays its status as shown in Figure 4.3.

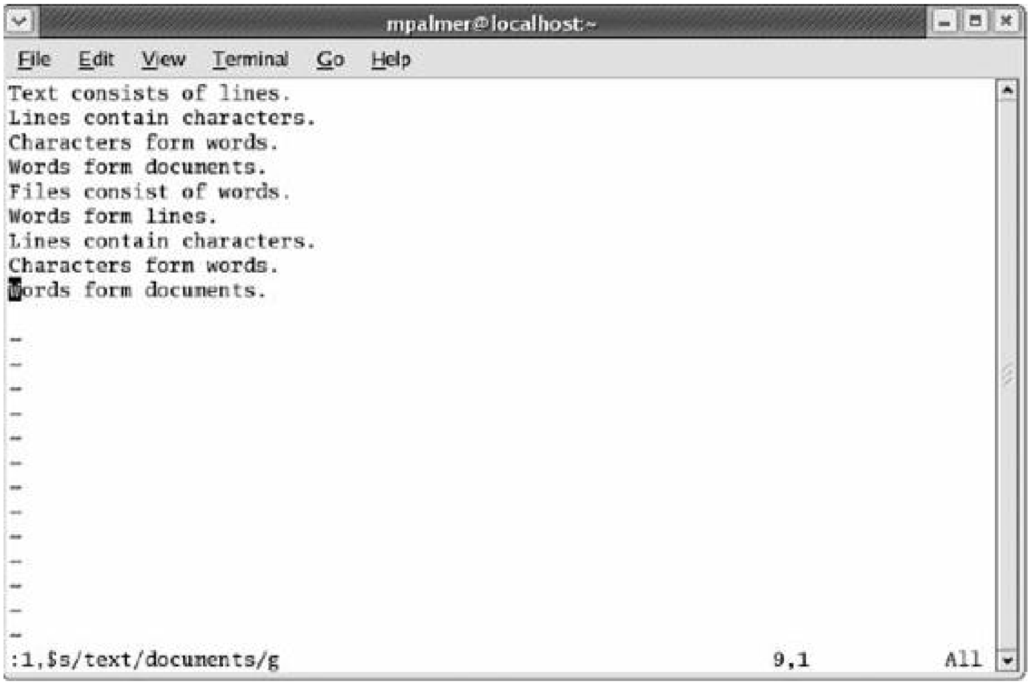


```
Text consists of lines.
Lines contain characters.
Characters form words.
Words forn text.
Files consist of words.
Words forn lines.
Lines contain characters.
Characters form words.
Words forn text.

~
~
~
~
~
~
~
~
~
~
~
~
~
~
"textfiles" [Modified][New file] 10 lines --50%--         5,7          All
```

Figure 4.3. vi screen after opening an existing file.

Figure 4.4 demonstrates the searching and replacing.

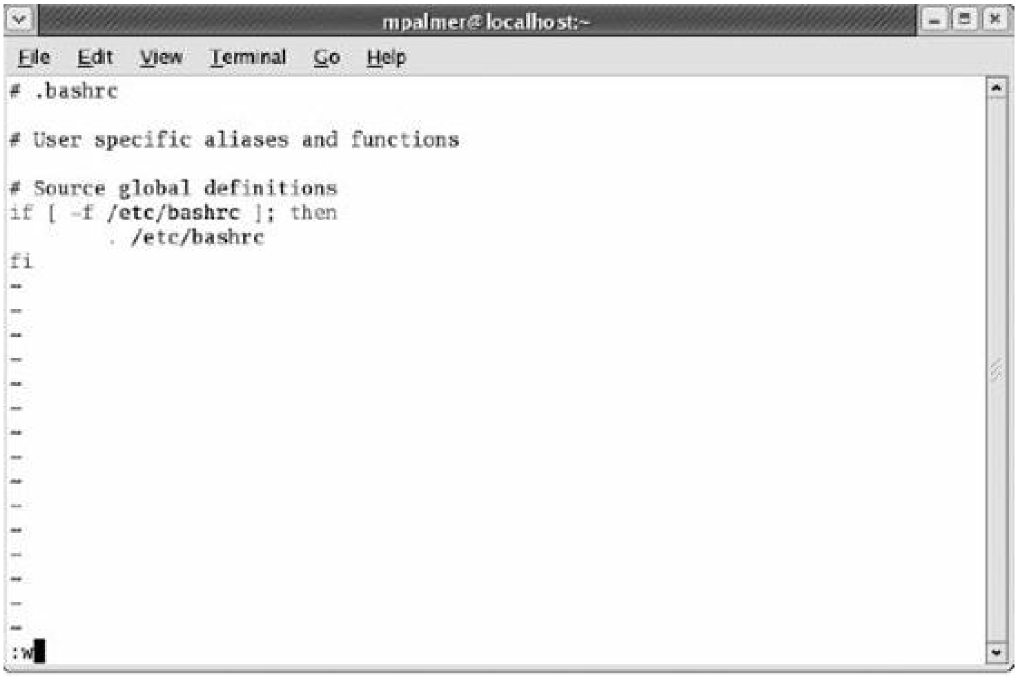


```
Text consists of lines.
Lines contain characters.
Characters form words.
Words form documents.
Files consist of words.
Words form lines.
Lines contain characters.
Characters form words.
Words form documents.

~
~
~
~
~
~
~
~
~
~
~
~
~
:1,$s/text/documents/g                                 9,1          All
```

Figure 4.4. Searching replacing globally.

While editing a file, we may wanted to save intermittently. The same can be achieved by executing :w command as shown in Figure 4.5 and explained earlier.



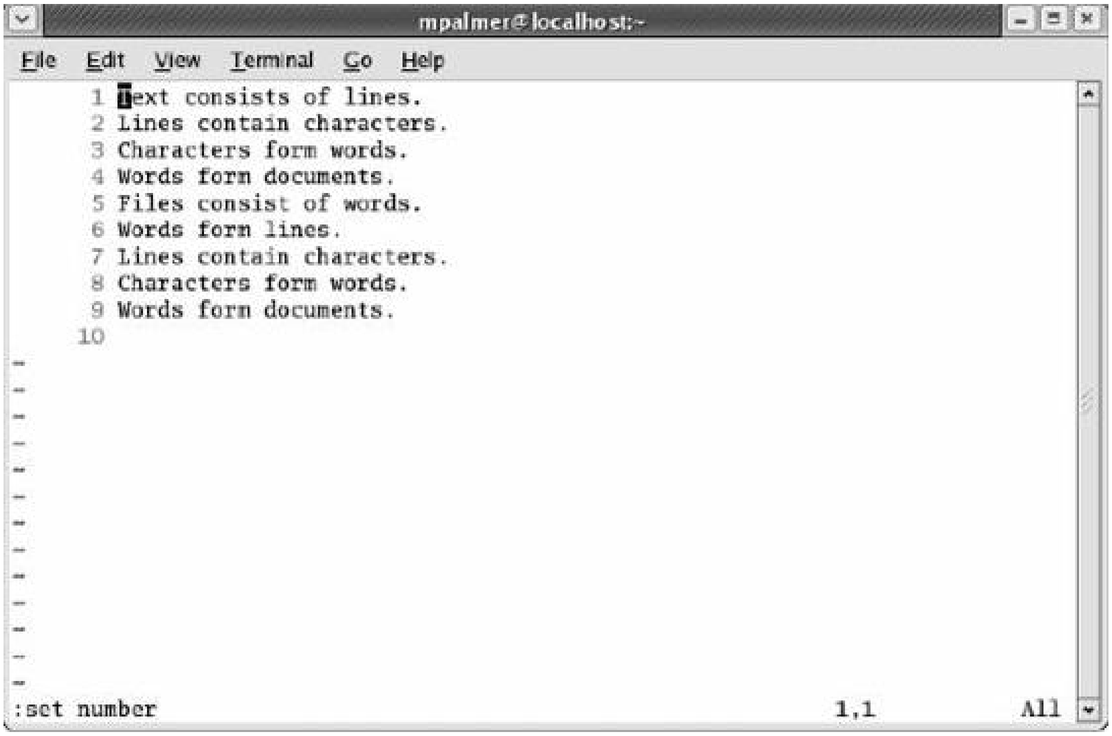
```
# .bashrc

# User specific aliases and functions

# Source global definitions
if [ -f /etc/bashrc ]; then
        . /etc/bashrc
fi
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
:w
```

Figure 4.5. Saving and continuing the editing without quitting

We have mentioned about some colon commands. For example, to have line numbers we may execute :set nu as shown in Figure 4.6.



```
      1 Text consists of lines.
      2 Lines contain characters.
      3 Characters form words.
      4 Words form documents.
      5 Files consist of words.
      6 Words form lines.
      7 Lines contain characters.
      8 Characters form words.
      9 Words form documents.
     10
~
~
~
~
~
~
~
~
~
~
~
~
~
:set number                                              1,1           All
```

Figure 4.6. Displaying line numbers

4.2 EMACS WITH SURVIVAL HINTS

Emacs has a vast range of commands, here is just a brief introduction of the ones which one need frequently.

Emacs command	Function
Ctrl-h	Get online help (*e.g. Ctrl*-h t for tutorial)
Ctrl-x *Ctrl*-f	Open file in new buffer
Ctrl-x *Ctrl*-s	Save active buffer to existing file
Ctrl-x *Ctrl*-w	Save active buffer to specified file
Ctrl-x s	Prompt to save changed buffers to their files
Ctrl-x i	Insert another file
Ctrl-x *Ctrl*-b	List all buffers
Ctrl-x b	Select specified buffer
Ctrl-x k	Kill specified buffer

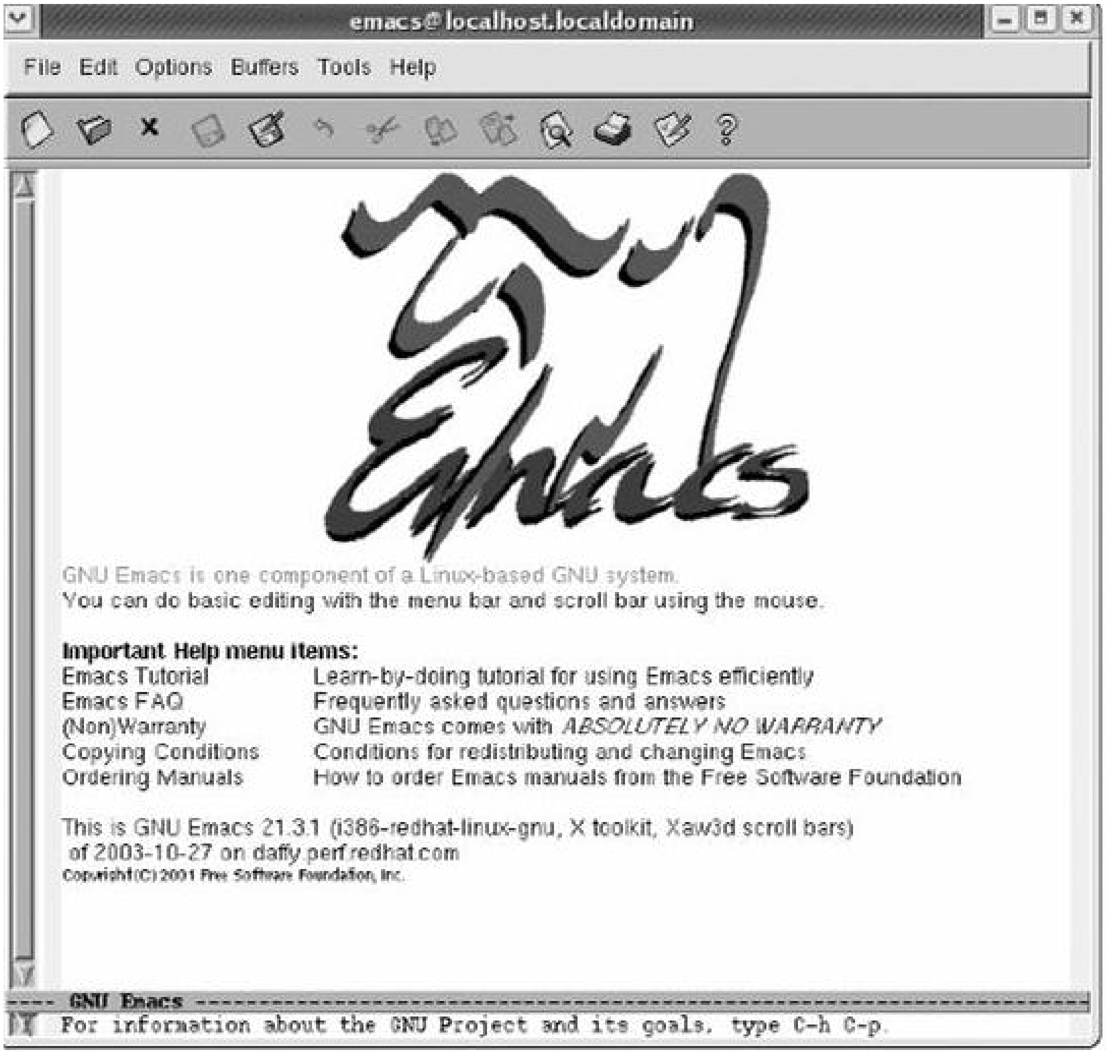
Note that the above file and buffer manipulation commands are not needed for versions of Emacs that run in an X-window and have a File/Buffers menu bar. In the following "motion" commands note that the *Alt Gr* key sometimes works as *Alt*.

Emacs command	Function
Up/down arrow	Move up/down 1 line
Left/right arrow	Move left/right 1 character column
Alt-f	Move forwards one word
Alt-b	Move backwards one word
Ctrl-e	Go to end of line
Ctrl-a	Go to beginning of line
Alt->	Go to end of file
Alt-<	Go to beginning of file
Ctrl-v	Go down one screen full
Alt-v	Go up one screen full
Ctrl-l	Centre current line
Alt-x goto-line*n*	Go to line number *n*
Ctrl-s	Incremental search forwards
Ctrl-r	Incremental search backwards
Alt-%	Global replace with prompt
Ctrl-g	Cancel current operation
Delete	Delete character before cursor
Ctrl-d	Delete character under cursor
Ctrl-k	Kill to end of line or delete blank line
Ctrl-y	Paste in (yank) last block of text killed or selected
Alt-y	Paste in any previous block of text killed or selected
Ctrl-_	Undo last change (may be repeated)
Ctrl-Space	Set mark
Alt-w	Select region between mark & cursor
Ctrl-w	Kill region (cut) between mark & cursor

Note that the above cut and paste commands are not needed for versions of Emacs that run in an X-window and have an Edit menu. Also the mouse can be used to select text by dragging with the left button down, moving the mouse cursor to the insertion point and using the middle button to paste.

Emacs command	Function
Ctrl-c *Ctrl*-r	Column numbers (Fortran mode only - starts at 0!)
Ctrl-c *Ctrl*-w	Make window 72 columns wide for Fortran
Ctrl-x 2	Split window vertically
Ctrl-x 3	Split window horizontally
Ctrl-x o	Select another window
Ctrl-x ^	Make window taller
Ctrl-x }	Make window wider
Ctrl-x 1	Kill all but active window
Ctrl-x 0	Kill active window
Alt-x *shell*	Start shell in Emacs window
Ctrl-x *Ctrl*-c	Exit Emacs

Some of the above functions are not implemented in some versions of Emacs.


emacs@localhost.localdomain
File Edit Options Buffers Tools Help
?

GNU Emacs is one component of a Linux-based GNU system.
You can do basic editing with the menu bar and scroll bar using the mouse.

Important Help menu items:

Emacs Tutorial	Learn-by-doing tutorial for using Emacs efficiently
Emacs FAQ	Frequently asked questions and answers
(Non)Warranty	GNU Emacs comes with *ABSOLUTELY NO WARRANTY*
Copying Conditions	Conditions for redistributing and changing Emacs
Ordering Manuals	How to order Emacs manuals from the Free Software Foundation

This is GNU Emacs 21.3.1 (i386-redhat-linux-gnu, X toolkit, Xaw3d scroll bars)
of 2003-10-27 on daffy.perf.redhat.com
Copyright (C) 2001 Free Software Foundation, Inc.

```
---- GNU Emacs ----------------------------------------------------------------
For information about the GNU Project and its goals, type C-h C-p.
```

Figure 4.7. Emacs screen

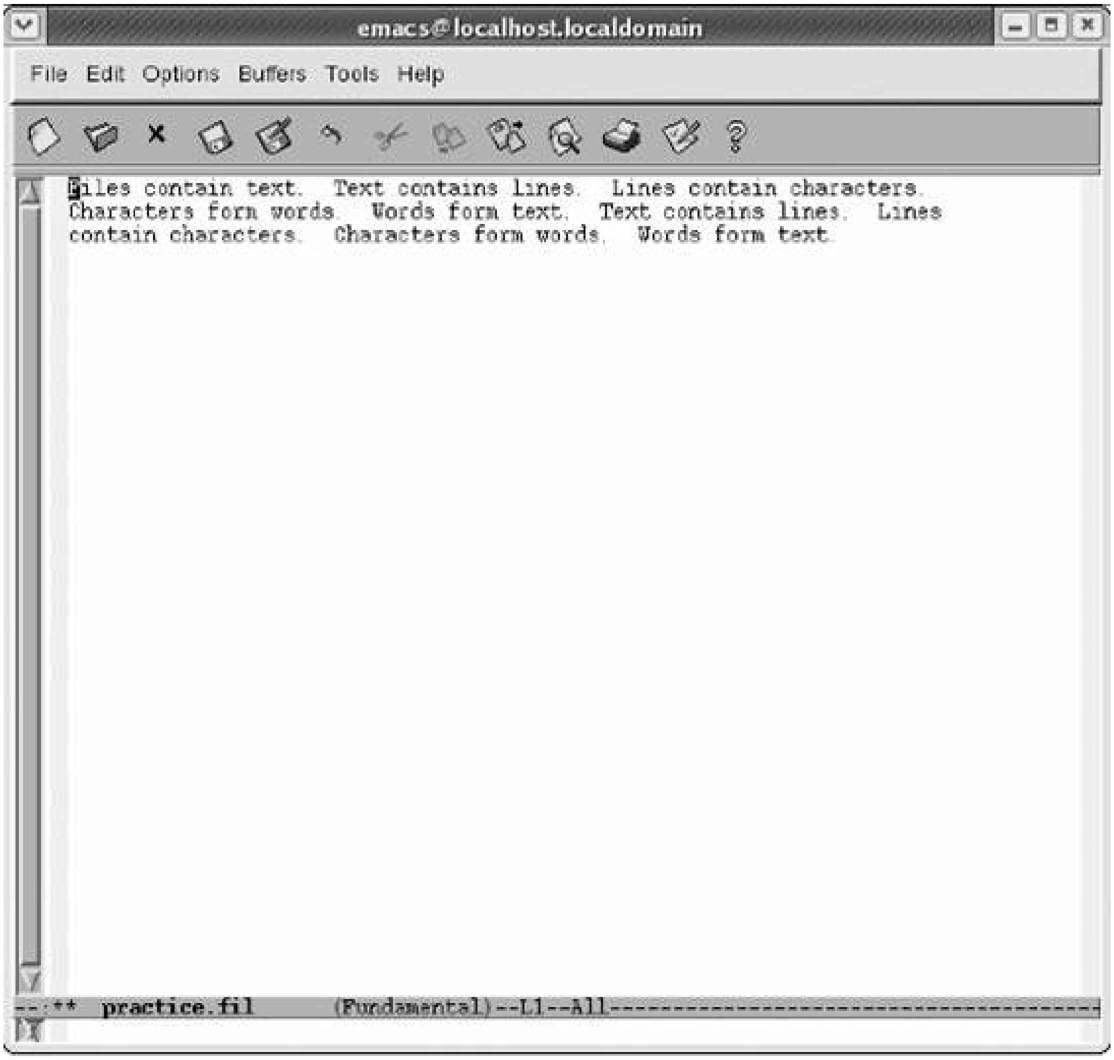
We can start emacs by simply typing its name along with the name of the file to be edited. We will get a screen shown in Figure 4.7. If we are having GNOME desktop, we can select emacs from the start or main menu.

The menu bar at the top of the Emacs screen has categories for commands :

— **File:** Options for file operations
— **Edit:** Text-editing functions
— **Options:** Special options
— **Buffers:** Access to storage buffers
— **Tools:** Options for extra tool functionality
— **Help:** Access to manuals and other help tools

- **Editing an Emacs file**
  - To navigate the cursor location, use the arrow keys, or Ctrl key combinations
  - Text is inserted by typing
  - Ctrl+x and then “u” is the undo command
  - You can cut, copy and paste text
  - There is a text search feature as well
  - Lets you format documents with word wrap

We can press commands as given in the previous tables. For example ALT+q will make emacs to allow word wrapping as shown in Figure 4.8.



```
Files contain text.  Text contains lines.  Lines contain characters.
Characters form words.  Words form text.  Text contains lines.  Lines
contain characters.  Characters form words.  Words form text.
```

```
--:**  practice.fil      (Fundamental)--L1--All--------------------------------------
```

Figure 4.8. Word wrapping under emacs

4.3 CONCLUSIONS

In this chapter we have introduced to vi and emacs editors under UNIX/Linux. Most commonly needed options are explained in detail with illustrative figures.