Tutorial on manipulating files

Taken from http://math.arizona.edu/~support/linux/tutorial.php on August 22, 2006

This tutorial assumes that you have logged in to your account and that you have opened a shell terminal from the graphical interface *(an x-windows xterm)*. In what follows the shell prompt will be denoted simply with the `$` sign. The actual shell prompt on your computer might look quite different. Usually it is of the form `[your_username@machine_name working_directory_name]$`. The boxes below represent small snapshot sections of your shell prompt terminal window. Lines in these boxes, that do not have a `$` sign as a leading character represent the shell's response.

While working at a shell prompt it is easy to lose site of the current working directory. To check in which directory you currently are, use the `pwd` command. Type the following at the shell prompt and hit `<enter>` (always hit the `<enter>` key after issuing a command):

```
$ pwd
/u5/rlakatos
$
```

In the above case the shell reports the working directory as `rlakatos` (which is the home directory of the author of this web page) and `/u5/` as the path leading to it from the `root` directory (the topmost directory in the file system tree denoted by `/`). You will most likely get a similar message on your screen. Note, that the first appearance of the symbol `/` in directory path names always stands for a real directory `/` while all the others are simply there as separator symbols!

To see all the files (in Linux directories are files too) in the current working directory use the `ls` command. Simply type:

```
$ ls
ann     my_documents   papers             research          Work
mail    nsmail         pictures           Screenshot-1.png
Matlab  pal-rujan.pdf  pub_http_internet  temp
$
```

Most likely you will get a different listing. Note, how Linux uses different colors to mark different file types. In the above case, for example, directories are colored blue. Some shells use an additional `/` character at the end of the directory name (for example `my_documents/`) to designate that a file is really a directory.

Let's create a new directory named `tutorial` in the current directory. Use the `mkdir` command as so:

```
$ mkdir tutorial
$
```

To see that you have created this new directory type the `ls` command again:

```
$ ls
ann           nsmail         pub_http_internet  tutorial
mail          pal-rujan.pdf  research           Work
Matlab        papers         Screenshot-1.png
my_documents  pictures       temp
$
```

and observe the name `tutorial` in the above list.

Let's copy a file containing words from a dictionary to this new directory. The dictionary file `linux.words` can be found in the directory `~abe691c/` (which you could now check with the `ls ~abe691c/` command). To copy the file type:

```
$ cp ~abe691c/linux.words tutorial
$
```

To see that the copy command was successful type:

```
$ ls -l tutorial
-rw-r--r--    1 rlakatos users      409305 Aug 18 11:03 linux.words
$
```

As you can see the `ls` command can provide you with additional information about files when you use the `- l` option. There are many other options that can be supplied to the `ls` command. The same is true for almost any Linux command. To see all the options for `ls` and to understand their effects consult the Linux built in manual pages by typing:

```
$ man ls
LS(1)                               FSF
LS(1)

NAME
       ls - list directory contents

SYNOPSIS
       ls [OPTION]... [FILE]...

DESCRIPTION
       List  information  about  the FILEs (the current directory
       by default).
       Sort entries alphabetically if none of -cftuSUX nor --sort.

       Mandatory arguments to long options are mandatory for short
       options too.

       -a, --all
              do not hide entries starting with .

       -A, --almost-all
              do not list implied . and ..

       --author
              print the author of each file
       -b, --escape
              print octal escapes for nongraphic characters
:
```

Now you can browse these pages for information, page by page using the `<page up>` and `<page down>` keys, or line by line using the <↑> and <↓> keys. Try it! To exit the manual hit the <q> key.

Alternatively you can consult the info pages for information about the same command as so:

```
$ info ls
File: coreutils.info, Node: ls invoke, Next: dir invoke, Up: dir list

`ls': List directory contents
=============================

   The `ls' program lists information about files (of any type,
including directories).  Options and file arguments can be intermixed
arbitrarily, as usual.

   For non-option command-line arguments that are directories, by
default `ls' lists the contents of directories, not recursively, and
omitting files with names beginning with `.'.  For other non-option
arguments, by default `ls' lists just the file name. If no non-option
argument is specified, `ls' operates on the current directory, acting
as if it had been invoked with a single argument of `.'.

   By default, the output is sorted alphabetically, according to the
locale settings in effect. (1) If standard output is a terminal, the
output is in columns (sorted vertically) and control characters are
output as question marks; otherwise, the output is listed one per line
and control characters are output as-is.
--zz-Info: (coreutils.info.gz)ls invocation, 48 lines --Top----------
Welcome to Info version 4.3. Type C-h for help, m for menu item.
```

Again you can browse these pages in the same manner and exit with the <q> key.

After quitting the help pages (i.e. the man and info pages) let's switch to our new directory `tutorial` using the `cd` command:

```
$ cd tutorial
$
```

Try now:

```
$ pwd
/u5/rlakatos/tutorial
$
```

As you can see the shell reports the name of the new working directory perpended with the complete path that leads from the *root* directory.

Our dictionary file has a nontraditional `.word` extension, although it is really a text file. To check this, type:

```
$ file linux.word
linux.words: ASCII text
$
```

For convenience, let's rename the file into `my_dictionary.txt` as follows:

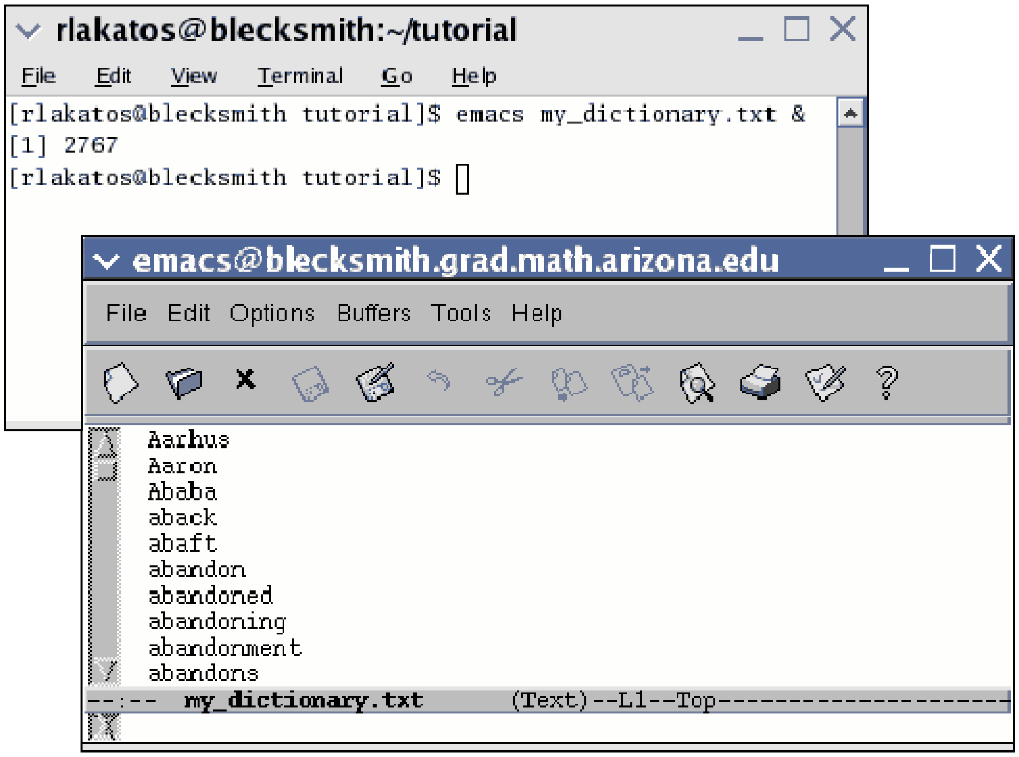
```
$ mv linux.words my_dictionary.txt
$
```

Remark: `mv` stands for *move* and is used to move files around but it can be also useful for renaming files. (Check the result with the `ls` command!)

To edit this file using the *GNU emacs* editor, type:

```
$ emacs my_dictionary.txt &
[1] 22673
$
```

Make sure you add the `&` symbol at the end. That will allow you to retain the shell prompt for further command typing while it will open the emacs editor in a new window. Note: the symbols, like `[1] 2767`, are assigned by the computer to various jobs and processes as identification numbers. In the above case the numbers are assigned to our new emacs window. You will see this shell message if you move away your emacs editor from the shell window as follows. Left click with your mouse on the emacs window title bar and while holding down the left mouse button drag the window to a suitable position on the screen. Your screen should look something like this:


rlakatos@blecksmith:~/tutorial
File Edit View Terminal Go Help

```
[rlakatos@blecksmith tutorial]$ emacs my_dictionary.txt &
[1] 2767
[rlakatos@blecksmith tutorial]$
```

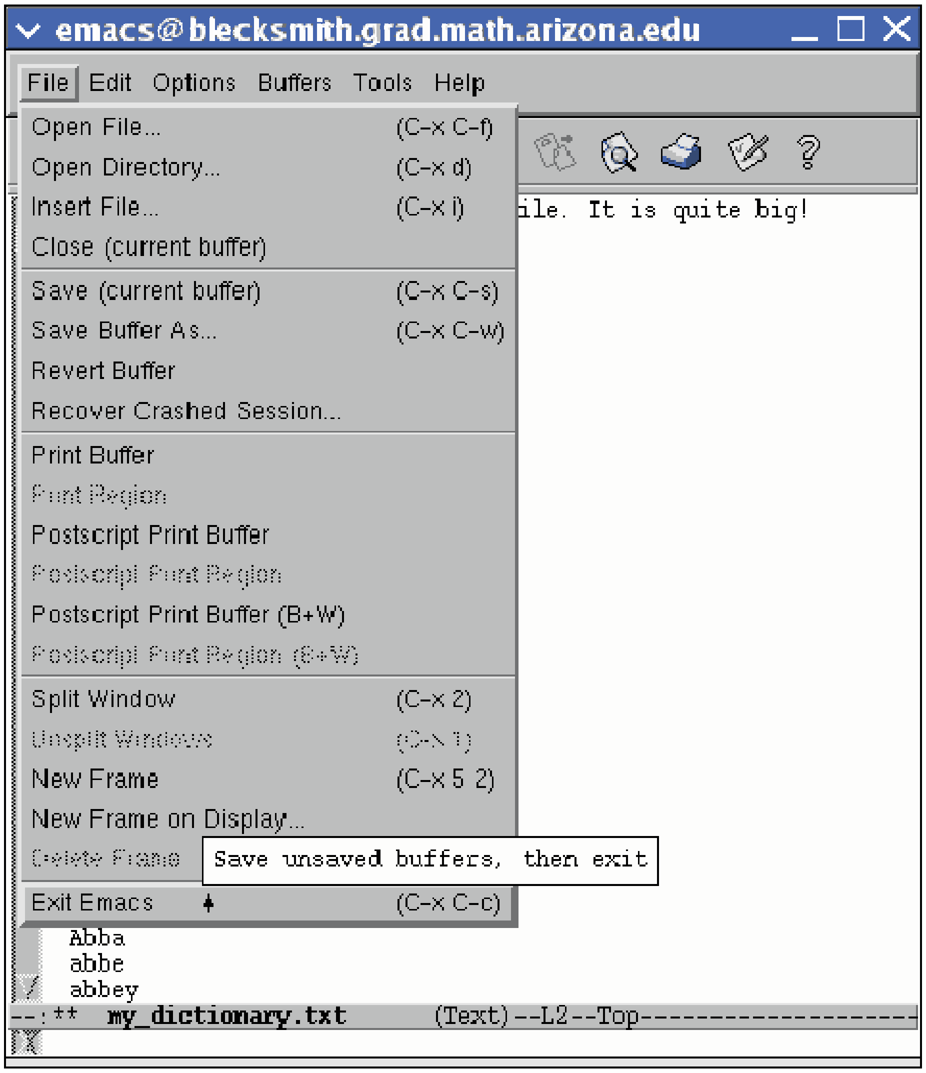
emacs@blecksmith.grad.math.arizona.edu
File Edit Options Buffers Tools Help

```
Aarhus
Aaron
Ababa
aback
abaft
abandon
abandoned
abandoning
abandonment
abandons
--:--  my_dictionary.txt      (Text)--L1--Top--------------------
```

Now, if you wish, you can edit the file. On the top line of the file type the following text (or anything you like):
`Hey, this is my new dictionary file. It is quite big!` and hit the `<enter>` key.

To save the file `my_dictionary.txt` left click with the mouse on the floppy diskette image in the emacs window or press `<ctrl+x>` and follow it immediately by pressing `<ctrl+s>`.

To exit the emacs editor using your mouse go to the `file` menu button in the top left corner of the emacs window, left click on it and choose the `Exit Emacs` option (the last one in the drop-down menu list), or alternatively press `<ctrl+x>` followed immediately by `<ctrl+c>`.


emacs@blecksmith.grad.math.arizona.edu
File Edit Options Buffers Tools Help
Open File... (C-x C-f)
Open Directory... (C-x d)
Insert File... (C-x i)
Close (current buffer)
Save (current buffer) (C-x C-s)
Save Buffer As... (C-x C-w)
Revert Buffer
Recover Crashed Session...
Print Buffer
Print Region
Postscript Print Buffer
Postscript Print Region
Postscript Print Buffer (B+W)
Postscript Print Region (B+W)
Split Window (C-x 2)
Unsplit Windows (C-x 1)
New Frame (C-x 5 2)
New Frame on Display...
Delete Frame
Save unsaved buffers, then exit
Exit Emacs (C-x C-c)

```
ile. It is quite big!
Abba
abbe
abbey
--:**  my_dictionary.txt      (Text)--L2--Top-------------------
```

After exiting emacs, if you press `<enter>` at the shell, you will see a message saying that the emacs job/process is finished.

```
$ <enter>
[1]+  Done                    emacs my_dictionary.txt
$
```

We can see the dictionary file displayed in the shell prompt terminal window by using the pager command `less` which allows you to browse a text file, page by page using the `<page up>` and `<page down>` keys, or line by line using the <↑> and <↓> keys.

```
$ less my_dictionary.txt
Hey, this is my new dictionary file. It is quite big!
Aarhus
Aaron
Ababa
aback
abaft
```

```
abandon
abandoned
abandoning
abandonment
abandons
abase
abased
abasement
abasements
abases
abash
my_dictionary.txt
```

Browse the file and exit it with the `<q>` key.

To end this tutorial we should clean up the file system using the `rm` command:

```
$ rm my_dictionary.text
$
```

The directory `tutorial/` should be empty now and we can remove it with the `rmdir` command. But first we need to get out of it and go back to its parent directory. To achieve all this, type:

```
$ cd ..
$ rmdir tutorial
$
```

Note: the two dots, `..`, tell the `cd` command to leave the current directory and go back to its parent directory which should be your home directory (try the `pwd` command to verify this).

Finally to exit the shell prompt terminal window, type:

```
$ exit
```