

A University of Texas at Austin
Academic Computing and Instructional
Technology Services (ACITS)
Short Course

Introduction to GNU Emacs

A self-paced tutorial for learning to use the
GNU Emacs screen editor

12 February 2001

Introduction to GNU Emacs

Academic Computing & Instructional Technology Services
University of Texas at Austin

Copyright 1991-2001

Unix/OpenVMS Consulting
(512) 475-9400

The exercises on the following pages are for use in a "hands-on" short course to introduce the GNU Emacs screen editor. You can do these exercises on either the ACITS VMS Cluster (UTXVMS, which runs the OpenVMS operating system) or on the ACITS Unix system (uts). Select the system you plan to use in the future. Slight differences exist between the versions of GNU Emacs that are offered on Unix and OpenVMS systems.

NOTE: This handout assumes that you already know how to log in on the system you plan to use. If you do not, take the introductory short course for that system first. For information on obtaining documentation or signing up for short courses, call the ACITS Help Desk at 475-9400. If you are using a Unix system, this handout assumes that you are using the C shell (the default).

Some other versions of Emacs exist. However, because GNU Emacs is the default on all ACITS systems that offer Emacs, this handout uses the term Emacs hereafter to mean GNU Emacs.

Before You Begin

1. If you are enrolled in the short course, the terminal you will use for these exercises has already been set up as a VT100. If you are using a workstation in a lab managed by ACITS, then the workstation should already be set up correctly. For other types of terminals supported on campus, follow the instructions appropriate to the terminal or workstation you are using.
2. Because this short course is taught on both Unix and OpenVMS systems, commands are shown in the lowercase letters required on Unix systems. Unix users must always use the same character case as is shown in the command. On OpenVMS, case does not matter. (The only exception is when you are setting your terminal type for use with a Unix-style product such as Emacs.)

Tasks that you should try out or practice are enclosed in boxes like this one.

- If you have trouble, ask the instructor supervising the short course for help.
- The footnotes in this handout are for those who are working without an instructor. If you are enrolled in the short course, you can ignore the footnotes.

§§ NOTE: Please work through the exercises in the order they are given. Skipping exercises may
§§ result in incorrect settings for the next exercise. (In this document, the §§ marks at the left
§§ indicate important text.) Now you are ready to start working through the exercises.

Contents

The topics discussed in this handout are organized so that you can use the commands you have learned in previous lessons to do the exercises in the next lesson. Thus, the exercises should be done in the order they are presented. Please do not skip around among topics while you are taking the course.

Before You Begin.....	1
Contents.....	2
Documentation Conventions	3
Initialization Files.....	4
Setting Up.....	5
Modes	6
A Practice File.....	7
Running Emacs	7
Terminology	8
Inserting Text	9
Control Characters vs. ESCAPE Characters	10
Aborting or Undoing Commands.....	11
Moving the Cursor	12
Saving Your Editing.....	13
Exiting from Emacs.....	14
Suspending Emacs.....	15
Resuming a Suspended Session	16
Avoiding Overflow Lines.....	17
Repeating Commands	18
Deleting Text.....	19
Restoring or Retrieving Deleted Text	20
Retrieving the Most Recent Deletion	20
Retrieving a Previous Deletion	21
Text Regions	22
Searching for and Replacing Text.....	23
Transposing Text and Changing Case.....	24
Editing Multiple Buffers	25
Viewing Multiple Buffers	26
Including Files.....	27
Macros.....	28
Using Emacs Internal Help.....	29
Using the Info System	30
Additional Information.....	31
Additional Practice.....	31
Logging Out	32

Documentation Conventions

One or more words in a box, such as `RETURN` or `CTRL C` , represent a key or key sequence that you should press.

- A "control character" such as `CTRL Z` means to hold down the key marked CTRL and type a lowercase letter (in this case, z). These same characters are represented online in the form ^z or CTRL-z (on Unix) and CTRL-Z or CTRL/Z (on OpenVMS). In online Emacs documentation, they are sometimes represented in the form C-z.
- A "meta character" is represented by a box containing "ESC" and a letter. For example, `ESC x` means to press and release the ESCAPE key (Meta key) and type the letter x. In the online Emacs documentation, meta characters are represented in the form M-z.

When typing the commands shown in the exercises:

- Type the **boldface** words in command lines exactly as shown. On Unix, even the case must be the same; on OpenVMS, character case does not matter.
- Substitute an appropriate value for items in *italics*. For example, substitute an appropriate Unix filename or an appropriate OpenVMS file specification for *file* where it appears in a command line.
- Terminate each command by pressing the `RETURN` key.

For example, a Unix command (for copying a file from another directory into your own directory) and a similar OpenVMS command are shown below:

Unix: `cp file . RETURN`

OpenVMS: `copy file [] RETURN`

§§ NOTE: The standard prompts for the operating systems on ACITS systems are

§§ Unix: %

§§ OpenVMS: \$

§§ If your user number has a file named **.cshrc** (on Unix) or **login.com** (on OpenVMS) in the

§§ top-level directory, this file may contain commands that change the prompt to something else.

§§ A common choice is the name of the system on which you are logged in.

Two examples (Unix and OpenVMS) are:

uts.cc.utexas.edu> (instead of %)

VIOLET> (instead of \$)

When you see these prompts, remember that they are simply a fancy version of the regular system prompt. For example, if you are using Unix on uts with the standard `.cshrc` file, you will see the prompt "uts.cc.utexas.edu> ". However, this is the same as "%" and anything that you can type at the % prompt, you can also type at the "uts.cc.utexas.edu>" prompt.

Initialization Files

Decide whether you want to take the course on a Unix or an OpenVMS system. Log in on the system of your choice.

You can use an initialization file named ".emacs" to customize Emacs for your own use. Whenever you run Emacs, the commands in your initialization file are executed before you begin editing. In this exercise, you will copy an initialization file to use in this course.

If you are using Unix:

- Copy the initialization file by typing the following command at the % prompt:

```
cp /usr/local/lib/shortcourse/630/.emacs . 
```

NOTE: Be sure that you include the single-period character at the end of the line above. It represents the current directory (in this case, your login directory). The file will have the same name as the original (.emacs). Use **ls -a** to list your files, if necessary.

If you are using OpenVMS:

- Copy the initialization file by typing the following command at the \$ prompt:

```
copy sc:[730].emacs [] 
```

The empty square brackets represent the current directory (in this case, your login directory). The file will have the same name as the original (.emacs). Use **dir** to list your files, if necessary.

Setting Up

Before you run Emacs, you must set up your terminal correctly. This must be done once during each login session. To simplify matters, you can place the setup command in your login initialization file for the system (**.login** or **LOGIN.COM**). Once the setup commands are in effect, they do not need to be typed again during that login session (unless you deliberately change the settings) no matter how many times you run Emacs.

On Unix, the syntax of the command to set up your terminal is:

```
setenv TERM termtype RETURN
```

On OpenVMS, the syntax of the command to set up your terminal for using Emacs is:

```
define term "termtype" RETURN
```

You will probably also have to disable some control keys CTRLc , CTRLs , CTRLq that will interfere with your use of Emacs. The method of doing this will vary with the type of terminal you are using.

If you are using Unix:

- Set up your terminal¹ by typing at the % prompt:

```
setenv TERM vt100 RETURN
```

Remember to type the character case of the boldfaced characters exactly as shown, using lowercase for setenv and uppercase for TERM.

- To disable the control keys on the Macintosh workstations used in the short course, look in the Telnet **Session** menu and select **Setup Keys**. Delete all three entries and click OK.

If you are using OpenVMS:

- Set up your terminal¹ using the command below:

```
define term "vt100" RETURN
```

Be sure vt100 is in lower case. Don't forget the quotation marks. They are necessary.

- To disable the control keys on the Macintosh workstations used in the short course, look in the Telnet **Session** menu and select **Setup Keys**. Delete all three entries and click OK.

¹If you are taking the course on your own terminal, you may need to omit this command or substitute a different terminal type for vt100 in the **setenv** or **define** command. The default login initialization files for ACITS systems attempt to determine your correct terminal type. To disable the control characters, be sure your terminal type is defined correctly and that flow control is turned off ("NO XON/XOFF").

Modes

When you edit a file, Emacs uses a portion of the file name (on OpenVMS called the filetype and on Unix called an extension or suffix) to aid you in your work by ensuring that the "mode" of your work conforms to the formatting allowed by that type of file. Some common filetypes or suffixes that Emacs recognizes are shown in the chart below:

Type or Suffix	Description	Mode name	Example
c	C programs	C	prog1.c
lsp	Lisp files	Lisp	sample.lsp
tex	TeX/LaTeX files	TeX Fill	document.tex

If the file has a type or suffix that Emacs does not recognize, Emacs uses a mode called "Fundamental." Later in this script, you will learn how to edit more than one file at a time. When you do this, Emacs can use a different mode for each of the files.

NOTE: On Unix systems, suffixes are less frequently used than on OpenVMS. They can be omitted according to user preference.

A Practice File

Before you copy the practice file, decide what type of work you will usually be doing, and give that filetype or suffix to your practice file so that you will become used to using Emacs in the mode you will eventually need. If you cannot decide or do not know the appropriate filetype or suffix for the work you will be doing, talk to the instructor or simply use **txt**.

Use the commands below to copy an initialization file to use during the short course and to copy a file containing some text for you to edit.

If you are using Unix:

- Use the command below to copy your practice file. Give it any filename you want. You can use a suffix if you plan to use one in your editing later.

```
cp /usr/local/lib/shortcourse/630/emacs.tasks file 
```

If you are using OpenVMS:

- Use the command below to copy your practice file. Give the file any name you want, but use the filetype you have decided is best for you.

```
copy sc:[730]emacs.tasks file 
```

Running Emacs

The command to run Emacs is the same on Unix and OpenVMS systems. It is:

```
emacs file 
```

If you are using Unix or OpenVMS:

- To run Emacs, substitute an appropriate file specification for *file* in the command below:

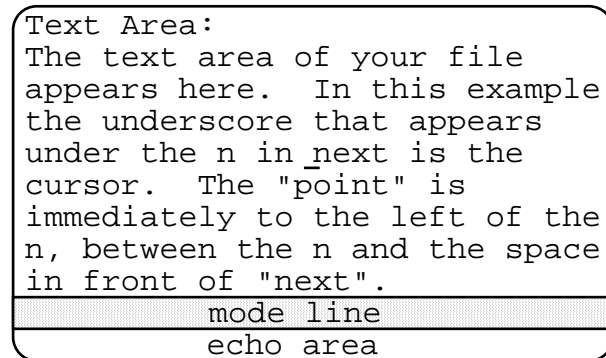
```
emacs file 
```

Terminology

These terms will be used throughout the course. Use the figure below to become familiar with them.

- buffer** A memory space that contains a file or piece of text that you are editing. You can use more than one buffer at a time during an editing session.
- cursor** The small blinking underscore you see on your terminal. On some terminals, it may be a small blinking rectangle. The cursor indicates where you are in the file.
- point** An imaginary point located to the left of the cursor, between characters. When you give editing commands in Emacs, they are usually performed at the point.
- window** Normally, the window is your terminal screen, which displays a portion of the buffer. Later you will learn how to divide your screen into more than one window. Each window is divided into a text area, a mode line, and an echo area.

The Window: Your Terminal Screen



- text area** The portion of the window that contains the text of the buffer you are editing.
- mode line** A line, in reverse video, near the bottom of the screen. This line displays the buffer name, mode, and location in the file.
- echo area** (Sometimes called a "minibuffer") The line at the bottom of the screen, where Emacs displays commands and errors.

On Unix or OpenVMS:

Look at your terminal screen and do the following:

- Find the cursor and figure out where the "point" is right now.
- Determine what constitutes the "text area" and read the first two paragraphs of the text of the buffer displayed there.
- Be sure that you have found the mode line in reverse video and that you understand the information that it provides for you: the buffer name (your file name), the mode, and the location.
- Notice the echo area (the blank line below the mode line), even though it has nothing written in it right now.

Inserting Text

The cursor is the small blinking line or box you see in the window. Right now it is probably in the upper lefthand corner of your screen. Remember that the point is immediately to the left of the cursor. The point is the place at which your editing commands will take effect.

When you type text, it will automatically be inserted into the file at the point. When you reach the right side of your screen, press `RETURN` and continue typing. By default, if you do not press `RETURN` and your text goes beyond the end of the line, you will see a backslash (\) at the right side of your screen and the line will wrap around.

§§ Avoid leaving wrap-around lines in files that you are editing. These long lines can cause
§§ problems when the files are used.

To delete a character to the left of the cursor (such as a character that you have just inserted), you can press the `DELETE` key. You will learn more sophisticated ways to delete text and to eliminate wrap-around lines later in the course.

On Unix or OpenVMS:

- Type a character on your keyboard. It will be inserted at the point. Insert another character or two to see how the position of the point relates to the text that you are inserting.
- Use the `DELETE` key to delete one of the characters that you have just inserted.

Control Characters vs. ESCAPE Characters

Control characters and ESCAPE characters have been mentioned before, but they are so important in using Emacs that they will be discussed in more detail here.

To type a control character:

- Press *and hold down* the `CTRL` key
- Type the character.

To type an ESCAPE character (meta character):

- Press the ESCAPE key.
- Release the ESCAPE key.
- Type the character.

On Unix or OpenVMS:

To practice using control characters and ESCAPE characters, look through the sample file by using `CTRL V` to move forward and `ESC V` to move backward through the file.

- Type a `CTRL V` to move forward one screen :
 - Press and hold the `CTRL` key.
 - While still holding the `CTRL` key, type the lowercase letter `v`.You should see the next screenful of text in the window. Repeat `CTRL V` , as necessary, until you reach the end of the file.
- Type `ESC V` to move backward one screen:
 - Press the ESCAPE key.
 - Release the ESCAPE key.
 - Type the lowercase letter `v`.You should see the previous screenful of text in the window. Repeat `ESC V` , as necessary, until you reach the beginning of the file.
- Practice moving backward and forward until you are comfortable with the difference between `CTRL V` and `ESC V` .

Aborting or Undoing Commands

§§ When you make mistakes or typographical errors, use "abort" or "undo" commands
§§ to restore your editing or return to a mode where you can give the correct command.

Abort and Refresh. If you begin typing a command then change your mind, you can cancel the command without completing it.

`CTRL g` Cancels a command that you have started typing, but which you have decided that you do not want to execute.

`CTRL l` `CTRL` key and lowercase *l* refreshes the screen. Useful because sometimes canceling a command does not remove the old display. `CTRL l` erases the old display and lets you see the current condition of the file.

NOTE: To cancel a command, press first `CTRL g` and then `CTRL l`. You can then be sure that what you see on the screen is accurate.

Undo. If you fail to abort and do make an error, you can undo the erroneous command.

`CTRL x u` ("Undo") Reverses the effect of the last command you typed. Repeat to undo the command before that, and so forth.

`CTRL _` (CONTROL SHIFT underscore) Same as `CTRL x u` above. This form does not work on some keyboards.

Remember that if you make a mistake typing a command during the short course, you can use that as an opportunity to practice using these commands to undo the error.

On Unix or OpenVMS:

- Pretend that you had begun to type a command that started with `ESC x` and then decided that you did not want to do it. Press `ESC x` now.
- Now cancel the partial command by pressing `CTRL g`. Note the message in the echo area.
- Press `CTRL l` to refresh the screen. You will see the screen rewrite itself as the refresh is done. In this case, not much else will change. However, if the partial command had displayed extra lines or messages on the screen, they would have been erased.
- Insert a word and use `CTRL x u` to undo the insertion.
- Press `CTRL l` to refresh the screen again.

Moving the Cursor

The commands below move the cursor to another position in the file. The letters used in commands are sometimes reminders of the commands' functions: `CTRL n` moves your cursor to the *next* line, for example. Also note that an ESCAPE character often performs a task similar to a corresponding CONTROL character, but uses a larger unit or reverses the direction. For example, `CTRL b` backs up one *character*, and `ESC b` backs up one *word*.

<code>CTRL p</code>	Moves to the previous (P) line.
<code>CTRL n</code>	Moves to the next (N) line.
<code>ESC v</code>	Moves to the previous screen.
<code>CTRL v</code>	Moves to the following screen.
<code>CTRL b</code>	Moves backward (B) one character.
<code>CTRL f</code>	Moves forward (F) one character.
<code>ESC b</code>	Moves backward one word. A space or a punctuation mark (such as a period) indicates the end of a word.
<code>ESC f</code>	Moves forward one word.
<code>CTRL a</code>	Moves to beginning (A) of the line.
<code>CTRL e</code>	Moves to end (E) of the line.
<code>ESC a</code>	Moves to beginning of the sentence.
<code>ESC e</code>	Moves to end of the sentence. Emacs recognizes a period (or end-of-line mark) followed by several spaces as defining the end of a sentence.
<code>ESC [</code>	Moves to beginning of the paragraph. (This key sequence may have a different function on some VT100 terminal emulators. If this is the case with your terminal, use one of the other commands to move the cursor to this position.)
<code>ESC]</code>	Moves to end of the paragraph.
<code>ESC <</code>	Moves to beginning of the buffer (file).
<code>ESC ></code>	Moves to end of the buffer (file).

On Unix or OpenVMS:

- Try using each of the cursor movement commands above to move around in the text file.
- When you are sure you understand all the cursor movement commands, move the cursor to the beginning of the file and continue with the exercises.

Saving Your Editing

Save your work before you exit. If you exit without saving your work, Emacs will usually ask you if you want to save it. However, try to save your work frequently. That way, if you make a mistake that you cannot fix and you must discard your editing to start again (or if the system goes down unexpectedly for some reason), you will lose less work.

`CTRL X` `CTRL S`

Saves the current buffer using the buffer name as the name of the file.

`CTRL X` `CTRL W` *file* `RETURN`

(Write) Saves the current buffer into the file named *file*.

In both of the above commands:

- On Unix, the previous version is overwritten.
- On OpenVMS, the file is given a version number *n*, in the format "*name.type;n*" so that for each new version of the file, *n* is one integer higher than when you last saved the file. All versions remain in your directory until you delete them.

Auto-Save mode saves your file whenever you have done a certain amount of editing. Auto-Save is on by default.

`ESC X` **auto-save** `RETURN`

Turns auto-save on or off. When on (default), Emacs automatically saves your file at certain intervals.

On Unix or OpenVMS:

- Use `CTRL X` `CTRL S` to save the current file. If you had made no changes, Emacs would display that information in the echo area. Since you have altered the file, Emacs will display a message that the file was saved. Notice the message in the echo area.

NOTE: If your terminal locks up when you type `CTRL S`, you probably forgot to disable the Setup Keys. In that case, press `CTRL Q` to release your frozen screen, and then talk to the instructor or check your terminal settings (see the "Setting Up" section).

- Turn Auto-Save mode off and on again. Be sure it is on before you proceed.

§§ NOTE: If you get an error message after you try to save a file, you may have used up all of your
§§ disk space. If this is the case, you may have to delete some old files before you can save your
§§ editing. If this happens frequently, you may need to request more disk space by sending e-mail
§§ to help@cc.utexas.edu.

- On Unix, delete old files using the following command at the % prompt:

rm *file* `RETURN`

- On OpenVMS, delete old files using one of the following commands at the \$ prompt:

delete *filename.filetype;n* `RETURN`

purge *fileError!name.filetypeError!*

Use **man rm** on Unix or **help delete** (or **help purge**) on OpenVMS to learn more about these commands. To return to the prompts where you can issue these commands, you must suspend Emacs. You will learn how to do that later in this course.

Exiting from Emacs

Be sure that you have saved your editing changes before you exit from Emacs. If you do not, your changes may be discarded. To exit from Emacs, use the following Emacs command:

`CTRL X` `CTRL C`

Exits (terminates) Emacs. Exit this way when you are finished with Emacs for the current login session. If you have unsaved changes, you will be asked whether you want to save them.

If you need to discard some extensive editing, use the above command sequence to exit, answering *no* when you are asked if you want to save the file. All changes you have made since your last save will be discarded.

If you are using Unix:

- Save your editing changes by pressing `CTRL X` `CTRL S` .
- Terminate the Emacs session completely (as if you were finished editing for the day) using `CTRL X` `CTRL C` . This is the command you will use when you are finished with the short course.

If you are using OpenVMS:

- Save your editing changes so far, by pressing `CTRL X` `CTRL S` .
- Terminate the session completely (as if you were finished editing for the day) by pressing `CTRL X` `CTRL C` . This is the command you will use when you are finished with the short course.

Suspending Emacs

You can suspend your editing session, if you plan to return to it later to continue your editing.

NOTE: If you use Emacs in an X window environment, your Emacs session will appear in a separate window from the one you were using. In that case, there is no need to suspend Emacs and you should avoid doing so. You can work in your other windows and return to the Emacs window by simply selecting the window.²

`CTRL Z`

Suspends (stops) Emacs and returns to the system prompt (usually % on Unix or \$ on OpenVMS). Exit this way when you intend to do more editing during the current login session. The command you use to resume the session is different depending on whether you are using Unix or OpenVMS and will be discussed on the next page.

§§ NOTICE: If you use `CTRL Z` to suspend Emacs on a Unix system, avoid re-issuing the
§§ **emacs** command to run Emacs again in the same login session. This creates additional
§§ processes that needlessly waste the system resources available to you and to other users.

If you are using Unix:

- Run Emacs again by substituting the name of your practice file in the command:

`emacs file` `RETURN`

- Move your cursor to somewhere in the middle of the file.
- Suspend Emacs by pressing `CTRL Z` .

If you are using OpenVMS:

- Run Emacs again by substituting the name of your practice file in the command:

`emacs file` `RETURN`

- Move your cursor to somewhere in the middle of the file.
- Suspend Emacs by pressing `CTRL Z` .

² If you are taking the short course on your own and a separate Emacs window was spawned, skip the exercises on this page and the next page.

Resuming a Suspended Session

On Unix: This information assumes that you are using the C shell (the default). If you have chosen to use another shell, the commands to resume your suspended session may be different.

To resume a suspended session, use the Unix command below to obtain the job number:

```
jobs 
```

and then substitute that number for n in the command:

```
fg %n 
```

You will be returned to your editing at the same point that you left it.

Remember that if you used to suspend an Emacs session on a Unix machine, then avoid using the **emacs** command again while the first session still exists. Creating additional processes needlessly wastes system resources.

On OpenVMS: To resume a suspended session on OpenVMS, use the DCL command:

```
emacs 
```

You will be returned to your editing at same point that you left it.

If you are using Unix:

- Type **jobs** at the % prompt to get the job number.

```
jobs
```

You will see a line with a number in it. This number is the job number.

- Resume the suspended session by substituting the job number shown for n in the command:

```
fg %n 
```

- Notice that your cursor is still in the same position that it was when you suspended Emacs.

If you are using OpenVMS:

- Resume the suspended session by typing the following at the \$ prompt:

```
emacs 
```

- Notice that your cursor is still in the same position that it was when you suspended Emacs.

Avoiding Overflow Lines

§§ NOTE: When inserting or adding text, avoid leaving overflow lines³ in a file. If the file is the text of a mail message or if the file is transferred to another system, these overflow lines may be unreadable, lost, or truncated. Frequently they prevent the file or message from being sent.

You can set a special mode called "Auto-Fill," which will eliminate the problem of overflow lines. Either turn on Auto-Fill mode or remember to press `RETURN` at the end of each line on the screen.

`ESC q` Rids the current paragraph of all overflow lines (those with a backslash at the right margin).

`ESC x` **auto-fill** `RETURN`
Turns auto-fill on or off. When auto-fill is on, Emacs automatically moves words to the next line when they don't fit on the current line. The default is *off*.

On Unix or OpenVMS:

- Use `ESC >` to move to the end of the buffer. Start a new paragraph and type several sentences without pressing `RETURN`, so that you have some overflow lines. Notice the backslashes at the right margin.
- Use `ESC q` to realign the overflow lines and eliminate the backslashes.
- Set auto-fill mode by typing:
`ESC x` **auto-fill** `RETURN`
Notice that the echo area shows you the command as you type it.
- Use `ESC >` to go to the end of the buffer again. Start another new paragraph and enter several more sentences without pressing `RETURN`. Watch what occurs now when you press the space bar at the end of a long line.
- Leave Auto-Fill mode set for the rest of the course.
- Press `ESC <` to move back to the beginning of the buffer.

³ An overflow line is a line that runs beyond the edge of the screen.

Repeating Commands

You can tell Emacs to perform a command a specific number of times. Use the command sequences below to do this.

`ESC` *n* `command`

Repeats the Emacs command you substitute for `command` *n* times. (This works with the cursor movement commands, but does not work with screen movement commands.) You can substitute any positive or negative integer for *n*. For example:

`ESC` 13 `CTRL p`

`CTRL u` *n* `command`

Same as the above.

On Unix or OpenVMS:

- Move down 10 lines using:

`CTRL u` **10** `CTRL n`

- Move up 10 lines using:

`ESC` **10** `CTRL p`

- Practice repeating some other commands that you know.

Deleting Text

In Emacs, the commands you use to delete text may vary depending on whether you want to move the text to another location or discard it entirely.

`DELETE` Deletes one character backward from the point. Characters deleted this way cannot be moved or restored.

`CTRL d` Deletes one character forward from the point. Characters deleted this way cannot be moved or restored.

`ESC DELETE` Deletes one word backward from the point. A space or a punctuation mark (such as a period) defines the end of a word.

`ESC d` Deletes one word forward from the point.

`ESC 0` `CTRL k` (That's ESCAPE-zero.) Deletes the line backward from the point. If you are in the middle of a line, this command deletes back to the beginning of the line. If you are already at the beginning of the line, this command accomplishes nothing.

`CTRL k` Deletes one line forward from the point. If you are in the middle of a line, this command deletes the remainder of the line. If you are already at the end of the line, this command deletes the carriage return at the end of the line and appends the next line to the end of the current line.

`CTRL x` `DELETE` Deletes backward to the beginning of the current sentence. Emacs recognizes a period or an end-of-line mark followed by several spaces as defining the end of a sentence.

`ESC k` Deletes forward to the end of the current sentence.

On Unix or OpenVMS:

- Insert some characters and use the `DELETE` key to delete them.
- Insert several words into the text anywhere in the file and then use `ESC DELETE` or `ESC d` to delete them.
- Insert four or five sentences into the file to make a paragraph several lines long.
- Use `CTRL x DELETE` and `ESC k` to delete parts of the sentences.
- Use `ESC 0 CTRL k` and `CTRL k` to delete the rest of the lines.
- Repeat the three previous steps until you understand how Emacs defines sentences and lines for the purpose of these commands.

Restoring or Retrieving Deleted Text

Emacs preserves up to 16 deleted text areas of one or more words, and any of this text can be restored or retrieved. Thus, deleted text is not really lost until after 16 more deletions have been made. The "list" of deletions is circular and your position in the circle depends not only on what was deleted last, but on which text you last retrieved.

Retrieving the Most Recent Deletion

Use `CTRL-y` to retrieve the text that you most recently deleted. You can use `CTRL-y` in the following ways:

- To restore text to the same position (e.g., when you delete something by accident).
- To insert a piece of text at several different points throughout your file.
- To move text from one location to another (the "cut-and-paste" concept).

NOTE: Text deleted character by character using the `DELETE` key or `CTRL-d` is not preserved and cannot be retrieved.

`CTRL-y` Retrieves (yanks back) the last deleted text and inserts it at the point. To yank back multiple copies, simply repeat the command.

NOTE: If you performed several delete commands at the same location without moving the cursor between them, Emacs treats them as a single deletion, and `CTRL-y` will restore all the deleted text at once.

On Unix or OpenVMS:

Move a line from one paragraph to the next.

- Move your cursor to the beginning of a line and press `CTRL-k` to delete one line forward from the point.
- Move the cursor to a position in the next paragraph.
- Press `CTRL-y` to retrieve the text and insert it at the current cursor location. Press `RETURN`, if necessary, to cause the lines to display properly again.

Delete a word from one paragraph and re-insert it. Then insert it again in two other places.

- Press `ESC d` to delete the word to the right of the cursor.
- Without moving the cursor, press `CTRL-y` to restore the word to the same position as before.
- Move the cursor to a position in the next paragraph.
- Press `CTRL-y` to retrieve the word and insert it at the current cursor location.
- Move the cursor again to another location, and press `CTRL-y` again.
- Move the cursor one more time to another position, and press `CTRL-y` again.

Retrieving a Previous Deletion

Remember that Emacs retains up to 16 deletions. Use the command below to retrieve previous deletions.

`CTRL y` `ESC y` Retrieves the next-to-last deleted text. Repeating `ESC y` moves backward through the list of deletions until you find the deleted text that you want. (Remember that if 16 deletions have occurred since you deleted the text, it is gone for good.)

§§ NOTE: When you delete text that you plan to insert at another location, it is a good idea to move promptly to the new location and make the insertion, so that you do not accidentally lose the deleted text as you continue your editing.

On Unix or OpenVMS:

The purpose of the exercises below is to learn to group three separate lines together in a particular order.

- Insert three sentences at different locations in the file. End each sentence with a period and two spaces. Give the sentences a sequence so that you can tell which one is first, second, and third.
- Move the cursor to the beginning of the first sentence and use `ESC k` to delete it.
- Move the cursor to the beginning of the second sentence and use `ESC k` to delete it. Repeat this for the third sentence.
- When you have deleted all three sentences, move to another location in the file and press `CTRL y` to insert the third sentence there. Insert the two spaces after the period.
- Press `ESC a` to move the cursor to the beginning of that sentence.
- Press `CTRL y` `ESC y` to insert the second sentence before the third one. The third sentence will appear for a moment when you press `CTRL y`, but when you press `ESC y`, sentence number two will replace it. Insert the two spaces after the period.
- Press `ESC a` to return to the beginning of the second sentence.
- Press `CTRL y` `ESC y` again. This time when you press `CTRL y` sentence number two (the last retrieval) will appear momentarily, to be replaced by sentence number one.
- Use `ESC a` to move to the beginning of sentence number one.
- Delete the three sentences again, moving the cursor between each deletion to separate the sentences. This time insert them in reverse order (three, two, one) in a new location.

Text Regions

You can mark a region of text that you want to copy, delete, or change. This enables you to copy or move large blocks of text from one place in the file to another or even from one file to another. To move text from one file to another, you will have to edit more than one buffer at a time. You will learn how to do this later in the course.

Marking a text region:

`CTRL @` Marks the beginning of a region of text. Move the cursor to define the region. The cursor position marks the end of the region.

`CTRL SpaceBar` Same as the above (on some terminals). Use this on the Macintoshes at the Student Microcomputer Facility.

`CTRL X` `CTRL X` Exchanges locations of the point and the mark. Pressing `CTRL X` `CTRL X` again moves you back and forth between the region's beginning and its end.

NOTE: Some commands, such as `ESC <` or a search terminated by an `ESC`, also set marks when they are executed, since Emacs assumes that your use of the command indicates that you want to be at that location. The marks set by these commands will supersede any mark that you have set with a previous `CTRL @` or `CTRL SpaceBar` command.

Using the text region:

`CTRL W` Cuts ("wipes" or deletes) the marked region and places the deleted text in the circular list of deletions.

`ESC W` Copies the region of text. You can now insert the text at another location, but the text remains in its current position as well.

`ESC =` Reports the number of lines in the region you've marked.

To restore or retrieve regions of text, use the same commands you learned on the previous page.

On Unix or OpenVMS:

Copy a text region from one part of your file to another by doing the following:

- Mark the beginning of a paragraph as the beginning of a region of text.
- Move to the end of the paragraph to define a region.
- Use `CTRL X` `CTRL X` to switch between the beginning and the end of the region.
- Use `ESC W` to copy the region. Notice that the text still remains in the file.
- Use `ESC <` to move to the beginning of the file and use `CTRL Y` to insert the copied text at that point.

Move a text region from one part of your file to another by doing the following:

- Mark another paragraph to be a region.
- Use `ESC =` to display the number of lines in the region.
- Use `CTRL W` to delete the region. Notice that it is gone from the original location. It is now in a buffer in the list of deletions described on the previous page.
- Move to the end of the file and use `CTRL Y` to insert the deleted text at the new location.

Searching for and Replacing Text

Use the commands below to search for text strings. You can also cause the string, if found, to be replaced by a different one. Searches can be incremental or non-incremental.

Incremental searches: As you type each character, Emacs advances to the next occurrence of the string you have entered so far. This is the type of search most often used in Emacs.

CTRL S *text*

Searches forward for *text*. To proceed to the next occurrence of *text*, press **CTRL S** again. Press **ESC** to end the search, and leave the cursor at the last occurrence found or press **CTRL g** to return the cursor to the point at which the search began.

CTRL F *text*

(Reverse) Searches backward for *text*. To go to the previous occurrence of *text*, press **CTRL F** again. Press **ESC** to end the search, and leave the cursor at the last occurrence found or press **CTRL g** to return the cursor to the point at which the search began.

Non-incremental searches: With non-incremental searches, Emacs waits until you have completed the entire string and pressed **RETURN** before it begins its search.

CTRL S **ESC** *text* **RETURN**

Searches forward for *text*.

CTRL F **ESC** *text* **RETURN**

(Reverse) Searches backward for *text*.

Replacing text:

ESC % *oldtext* **RETURN** *newtext* **RETURN**

Moves to first occurrence of *oldtext*, asks for instructions, and replaces *oldtext* with *newtext*, based on your response. Possible responses are:

! Replaces all (or all remaining) occurrences of *oldtext*.

SpaceBar Replaces only the current occurrence of *oldtext*.

DELETE Does not replace the current occurrence of *oldtext*, but continues the replacement session.

ESC Terminates the replacement session.

On Unix or OpenVMS:

- Use **ESC <** to move to the beginning of the file.
- Use **CTRL S** to search forward for any text string (the word *cursor*, for example). Type the text string slowly, character by character, and watch the action of the incremental search.
- Search forward for the next occurrence of the same text by pressing **CTRL S** again.
- Terminate the search.
- Move to the beginning of the sample file, and replace any or all occurrences of the word *cursor* with the word *point*.

Transposing Text and Changing Case

Use the following commands to transpose text or change character case:

CTRL t

Transposes characters around the point.

SPECIAL: If your cursor is at the end of a line, the two previous characters are transposed. At the beginning of a line, the end-of-line mark for the previous line is treated as a character to be transposed.

ESC t

Transposes the words around the point.

SPECIAL: If you are at the beginning of the file, the first two words are transposed. If you are at the end of the file, no transposition is performed.

CTRL x CTRL t

Transposes the lines at and above the cursor.

SPECIAL: If you are at the beginning of the file, the first two lines of the file are transposed.

ESC c

Capitalizes the first character after the point and makes the rest of the word lowercase.

ESC u

Converts the word following the point to UPPERCASE characters.

ESC l

(lowercase letter *l*) Converts the word following the point to lowercase characters.

CTRL x CTRL u

Converts the region you've marked to UPPERCASE.

CTRL x CTRL l

(Lowercase letter *l*) Converts the region you've marked to lowercase.

On Unix or OpenVMS:

- Transpose any two words.
- Transpose any two lines.
- Convert any word to all UPPERCASE.
- Practice changing the case of an entire region:
 - Select the beginning of the region by pressing **CTRL @** or **CTRL SpaceBar** .
 - Move the cursor to define the region.
 - Use **CTRL x CTRL u** to convert the region to UPPERCASE.
 - Use **CTRL x CTRL l** to convert the region to lowercase.

Editing Multiple Buffers

Use the commands below to place a different file into an Emacs buffer for editing, to change buffers, and to kill buffers that you no longer want. All the things that Emacs remembers as you edit (search strings, the circular "list" of deletions, and so on) are the same for all the buffers you are editing. This makes it easy to copy or delete text from one file and insert it into another.

`CTRL X` `CTRL f` *file* `RETURN`

Creates a new buffer and puts *file* (an existing file or a new filename) into it. If you were already editing another file, that file remains available for editing in the old buffer until you kill the buffer or terminate the editing session.

`CTRL X` `CTRL v` *file* `RETURN`

Places *file* (an existing file or a new filename) into the current buffer, replacing the current file. If the file has not been saved since it was last modified, you will be asked to confirm that you want to kill the old buffer anyway.

`CTRL X` `b`*buffername* `RETURN`

Makes *buffername* the current buffer. After you press `CTRL X` `b`, a default buffer name will be displayed in parentheses on the mode line. If you don't specify *buffername*, Emacs makes the buffer named on the mode line the current buffer.

`CTRL X` `k`*buffername* `RETURN`

Kills *buffername*. Default is the current buffer. Emacs prompts you to save any changes that have been made before it kills the buffer.

On the next page, you will learn how to display a list of your buffers.

On Unix or OpenVMS:

Create a new buffer.

- Use `CTRL X` `CTRL f` to call up a new buffer (give it a filename that does not already exist).
- Insert some text into the new buffer.
- Use `CTRL X` `CTRL s` to save your new buffer.

Now move some text from your original buffer to the new one.

- Use the `CTRL X` `b` command to return to the original buffer. (If the correct buffer name appears in parentheses on the echo line, you do not have to type the buffer name. Pressing `RETURN` will display that buffer.)
- Use `CTRL @` (or `CTRL SpaceBar`), cursor motion, and `CTRL X` `CTRL X` to mark and check a paragraph as a text region. Then use `CTRL w` to cut ("wipe") it.
- Return to the new buffer by using the `CTRL X` `b` command again. Specify the *buffername*, if necessary, and press `RETURN` to display that buffer.
- Use `ESC >` to move your cursor to the end of the file, and press `CTRL y` to insert the paragraph from the original file into the new buffer.

Don't practice killing the extra buffer yet. While you still have multiple buffers, use them to perform the exercises on the next page.

Viewing Multiple Buffers

You can divide the current window, allowing you to look at more than one buffer at a time. Some Emacs commands, such as those that display information, also create a second window automatically. Commands used to view more than one window at a time are:

- CTRL X 2** Divides the current window in two horizontally. Both windows will contain the same buffer. The window with the cursor in it is the current window. Use the **CTRL X CTRL f** or **CTRL X b** to change the file in the current window.
- CTRL X o** (Letter *o*) Moves the cursor to the other window.
- CTRL X 1** (Number *1*) Returns the screen to a single window. The window containing the cursor becomes the only window. The other buffers are still available, although not displayed.
- CTRL X 0** (Zero) Closes the current window. The file is still in its buffer, but is not displayed.
- CTRL X 5** Divides the window in two vertically. Some people prefer this to horizontally.
- ESC CTRL V** Scrolls the other window (the one that does not contain the cursor) forward. NOTE: To scroll the other window backward, you must move to that window.
- CTRL X CTRL b**
- Displays a list of all buffers, whether or not they are currently visible in a window. If you have only one window at the time you give this command, Emacs will create a second window automatically to display the information.

If, for some reason, your windows seem frozen or do not close as expected, press **CTRL I** (letter *I*) to refresh the screen.

On Unix and OpenVMS:

Display a list of your buffers:

- Use **CTRL X CTRL b** to list your buffers. You will see the files that you have been editing and possibly some additional buffers created by Emacs.
- Use **CTRL X 1** (one) to make the window containing your text file the only window again.

Create multiple windows and practice moving around in them:

- Use **CTRL X 2** to divide the text window into two windows containing the same buffer.
- Use **CTRL X o** to position your cursor in the other window and then use **CTRL X b** to place the buffer that you cannot see in that window.
- Move back and forth between the windows and scroll each file both directions using **CTRL V** and **ESC V**. Do this until you have the feel of using two windows. Use **ESC CTRL V** at least once.
- Use **CTRL @** (or **CTRL SpaceBar**), cursor motion, **CTRL W**, **CTRL X o** and **CTRL Y** to move some text from one of the files to the other.
- Move to the window containing the new file that you created.
- Use **CTRL X CTRL S** to save the file.
- Use **CTRL X 0** (zero) to delete that window.
- Use **CTRL X CTRL b** to list your buffers again. Notice that the buffer still exists, even though the window it was in was deleted.

Including Files

To place the text of an entire file into another file, you can use one of the commands below:

`CTRL X ifile RETURN`

Inserts the entire contents of *file* into the current file, beginning at the point. A prompt will be displayed in the echo area after you type the character **i**. NOTE: If you type a space instead of *file* after the **i**, you will see a list of possible completions.

`ESC X insert-file RETURN file RETURN`

Same as the above. The prompt will be displayed after you press `RETURN` the first time.

- If you are not already in a single window containing the text of your original practice file, use `CTRL X 1` to reduce the number of windows to one and use the `CTRL X b` command to place the practice file in the buffer.
- Use `CTRL X kbuffername RETURN` to kill the buffer containing the new file that you have created. Now, although the new file exists in your directory, it is not one of your Emacs buffers.
- Type `CTRL X ifile RETURN`. For *file*, substitute the name of the file in the buffer you just killed. The text of that file will be inserted at the current cursor location.

Macros

A macro is a set of commands that Emacs will store and perform repeatedly at your request. You type:

- an opening command (indicating that the commands that follow belong to a macro),
- the command sequence you want in the macro,
- then the closing command (to terminate the macro).

If you plan to make several macros, you must give each macro a name. Otherwise, only the most recently defined macro is retained.

`CTRL X (` Opens the macro (begins recording keystrokes).

`CTRL X)` Closes the macro (terminates recording).

`CTRL X e` Runs the macro that you have just defined. You can use a repeat command with this to perform the macro a number of times.

`ESC X name-last-kbd-macro RETURN macro RETURN`

Prompts you for a name to give the last macro that you defined. If you plan to keep a macro, you must use this command before defining another macro, or your first macro will be lost.

`ESC X macro RETURN`

Runs the macro named *macro*.

`ESC X insert-kbd-macro RETURN`

Inserts the most recent macro definition into a file at the point. Use this command to save the current macro in your **.emacs** file for later use.

NOTE: Because the last command actually changes your **.emacs** initialization file, you will not practice this command in the short course. You can use it to customize your **.emacs** file when you are sure which macros you will use frequently.

On Unix and OpenVMS:

Create a macro that moves to and transposes the first two words of the next line.

- Use `ESC <` to move to the beginning of the file.
- Type `CTRL X (` to open the macro.
- Enter the following commands (which will execute as you type them):

`CTRL n`

`CTRL a`

`ESC f`

`ESC t`

- Type `CTRL X)` to close the macro.
- Type `CTRL X e` to execute the macro.
- Name the macro by typing `ESC X name-last-kbd-macro RETURN macro RETURN`
- Use a repeat command to execute the macro 3 times.

Using Emacs Internal Help

Emacs has an extensive internal help system, which includes everything from a beginner's tutorial to a complete online manual (called the Info system). Almost any command you need will be described *somewhere* in one of these help systems. However, new Emacs users sometimes find it difficult to locate the specific command they want in the volume of information available.

The commands on this page offer some ways to approach using Emacs internal help. The Info system is described on the following page.

[CTRL h] Starts Emacs internal help. You will see a prompt in the echo area. You can type single-character commands to get more information. Some useful characters are:

- ?** Displays a list of the available single-character commands. For a description of these, type **[CTRL h]** again after typing the question mark.
- atopic** (Apropos) Displays a list of commands containing the named topic (*mail*, or *kill*, for example.) in their descriptions.
- i** Places you in the Info system.
- k[command]** Displays the function of the named CTRL or ESC command.
- m** Lists the available commands for the current mode.
- t** Runs a tutorial for new Emacs users.

To return to your editing from a help (or Info) session, use the Emacs commands that you already know. Normally a HELP buffer remains available, so that you can go back to the same place in it later. Notice the current position of your cursor.

- *If your cursor is already back in your editing window*, you are already out of the help system. Use **[CTRL l]** or **[CTRL x] 1** to remove remaining unwanted help displays.
- *If your cursor is in the echo area*, press **[CTRL g]** to return to your editing. You may need to use **[CTRL l]** also to clean up the window.
- *If your cursor is in a special help buffer* (for example, the tutorial buffer) or in a help window (even though your editing window may also be on the screen), use the commands you know for using multiple windows and changing buffers to get back to your editing. **[CTRL x] o**, **[CTRL x] 1**, and **[CTRL x] b** will all be useful.

On Unix and OpenVMS: Try using Emacs internal help.

- Press **[CTRL h]** and notice the line in the echo area.
- Type a question mark (?). Notice the display of single-character commands in the echo area.
- Press **[CTRL h]** again for more information. Read through the descriptions there.
- Try using some of the single-character commands to see the definitions of commands you know or want to know. (Don't go into the Info system yet. Save that for the next page.)
- Once you understand how this type of help works, determine the best way to exit from your current location in the internal help and return to your editing. Then move on to the next page to learn about Info. This help system is always available to you when you are using Emacs, so you can come back to it any time.

Using the Info System

The Info system is an extensive online manual of help information about Emacs. Info uses some commands that are different from regular Emacs commands, and it has a tutorial with its own set of instructions.

`[ESC X] info [RETURN]`

Enters the Info system. You will see brief instructions and a list of menus. Useful Info commands are:

- ?** Displays the list of Info commands you can use.
- d** Returns to the top-level Info **directory** from anywhere in the Info system.
- m** Prompts you for the name of a **menu**. Type a menu name and press the `[RETURN]` key. If you position the cursor on a menu name, it becomes the default, and you can simply press `[RETURN]` to see that information.
- n** Displays the **next** node (topic of information) at the current level.
- p** Displays the **previous** node at the current level.
- q** Exits from (**quits**) the Info system and returns you to your editing.
- u** (**Up**) Displays the node on the level above the current nodes.

NOTE: A line at the top of each Info node displays information about that node, the next node, the previous node, etc. If you forget where you are in the Info system, use that information to help determine how to proceed.

On Unix and OpenVMS:

One interesting item not discussed in this course is key binding--the ability to assign a key sequence to represent an entire macro. To practice using the Info system, look up some information on key binding.

- Use `[ESC X] info [RETURN]` to enter the Info system. Use `[CTRL V]` and `[ESC V]` to scroll through the list and see the menus that are available. Find the menu named *Emacs*. This is the online Emacs manual.
- Position the cursor on the Emacs menu name. Type an **m** and notice that the default menu name in parentheses is the one on which your cursor is positioned. Press `[RETURN]` to go to that menu. (Alternatively, you could have simply typed **memacs** to do the same thing.)
- You will see another list of menus. Look at the menus available and find the one named *Customization*. Type the Info command below to look at information on customizing Emacs to your needs.

`mcustomization [RETURN]`

If you pause after typing the **m**, you will see the prompt and you can type the remainder of the command (the menu name) at the prompt.

- Use the **m** and **u** commands to look at the following menus: Init File and Key Bindings. Under the Key Bindings menu, look at the Rebinding menu also.
- If you have time, explore any other menus that look interesting.
- When you are finished, use the **d** command to return to the top directory and the **q** command to exit from the Info system. If that doesn't return you to your editing buffer, notice the location of your cursor and use the guidelines on the previous page to return to your editing.

Additional Information

The following documents or manuals are especially useful for Emacs users:

GNU Emacs Reference List (CCRL-34). ACITS, University of Texas at Austin.

Available online at <http://www.utexas.edu/cc/docs/ccrl34.html>.

Stallman, Richard.

GNU Emacs Manual. Thirteenth Edition, for Emacs Version 20.3. Free Software Foundation, 1998. <http://www.gnu.org/manual/emacs-20.3/emacs.html>.

The original comprehensive Emacs reference manual. Not a tutorial.

Lewis, Bill, Dan LaLiberte, and the GNU Manual Group.

The GNU Emacs Lisp Reference Manual. For Emacs Version 20.3. Free Software Foundation, 1996.

http://www.delorie.com/gnu/docs/elisp-manual-20/elisp_toc.html.

Reference manual for Lisp programmers who write functions to customize Emacs.

The resources below will be useful for Unix users:

O'Reilly & Associates.

Learning GNU Emacs, Second Edition (1996).

See <http://www.oreilly.com/catalog/gnu2/>.

GNU Emacs Pocket Reference (1998).

See <http://www.oreilly.com/catalog/gnupr/>.

Unix in a Nutshell: System V Edition, Third Edition (1999).

See <http://www.oreilly.com/catalog/unixnut3/>.

ZDNet.

Master Emacs Text Editing in Linux. (2001)

See <http://www.zdnet.com/zdhelp/stories/main/0,5594,2682833-1,00.html>

Additional Practice

- Examine the GNU Emacs Pocket Reference List you were given at the start of class.
- Find the commands that you already know, so that you can locate them quickly if you forget the command syntax.
- Experiment with some of the commands you have not yet tried.
- Explore the internal help and the Info system further.
- If you have any questions about Emacs, ask the short course instructor⁴.

TURN TO THE NEXT PAGE FOR INSTRUCTIONS ON LOGGING OUT.

⁴ If you are taking the course on your own and you have questions, send E-mail to help@cc.utexas.edu. You will get a response within a few days. If you need immediate assistance (for example, because you cannot log out), call the Help Desk at 475-9400. If the problem occurs after the Help Desk is closed, call 475-9300 and talk to the operator on duty.

Logging Out

**When you are finished with the short course,
be sure to exit from GNU Emacs, log out from the system, and
close your network connections.**

On Unix and OpenVMS:

- Exit from Emacs by typing:

`CTRL X` `CTRL C`

- To log out, type at the % or \$ prompt:

logout

- Close any network connections you may be using (CCNET, Telnet, etc.) with the appropriate **quit** or **exit** command.
- If necessary, log out from the workstation you are using (for example, a workstation in the Student Microcomputer Facility).

When you have finished, please complete the evaluation form at the end of the handout and give it to the instructor or mail it to ACITS through campus mail or U.S. Mail.

Short Course Administration
Academic Computing and Instructional Technology Services
G2700

CAMPUS MAIL

Place
Stamp
Here

Short Course Administration
ACITS G2700
The University of Texas at Austin
Austin, TX 78712-1110

Instructor's Information

(Introduction to GNU Emacs)

- 1. Things to Do before Class**
- 2. Things to Bring to Class**
- 3. Things to Mention during Class**
- 4. Things to Do after Class**

I. Things to do before class:

1. **Verify login procedures.** Be sure you know how to log in on both Unix and VMS from the room where the course is being taught. Labscripts do not include login instructions.
2. **Get Unix staff to validate user numbers and change passwords 4 days in advance.** Unix short course numbers (usually a CZVL or CZAL stem), do not remain validated. Validation requires 48 hours and cannot be done by hand — so if validation fails, it takes another 48 hours. Also, password changes on Unix systems do not take effect for about 30 minutes. The ACITS Accounting Office cannot do the validation.
3. **Get OpenVMS staff to set up user numbers and change the password.** They need to run STAFF_UTIL:SC_INIT.COM before the course to set up the user numbers and change the password so that users don't have to do that the first time they log in.
4. **Verify short-course files.** Be sure that the **.emacs** and **emacs.tasks** short-course files in /usr/local/lib/shortcourse/630 and SC:[730] are readable by the user.
5. **Arrange for copying of this labscript and any other handouts.** Call the ACITS mailroom a day before you are ready to teach and be sure copies of the labscript are being made from the most recent labscript master for the course.
6. **Disable control keys.** You (or each user) will need to disable the CTRL-S, CTRL-Q, and CTRL-C characters before using Emacs. If the course is taught in the SMF, this is done using Telnet.* After you open a Telnet connection to a system, select the Telnet **Session** menu, then select **Setup Keys**. Delete the entries for ^S, ^Q, and ^C, leave the boxes blank, and click OK. Be sure that the users understand that these control keys have to be disabled, but that the procedure may be different depending on how they connect (e.g., by Telnet or other means) or where they connect from (e.g., a terminal somewhere else).

*NOTE: If the course is taught in the SMF, tell people to use CTRL SpaceBar instead of CTRL @ to mark a region, since Telnet and the Macintoshes cause CTRL @ not to work.

II. Things to bring to class:

Required:

1. **Introduction to GNU Emacs** (CCTUV-03). <http://www.utexas.edu/cc/docs/cctuv03.html> (36 pp.)
Call the ACITS mailroom (COM 1) at least a day ahead of time to order a sufficient number of copies of the labscript. You are responsible for getting them from the mailroom personally or for making other arrangements to have them in your possession before class.

Optional:

2. **GNU Emacs Reference List** (CCRL-34). <http://www.utexas.edu/cc/docs/ccr134.html> (16 pp.)
You can request that the mailroom staff make copies of the GNU Emacs Reference List, too; however, the document is easier to use online than it is in paper form. You may prefer merely to post the URL in the classroom and draw students' attention to it.

3. **Unix Pocket Reference List** (CCRL-20). <http://www.utexas.edu/cc/docs/ccr120.html> (13 pp.)
You can request copies of this list, too, or you may prefer to just post the URL in the classroom.

4. **VMS Pocket Reference List** (CCRL-21). <http://www.utexas.edu/cc/docs/ccr121.html> (9 pp.)
You can request copies of this list, too, or you may prefer to just post the URL in the classroom.

III. Things to mention in lecture:

1. **Emacs on OpenVMS systems.** OpenVMS systems that offer Emacs include the general computing nodes of the ACITS VMS System (see UTHELP CONFIGURATION for a current list). For our purposes, the terms VMS and OpenVMS are synonymous.
2. **Emacs on Unix systems.** Unix systems that offer Emacs are **uts** and **ccwf**.
3. **Password.** Normally the very first time you log in on a new user number, you will be required to change your password. The user numbers in this short course have been specially prepared so that you do not have to do that today.
4. **Footnotes.** Short course participants can disregard the footnotes in this handout. They are for people who are using the handout in an independent self-paced course of study.
5. **Help.** Towards the end of the class time, encourage participants who are clearly not going to finish the script to look at the help-system pages near the end of the script.
6. `CTRL @` If the course is taught in the SMF, tell people to use `CTRL SpaceBar` instead of `CTRL @` to mark a region, since Telnet and the Macintoshes cause `CTRL @` not to work.

IV. Things to do after class:

- Be sure that all users logged out when they finished.