View the Lesson (Using the Command Line to Get Help)

Man Pages

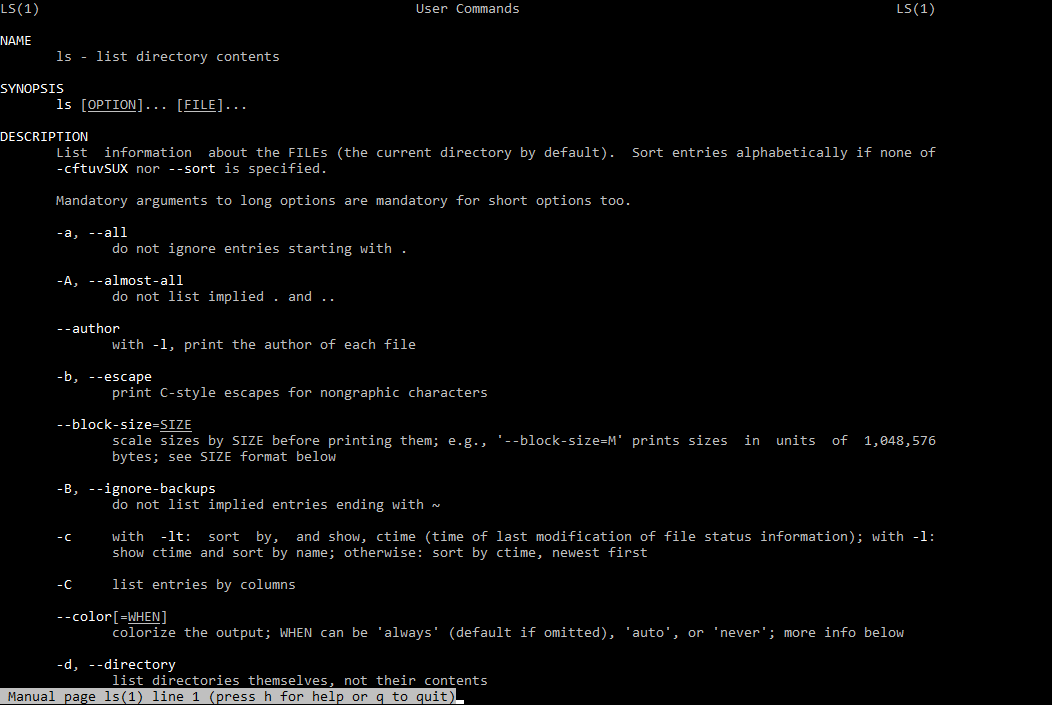
* A man page (short for manual page) is a form of software documentation usually found on a Unix or Unix-like operating system.
* Man pages are the traditional package documentation for application usage. What that means is that they're typically installed when you install a package. So if we install a package to do some task, the man page for that package will typically be installed at the same time. This gives us the ability to take a look at that documentation and make sure that we're using it in a manner consistent with its design.

Usage

The man page for a particular command is invoked by preceding the command with man.

man <command>

man ls



NAME

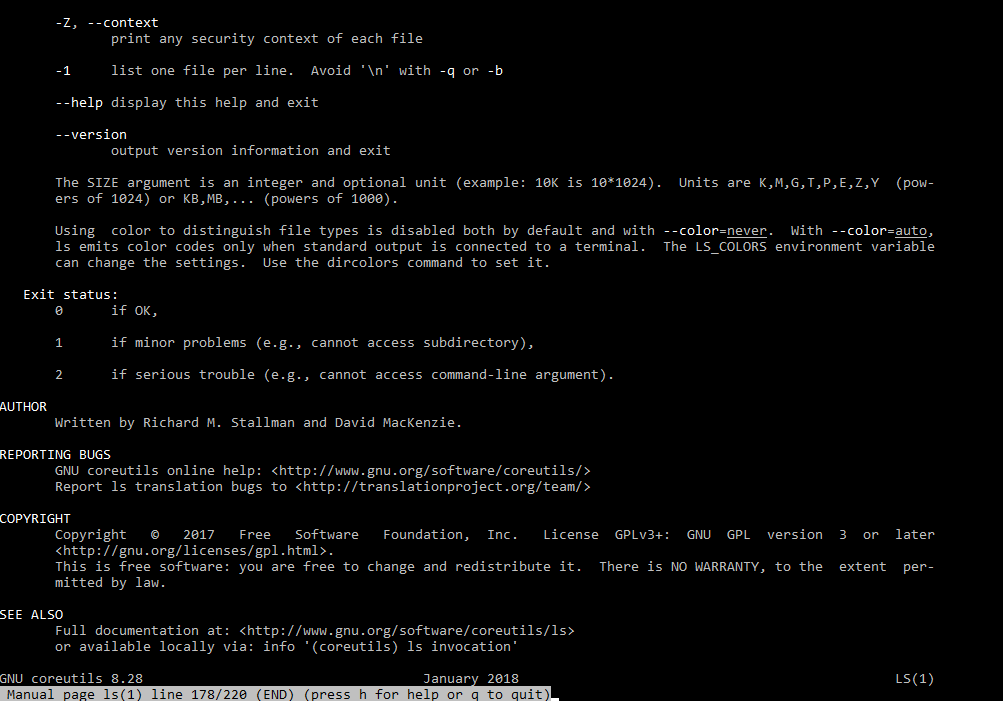
Program or Function name(s) followed by descriptions of functionality.

SYNOPSIS

A short overview of available options

DESCRIPTION

Detailed information about arguments and options.



## View the Lesson (Using the Command Line to Get Help)

### Info Pages

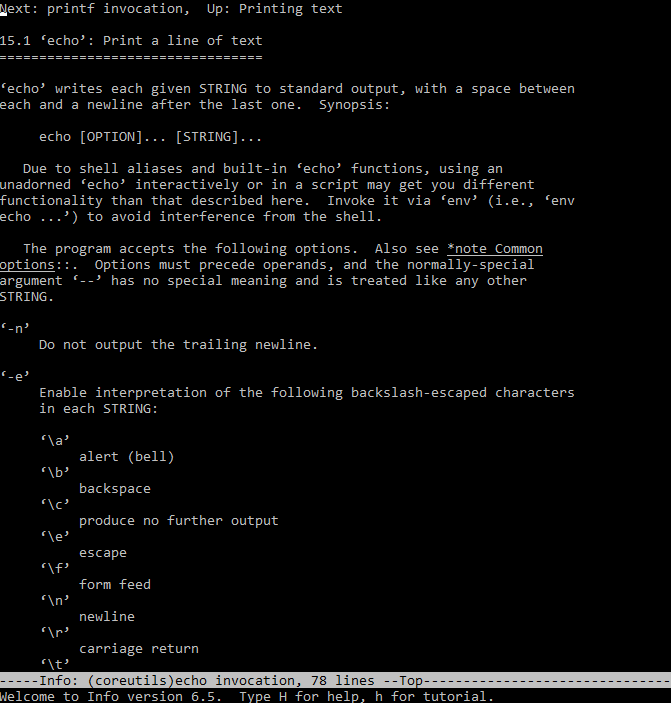
Info pages are additional documentation with more robust capability in detail. Info Page normally provides more detailed information about a command than its respective man page. Additionally, Info uses a structure for linking these pages together, and they may be assembled into a larger collection.

### Usage

The info page for a particular command is invoked by preceding the command with info.

info <command>

info echo



The main difference between Man and Info is the amount of content that they have; Info contains a whole lot more than Man does.

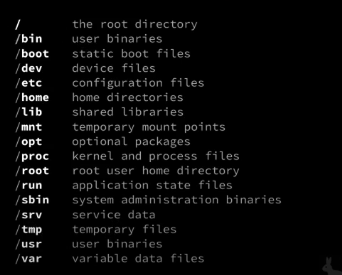
If no info page exists, info can pull documentation from the man page.

View the Lesson (Using Directories and Listing Files)

Files and Directories

The file system hierarchy standard (FHS) defines the structure of the file systems on Linux.

* In the FHS, all files and directories appear under the root directory / , even if they are stored on different physical or virtual devices.
* Most of these directories exist in all UNIX operating systems and are generally used in much the same way; however, the descriptions here are those used specifically for the FHS, and are not considered authoritative for platforms other than Linux.

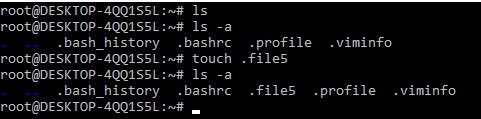


1. **/ (Root) :** Primary hierarchy root and root directory of the entire file system hierarchy. Every single file and directory starts from the root directory.
2. **/bin :** Essential command binaries that need to be available in single user mode.
3. **/boot :** Boot loader files.
4. **/dev :** Essential device files.
5. **/etc :** Host-specific system-wide configuration files.
6. **/home :** Users’ home directories, containing saved files. Home directories for all users to store their personal files
7. **/lib :** Libraries essential for the binaries in /bin/ and /sbin/
8. **/mnt :** Temporarily mounted filesystems.
9. **/opt :** Optional application software packages.
10. **/proc :** Contains information about system process.
11. **/sbin :** Essential system binaries, e.g., fsck, init, route.
12. **/srv :** Site-specific data served by this system, such as data and scripts for web servers, data offered by FTP servers, and repositories for version control systems. srv stands for service.
13. **/tmp :** Temporary files. Often not preserved between system reboots, and may be severely size restricted.
14. **/usr :** Secondary hierarchy for read-only user data; contains the majority of (multi-)user utilities and applications.

## View the Lesson (Using Directories and Listing Files)

### Hidden Files and Directories

Files and directories can be hidden from the basic listing. Files and directories in Linux may be set as hidden by preceding their name with a . (a single period).



Hidden files can be listed via ls using the -a option (meaning show all files).

### Home Directories

User home directories contain files specific to users. A directory under /home is typically created for every ordinary user on the Linux system.

cd (change directory) without a specified path

~ used to represent the user's home directory

$HOME environment variable used to store home directory path

To navigate to your home directory, use cd or cd ~

### Absolute and Relative Paths

The path is the unique location of a file or directory.

The path to a file or directory location can be specified as an absolute path (starts with a /), or a relative path. Below is the pwd command show the absolute path of current working directory



[cloud\_user@ip-10-0-1-10 ~]$ pwd

/home/cloud\_user

### Relative Path

Below, the cat command is used to show the content of file1 specified with a relative path.



[cloud\_user@ip-10-0-1-10 ~]$ cat file1

Good Day

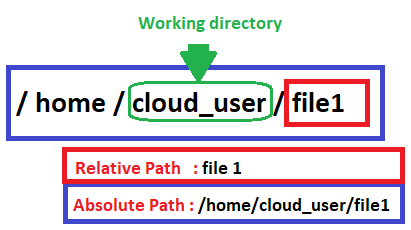
### Absolute Path

Below, the cat command is used to show the content of same file1 specified with a absolute path.



[cloud\_user@ip-10-0-1-10 ~]$ cat /home/cloud\_user/file1

Good Day



## View the Lesson (Creating, Moving, and Deleting Files)

### Creating, Moving, Deleting

Creating, moving, and deleting files and directories.

touch - create a file

rm - delete the file

cp - used to copy file or folder

mv - used to move file or folder

mkdir - create a folder

rmdir - delete folder

### Working with directories

Create a new directory mkdir <NAME>

Copy a directory cp -r <SOURCE> <DESTINATION>

Move a directory mv <SOURCE> <DESTINATION>

Delete a directory rm -r <DIRECTORY>

Delete an empty directory rmdir <DIRECTORY>

### Working with files

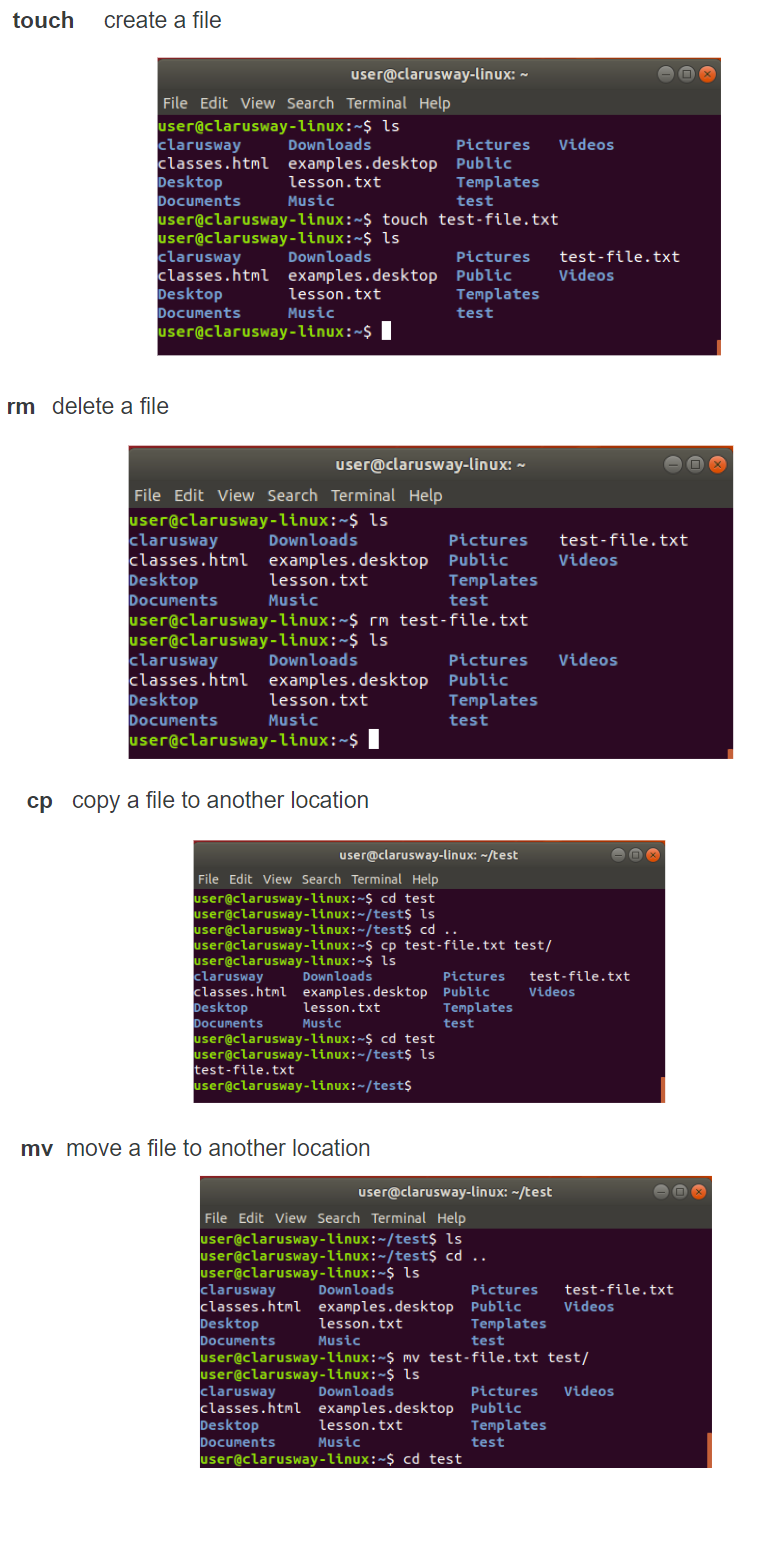
Create a new file touch filename

Removing Files rm -option filename

**Option** -r includes the contents of a directory and the contents of all subdirectories when you remove a directory.

**Option** -i prevents the accidental removal of existing files or directories.

**Example:**



## View the Lesson (Creating, Moving, and Deleting Files)

Case Sensitivity

Most of the common Linux file systems are case sensitive; this is something to keep in mind when creating and moving directories and files.

**Using Case Sensitivity**

Lower-case and upper-case letters have different ASCII representation.

Example;

touch newfile

touch Newfile

The commands above will create two different files.

View the Lesson (Creating, Moving, and Deleting Files)

Simple Globbing

Globbing is primarily used to match patterns in filenames or text by using a wildcard character to create the pattern.

? (Question mark) : Match any single character

\* (Asterisk) : Match any number of character(s)

[] (Brackets) : Match character from a range

^ (Caret) : Used to match starting character

$ (Dollar sign) : Used to match ending character

{} (Curly brace) : Used to match more than one pattern

| Pipe : Used for applying more than one condition

Example :



clarusway@f85a0c1549f4:~$ ls

all file1 file2 file3 file4 happiness loneliness

    reverse

clarusway@f85a0c1549f4:~$ ls file?

file1 file2 file3 file4

clarusway@f85a0c1549f4:~$ ls ?????

file1 file2 file3 file4

clarusway@f85a0c1549f4:~$ ls ????1

file1

clarusway@f85a0c1549f4:~$ ls file\*

file1 file2 file3 file4

clarusway@f85a0c1549f4:~$ ls \*

all file1 file2 file3 file4 happiness loneliness

    reverse

clarusway@f85a0c1549f4:~$ ls \*ness

happiness loneliness

clarusway@f85a0c1549f4:~$ ls \*[1-4]

file1 file2 file3 file4

clarusway@f85a0c1549f4:~$ ls \*[2-3]

file2 file3

clarusway@f85a0c1549f4:~$ ls \*[[:digit:]]

file1 file2 file3 file4

* [[:upper:]] or [[A-Z]] - match upper-case character
* [[:lower:]] or [[a-z]] - match lower-case character
* [[:digit:]] or [[0-9]] - match digits
* [[:alpha:]] or [[a-zA-Z]] - match either case character
* [[:alphanum:]] or [[a-zA-Z0-9]] - match alphanumeric

## View the Lesson (Manipulating Text Files with Text Editors)

### Why should I use an editor?

It is very important to be able to use at least one text mode editor. The first step toward independence is learning how to use an editor on your system.

We need to master an editor because we need it to edit files that affect our environment. You might want to start writing scripts, or books, creating websites or new programs as an advanced user. Mastering an editor will boost your productivity and your skills tremendously.

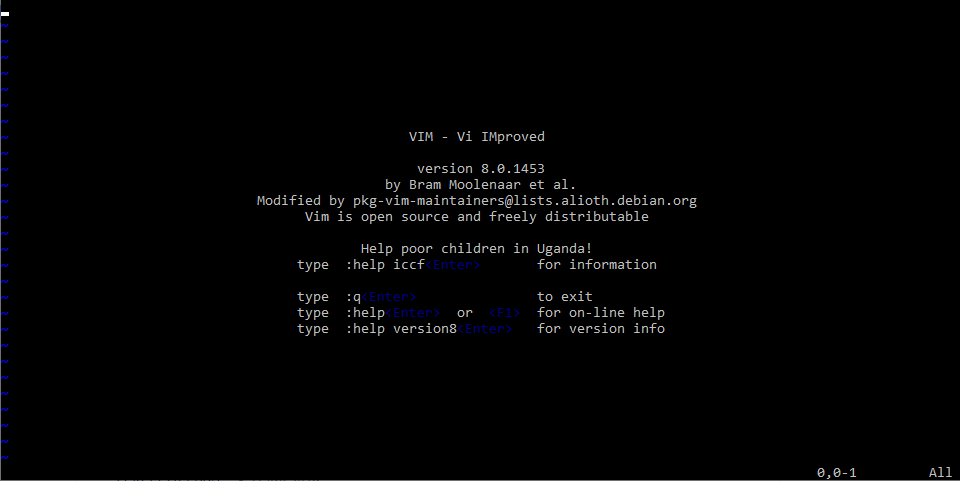
### Which editor should I use?

Our focus for text editors is on the ones which can also be used in terminal windows and in systems without a graphical environment. The further advantage of mastering a text editor is its use on remote computers. Because you don't need to move the whole graphical environment over the network, working with text editors greatly enhances network speed.

### Using the Vim editor

## What’s Vim:

Vim stands for “Vi Improved”. Vim is a highly configurable text editor for efficiently creating and changing any kind of text. It is included as "vi" with most UNIX systems and with Apple OS X.



Vim can operate in two modes: command mode and insert mode.

The editor always starts in command mode. When you run vim filename command to edit a file, Vim starts out in command mode. It means that, instead of adding those characters, all alphanumeric keys are bound to commands. For example, Typing h won’t insert the character “h”, it will move the cursor to the left.

To enter the insert mode, type i for (“insert”) and now the keys will act the way you would expect. In the insert mode, you can type, move through the text, search, replace, mark blocks and perform other editing tasks. After pressing Escape, you are back in command mode.

## Basic commands

## Commands that switch the editor to insert mode

| **Vim Command** | **Description** |
| --- | --- |
| a | append: it moves the cursor one position to the right before switching to insert mode |
| i | insert |
| o | insert a blank line under the current cursor position and move the cursor to that line. |

### Moving through the text

Moving through the text is usually possible with the arrow keys. If not, use:

| **Vim Command** | **Description** |
| --- | --- |
| h | move the cursor one character to the left. |
| j | move the cursor down one line. |
| k | move the cursor up one line. |
| l | move the cursor one character to the right. |
| 0 | move the cursor to the beginning of the line. |
| $ | move the cursor to the end of the line. |
| w | move forward one word. |
| b | move backward one word. |
| G | move to the end of the file. |
| gg | move to the beginning of the file. |

### Basic operations

| **Vim Command** | **Description** |
| --- | --- |
| ndd | delete n lines starting from the current cursor position. |
| ndw | delete n words at the right side of the cursor. |
| x | delete the character on which the cursor is positioned |
| :n | move to line n of the file. |
| u | undo the last operation. |
| Ctrl-r | redo the last undo. |
| :w | save (write) the file |
| :q | exit the editor. |
| :q! | force the exit when you want to quit a file containing unsaved changes. |
| :wq | save and exit |
| :w newfile | will save the text to newfile. |
| :wq! | override read-only permission (if you have the permission to override permissions, for instance when you are using the root account. |
| :recover | will recover a file after an unexpected interruption. |

### Searching And Replacing

| **Vim Command** | **Description** |
| --- | --- |
| /text | search the text in the file and position the cursor on the first match below its position. |
| n | move the cursor to the next instance of the text from the last search. This will wrap to the beginning of the document. |
| N | move the cursor to the previous instance of the text from the last search. |
| ?text | search for text in the document, going backward. |
| :%s/text/replacement text/g | search through the entire document for text and replace it with the replacement text. |
| :%s/text/replacement text/gc | search through the entire document and confirm before replacing text. |

## Copying And Pasting

| **Vim Command** | **Description** |
| --- | --- |
| v | highlight one character at a time. |
| V | highlight one line at a time. |
| Ctrl-v | highlight by columns. |
| y | yank text into the copy buffer. |
| p | paste text after the current line. |
| P | paste text on the current line. |
| np | paste it n times |

## View the Lesson (Manipulating Text Files with Text Editors)

### Using the nano editor

## What’s nano:

GNU nano is a small and friendly text editor. Besides basic text editing, nano offers features like undo/redo, syntax coloring, interactive search-and-replace, auto-indentation, line numbers, word completion, file locking, backup files, and internationalization support.

## Opening and Creating Files

## To open an existing file or to create a new file, type nano followed by the file name:

nano filename

## nano screen

The default screen of nano consists of five areas. From top to bottom these are the title bar, a blank line, the edit window, the status bar, and two help lines.

**The title bar** consists of three sections: left, center and right. The section on the left displays the version of nano being used. The center section displays the current filename, or "New Buffer" if the file has not yet been named. The section on the right displays "Modified" if the file has been modified since it was last saved or opened.

**The status bar** is the third line from the bottom of the screen. It shows important and informational messages.

**The two help lines** at the bottom of the screen show some of the most essential functions of the editor. These two lines are called the Shortcut List.

## Editing Files

Unlike vi, nano is a modeless editor, which means that you can start typing and editing the text immediately after opening the file.

## Shortcut List

In both the toolbar and in the help text within the editor, key combos appear with standard Linux shortcut codes. All commands are prefixed with either ^ or M character. The caret symbol (^) represents the Ctrl key. For example, the ^G commands mean to press the Ctrl and G keys at the same time. The letter M represents the Alt key.

| **Command** | **Meaning** | **Description** |
| --- | --- | --- |
| Ctrl G | Get Help | display a complete list of commands and associated hotkeys. |
| Ctrl X | Exit | Close the program. You'll be prompted to save the current buffer or to exit without saving. |
| Ctrl O | Write Out | Save the current buffer to the filesystem. |
| Ctrl R | Read File | Open a file from the filesystem. |
| Ctrl W | Where Is | Search for a text string. |
| Ctrl \ | Replace | Replace one text string for another. |
| Ctrl K | Cut Text | Remove the current line of text. |
| Ctrl U | Uncut Text | Uncut from the cutbuffer into the current line |
| Ctrl J | Justify | Justify the current paragraph |
| Ctrl T | To Spell | Spell-check the current buffer. |
| Ctrl C | Cur Pos | Display the position of the cursor relative to the contents of the buffer as a whole. |
| Alt U | Undo | Undo the last operation |
| Alt E | Redo | Redo the last undone operation |
| Alt M | Mark Text | Mark text starting from the cursor position |
| Alt 6 | Copy Text | Copy the current line and store it in the cutbuffer |