

# **Delhi Technical Campus, Greater Noida**

(Affiliated to GGSIPU, New Delhi)

## **Lab Manual**

Linux Environment

(Bachelors of Computer Application)



Affiliated to GGSIP University, New Delhi  
Approved by AICTE & Council of Architecture

## **Department of Computer Science and Engineering**

**Submitted to:** Dr. Nadeem Malik  
(Assoc. Professor)

**Submitted by:** Kartikey Raghuvanshi  
00718002020  
BCA 3<sup>rd</sup> Year

<b>INDEX</b>		
<b>S.No.</b>	<b>Particulars</b>	<b>Pg No.</b>
1	Demonstrate use of help, what is, man, info and pwd.	1-3
2	Demonstrate the use of whoami with various options wherever applicable.	4
3	Demonstrate the use of mkdir, rm, mv, cp, cmd with various option.	5-7
4	Demonstrate the use of which, where is, locate cmd.	8-9
5	Demonstrate the use of cat cmd in three ways - creating a file, display file content and appending a file (concatenation)	10
6	Demonstrate the use of wc, grep.	11-12
7	Demonstrate the use of pipes in at least 5 commands.	13

**Teacher's Signature:**

## Program – 01

**Ques 01:** Demonstrate use of help, what is, man, info and pwd.

i) **help** – It displays information about built-in commands.

**Syntax:** help [option]... [pattern]...

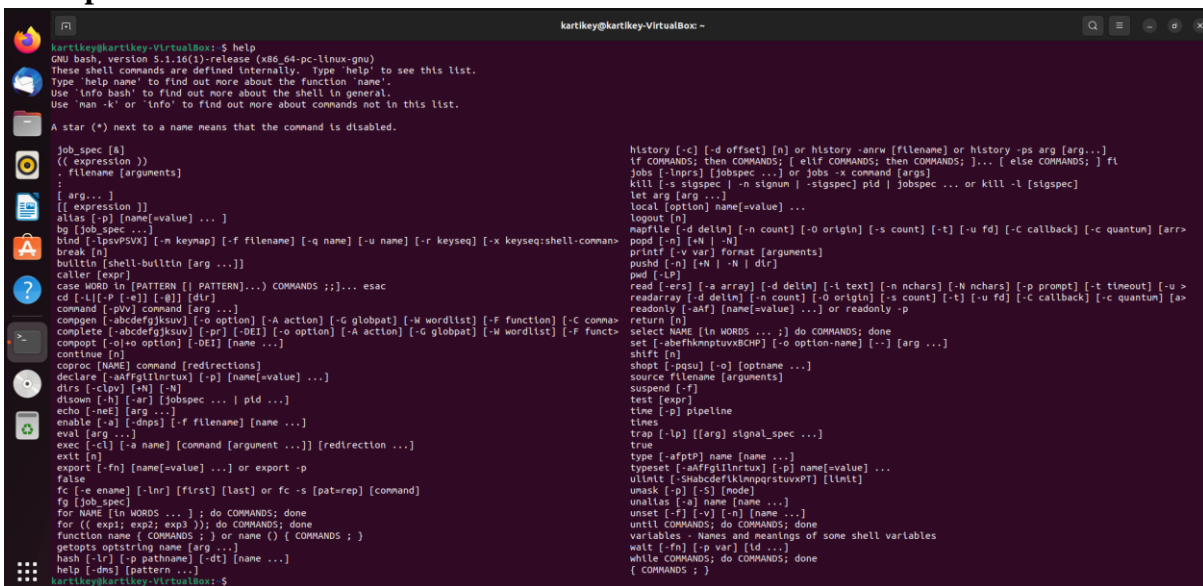
**Various options:**

help-d: It displays the short description for each topic.

help-m: It displays usage in pseudo manpage format.

help-s: It shows output only a short synopsis.

**Example:**



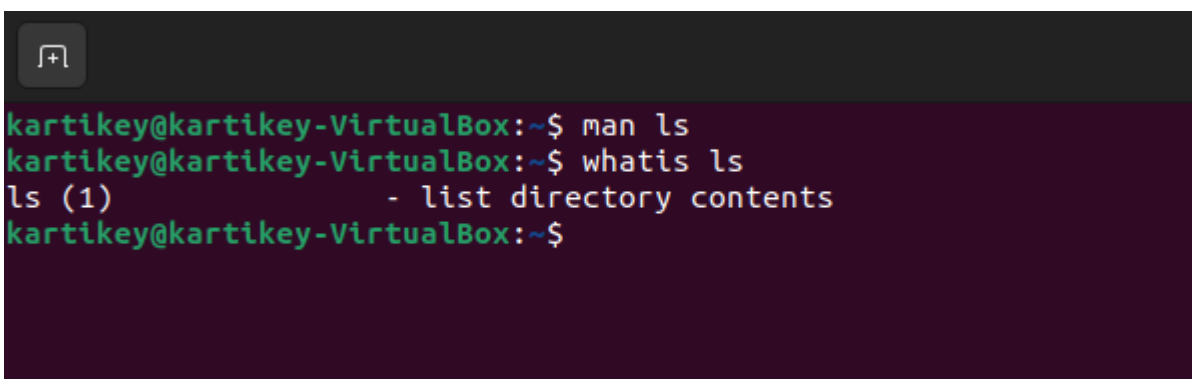
```
kartikey@kartikey-VirtualBox:~$ help
GNU bash, version 5.1.16(1)-release (x86_64-pc-linux-gnu)
These shell commands are defined internally.  Type 'help' to see this list.
Type 'help name' to find out more about the function 'name'.
Use 'info bash' to find out more about the shell in general.
Use 'man -k' or 'info' to find out more about commands not in this list.

A star (*) next to a name means that the command is disabled.

job_spec [a]
{([ expression ])
  . filename [arguments]
  :
  [ arg... ]
  {([ expression ])
  alias [-p] [name=value] ... ]
  bg [job_spec ...]
  bind [-lpsvPSVA] [-n keymap] [-f filename] [-q name] [-r keyseq] [-x keyseq:shell-command]
  break [n]
  builtin [shell-builtin [arg ...]]
  caller [expr]
  case WORD in [PATTERN] ... ) COMMANDS ;; ... esac
  cd [-L|-P [-e]] [-@] [dir]
  command [-p] command [arg ...]
  compgen [-abcefgjkquv] [-o option] [-A action] [-G globpat] [-W wordlist] [-F function] [-C command]
  complete [-abcefgjkquv] [-pr] [-DEI] [-o option] [-A action] [-G globpat] [-W wordlist] [-F function]
  compopt [-o+o option] [-DEI] [name ...]
  continue [n]
  coproc [NAME] command [redirections]
  declare [-aAfFgIlntux] [-p] [name=value] ...
  dirs [-c] [-p] [-n] [-n]
  disown [-h] [-ar] [jobspec ...] pld ...
  echo [-neE] [arg ...]
  enable [-a] [-dnps] [-f filename] [name ...]
  eval [arg ...]
  exec [-cl] [-a name] [command [argument ...]] [redirection ...]
  exit [n]
  export [-fn] [name=value] ... or export -p
  false
  fc [-e ename] [-lnr] [first] [last] or fc -s [pat=rep] [command]
  fg [job_spec]
  for NAME [in WORDS ... ] ; do COMMANDS; done
  for (( exp1; exp2; exp3 )); do COMMANDS; done
  function name { COMMANDS ; } or name () { COMMANDS ; }
  getopts optstring name [arg ...]
  hash [-lr] [-p pathname] [-dt] [name ...]
  help [-dns] [pattern ...]
  history [-c] [-d offset] [n] or history -anrw [filename] or history -ps arg [arg...]
  if COMMANDS; then COMMANDS; [ elif COMMANDS; then COMMANDS; ]... [ else COMMANDS; ] fi
  jobs [-lnprs] [jobspec ...] or jobs -x command [args]
  kill [-s sigspec | -n signum | -t sigspec] pld | jobspec ... or kill -l [sigspec]
  let arg [arg ...]
  local (option) name[=value] ...
  logout [n]
  mapfile [-d delin] [-n count] [-O origin] [-s count] [-t] [-u fd] [-C callback] [-c quantum] [arr]
  popd [-n] [+N | -N]
  printf [-v var] format [arguments]
  pushd [-n] [+N | -N | dir]
  pwd [-LP]
  read [-ers] [-a array] [-d delin] [-t text] [-n nchars] [-N nchars] [-p prompt] [-t timeout] [-u >
  readarray [-d delin] [-n count] [-O origin] [-s count] [-t] [-u fd] [-C callback] [-c quantum] [a]
  readonly [-aF] [name=value] ... or readonly -p
  return [n]
  select NAME [in WORDS ... ] ; do COMMANDS; done
  set [-abefhknptuvxBCHP] [-o option-name] [--] [arg ...]
  shift [n]
  shopt [-psu] [-o] [optname ...]
  source filename [arguments]
  suspend [-f]
  test [expr]
  time [-p] pipeline
  times
  trap [-lp] [[arg] signal_spec ...]
  true
  type [-afpt] name [name ...]
  typeset [-aAfFgIlntux] [-p] name[=value] ...
  ulimit [-ShabcefklnmpqrstuvPT] [limit]
  unset [-p] [-S] [mode]
  unset [-a] name [name ...]
  unset [-f] [-v] [-n] [name ...]
  until COMMANDS; do COMMANDS; done
  variables - Names and meanings of some shell variables
  wait [-fn] [-p var] [id ...]
  while COMMANDS; do COMMANDS; done
  { COMMANDS ; }
```

ii) **whatis** - whatis command in Linux is used to get a one-line manual page description.

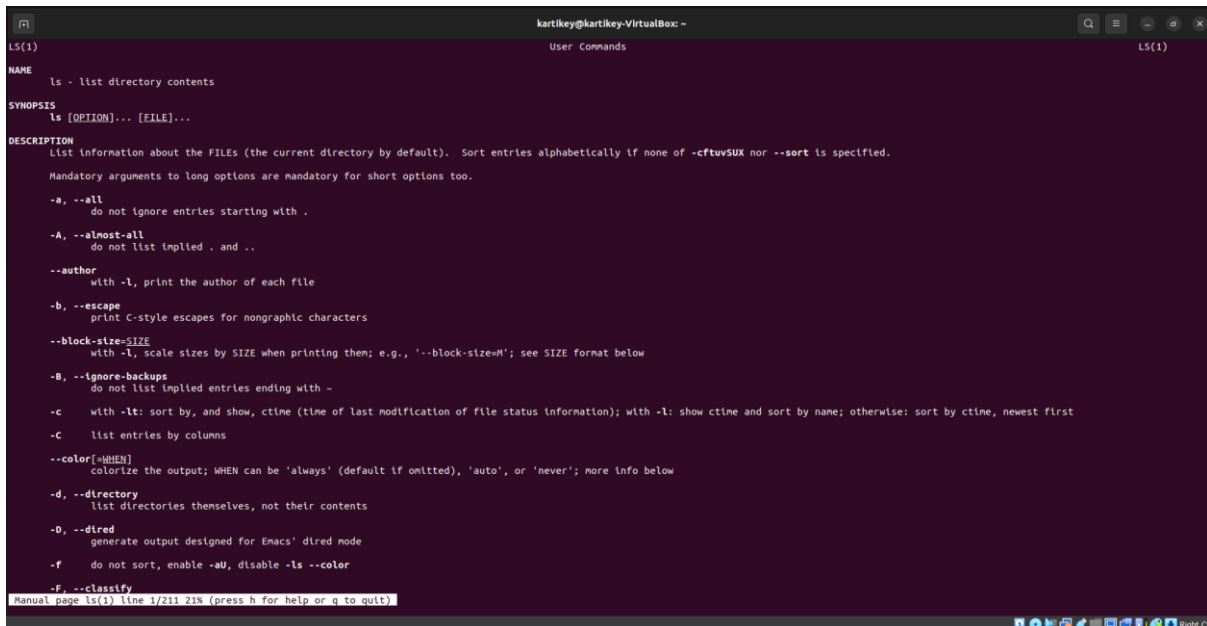
**Syntax:** whatis [option]... [command\_name]...



```
kartikey@kartikey-VirtualBox:~$ man ls
kartikey@kartikey-VirtualBox:~$ whatis ls
ls (1) - list directory contents
kartikey@kartikey-VirtualBox:~$
```

iii) **man** - man command in Linux is used to display the user manual of any command that we can run on the terminal. It provides a detailed view of the command.

**Syntax:** \$man [option]... [command\_name]...



```
kartikey@kartikey-VirtualBox: ~
LS(1)
User Commands
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 -cftuvSUX nor --sort is specified.
  Mandatory arguments to long options are mandatory for short options too.

  -a, --all
      do not ignore entries starting with .
  -A, --almost-all
      do not list implied . and ..
  --author
      with -l, print the author of each file
  -b, --escape
      print C-style escapes for nongraphic characters
  --block-size=SIZE
      with -l, scale sizes by SIZE when printing them; e.g., '--block-size=M'; see SIZE format below
  -B, --ignore-backups
      do not list implied entries ending with ~
  -c
      with -lt: sort by, and show, ctime (time of last modification of file status information); with -l: show ctime and sort by name; otherwise: sort by ctime, newest first
  -C
      list entries by columns
  --color[=WHEN]
      colorize the output; WHEN can be 'always' (default if omitted), 'auto', or 'never'; more info below
  -d, --directory
      list directories themselves, not their contents
  -D, --dired
      generate output designed for Emacs' dired mode
  -f
      do not sort, enable -aU, disable -ls --color
  -F, --classify
      Manual page ls(1) line 1/211 21% (press h for help or q to quit)
```

iv) **info** – info command reads documentation in the info format. It will give detailed information for a command when compared with the man page.

**Syntax:** info [option]... [menu-item]...



```
kartikey@kartikey-VirtualBox: ~$ info ls

Next: dir invocation, Up: Directory listing

10.1 '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.

Because 'ls' is such a fundamental program, it has accumulated many
options over the years. They are described in the subsections below;
within each section, options are listed alphabetically (ignoring case).
The division of options into the subsections is not absolute, since some
options affect more than one aspect of 'ls''s operation.

Exit status:

 0 success
 1 minor problems (e.g., failure to access a file or directory not
specified as a command line argument. This happens when listing a
directory in which entries are actively being removed or renamed.)
 2 serious trouble (e.g., memory exhausted, invalid option, failure
to access a file or directory specified as a command line argument
or a directory loop)

Also see *note Common options::.

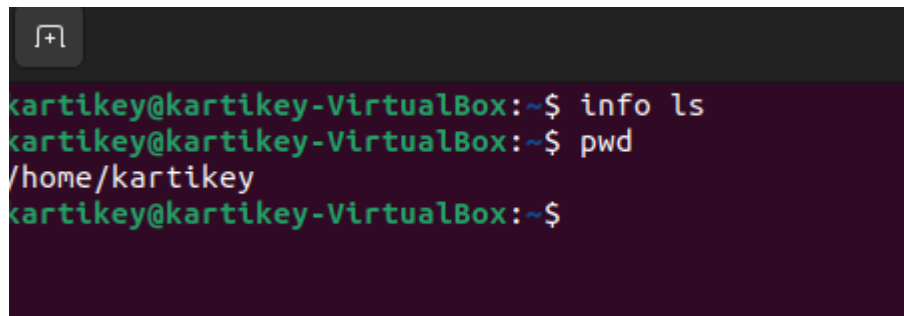
* Menu:

* Which files are listed::
* What information is listed::
* Sorting the output::
* General output formatting::

-----Info: (coreutils)ls invocation, 56 lines --Top-----
Welcome to Info version 6.8. Type H for help, h for tutorial.
```

v) **pwd** - It tells us about the present working directory.

**Syntax:** \$pwd

A terminal window with a dark background and green text. The prompt is 'kartikey@kartikey-VirtualBox:~\$'. The user enters 'info ls' and then 'pwd'. The output of 'pwd' is '/home/kartikey'.

```
kartikey@kartikey-VirtualBox:~$ info ls
kartikey@kartikey-VirtualBox:~$ pwd
/home/kartikey
kartikey@kartikey-VirtualBox:~$
```

## Program – 02

**Ques 02:** Demonstrate the use of whoami with various options wherever applicable.

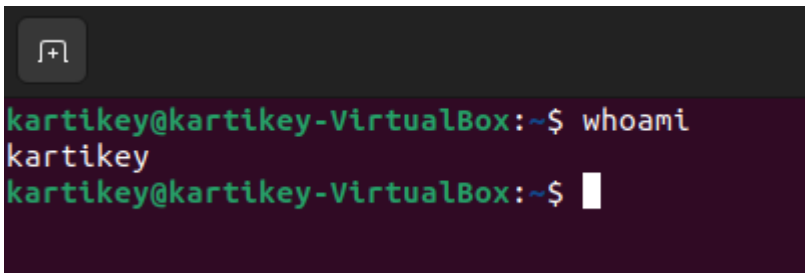
**whoami** - It is basically the concatenation of the strings “who”, “am”, ” i” as whoami. It displays the username of the current user when this command is invoked.

**Syntax:** whoami [option]

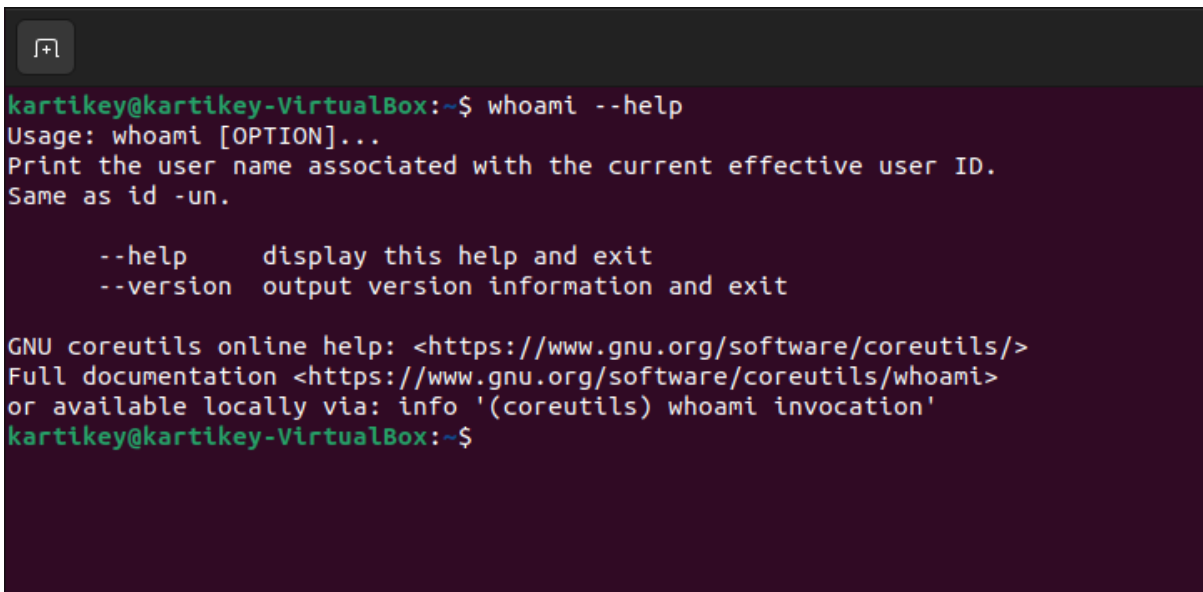
### **Various options:**

--help: shows a help message and exits.

--version: shows the version information and exits.

A terminal window with a dark background. The prompt is 'kartikey@kartikey-VirtualBox:~\$'. The command 'whoami' has been entered, and the output 'kartikey' is displayed on the next line. The prompt is now 'kartikey@kartikey-VirtualBox:~\$' with a cursor.

```
kartikey@kartikey-VirtualBox:~$ whoami
kartikey
kartikey@kartikey-VirtualBox:~$
```

A terminal window with a dark background. The prompt is 'kartikey@kartikey-VirtualBox:~\$'. The command 'whoami --help' has been entered, and the following help text is displayed: 'Usage: whoami [OPTION]...', 'Print the user name associated with the current effective user ID.', 'Same as id -un.', followed by two options: '--help display this help and exit' and '--version output version information and exit'. At the bottom, it provides links to GNU coreutils online help and full documentation, and mentions local availability via 'info'. The prompt is now 'kartikey@kartikey-VirtualBox:~\$'.

```
kartikey@kartikey-VirtualBox:~$ whoami --help
Usage: whoami [OPTION]...
Print the user name associated with the current effective user ID.
Same as id -un.

    --help      display this help and exit
    --version   output version information and exit

GNU coreutils online help: <https://www.gnu.org/software/coreutils/>
Full documentation <https://www.gnu.org/software/coreutils/whoami>
or available locally via: info '(coreutils) whoami invocation'
kartikey@kartikey-VirtualBox:~$
```

## Program – 03

**Ques 03:** Demonstrate the use of mkdir, rm, mv, cp cmd with various option.

i) **mkdir** – mkdir command in Linux allows the user to create directories (also referred to as folders in some operating systems). This command can create multiple directories at once as well as set the permissions for the directories.

**Syntax:** mkdir [options...] [directories...]

### **Various options:**

--help: It displays help-related information and exits.

--version: It displays the version number, some information regarding the license and exits.

-v: It displays a message for every directory created.

-p: A flag which enables the command to create parent directories as necessary.

```
kartikey@kartikey-VirtualBox:~$ mkdir file
kartikey@kartikey-VirtualBox:~$ ls
Desktop Documents Downloads file Music Pictures Public snap Templates Videos
kartikey@kartikey-VirtualBox:~$
```

```
kartikey@kartikey-VirtualBox:~$ mkdir --help
Usage: mkdir [OPTION]... DIRECTORY...
Create the DIRECTORY(ies), if they do not already exist.

Mandatory arguments to long options are mandatory for short options too.
  -m, --mode=MODE      set file mode (as in chmod), not a=rwx - umask
  -p, --parents         no error if existing, make parent directories as needed
  -v, --verbose         print a message for each created directory
  -Z                   set SELinux security context of each created directory
                        to the default type
  --context[=CTX]      like -Z, or if CTX is specified then set the SELinux
                        or SMACK security context to CTX
  --help               display this help and exit
  --version             output version information and exit

GNU coreutils online help: <https://www.gnu.org/software/coreutils/>
Full documentation <https://www.gnu.org/software/coreutils/mkdir>
or available locally via: info '(coreutils) mkdir invocation'
kartikey@kartikey-VirtualBox:~$
```

```
kartikey@kartikey-VirtualBox:~$ mkdir -v linux_enviroment
mkdir: created directory 'linux_enviroment'
kartikey@kartikey-VirtualBox:~$ ls
Desktop Documents Downloads file linux_enviroment Music Pictures Public snap Templates Videos
kartikey@kartikey-VirtualBox:~$
```

```
kartikey@kartikey-VirtualBox:~$ mkdir -p -v kartikey/linux/linux_enviroment
mkdir: created directory 'kartikey'
mkdir: created directory 'kartikey/linux'
mkdir: created directory 'kartikey/linux/linux_enviroment'
kartikey@kartikey-VirtualBox:~$
```

ii) **rm** - rm stands for remove here. rm command is used to remove objects such as files, directories, symbolic links and so on from the file system.

**Syntax:** rm [option...] file...

#### Various options:

-i (Interactive Deletion): Like in cp, the -i option makes the command ask the user for confirmation before removing each file, you have to press y for confirm deletion, any other key leaves the file undeleted.

-f (Force Deletion): rm prompts for confirmation removal if a file is write protected. The -f option overrides this minor protection and removes the file forcefully.

-r or -R (Recursive Deletion): rm command performs a tree-walk and will delete all the files and sub-directories recursively of the parent directory. At each stage it deletes everything it finds. Normally, rm wouldn't delete the directories but when used with this option, it will delete.

```
kartikey@kartikey-VirtualBox:~$ ls hello.txt
hello.txt
kartikey@kartikey-VirtualBox:~$ rm hello.txt
kartikey@kartikey-VirtualBox:~$ ls
1.txt  Documents  file  linux_enviroment  Pictures  snap  Videos
Desktop Downloads kartikey Music      Public  Templates
```

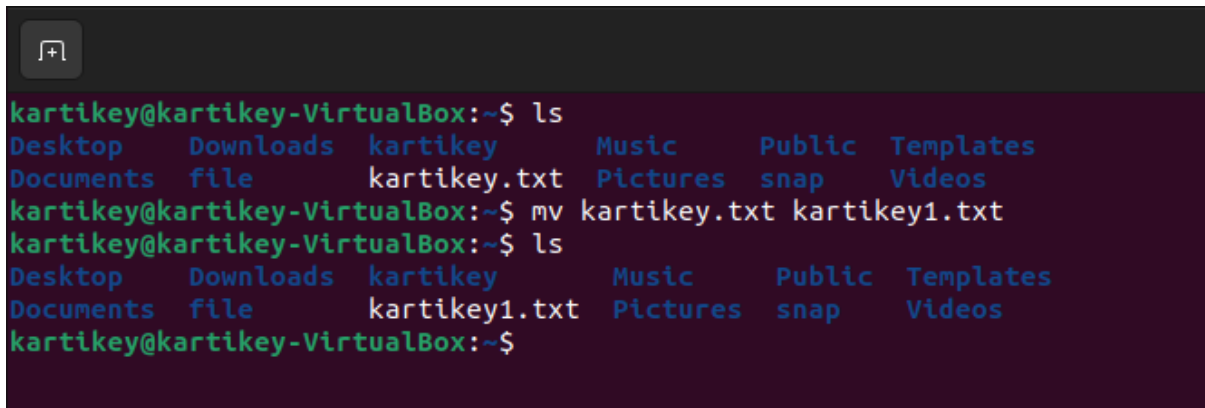
```
kartikey@kartikey-VirtualBox:~$ ls 1.txt
1.txt
kartikey@kartikey-VirtualBox:~$ rm -i 1.txt
rm: remove regular empty file '1.txt'? y
kartikey@kartikey-VirtualBox:~$ ls
Desktop Documents Downloads file kartikey linux_enviroment Music Pictures Public snap Templates Videos
kartikey@kartikey-VirtualBox:~$
```

```
kartikey@kartikey-VirtualBox:~$ ls new.txt
new.txt
kartikey@kartikey-VirtualBox:~$ rm -f new.txt
kartikey@kartikey-VirtualBox:~$ ls
Desktop Documents Downloads file kartikey linux_enviroment Music Pictures Public snap Templates Videos
kartikey@kartikey-VirtualBox:~$ rm -r linux_enviroment
kartikey@kartikey-VirtualBox:~$ ls
Desktop Documents Downloads file kartikey Music Pictures Public snap Templates Videos
kartikey@kartikey-VirtualBox:~$
```



iii) **mv** – mv stands for move. It has two distinct functions, i.e. renaming a file or directory and moving a file or directory from one location to another.

**Syntax:** mv [options(s)] [source\_file\_name(s)] [Destination\_file\_name]



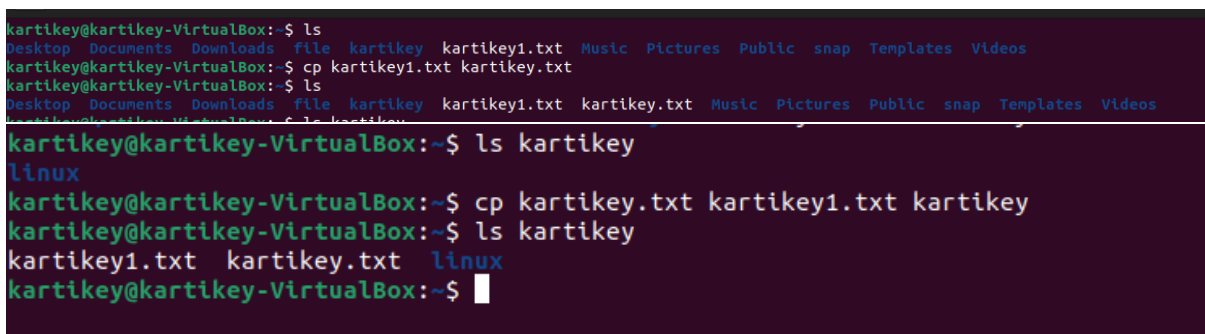
```
kartikey@kartikey-VirtualBox:~$ ls
Desktop  Downloads  kartikey  Music  Public  Templates
Documents file      kartikey.txt Pictures snap  Videos
kartikey@kartikey-VirtualBox:~$ mv kartikey.txt kartikey1.txt
kartikey@kartikey-VirtualBox:~$ ls
Desktop  Downloads  kartikey  Music  Public  Templates
Documents file      kartikey1.txt Pictures snap  Videos
kartikey@kartikey-VirtualBox:~$
```

iv) **cp** - cp stands for a copy. This command is used to copy files or groups of files or directories. It creates an exact image of a file on a disk with a different file name. cp command requires at least two filenames in its arguments.

**Syntax:** cp [option] Source Destination  
cp [option] Source Directory  
cp [option] Source-1 Source-2 Source-3 Source-n Directory

#### Various options:

- i (interactive): i stands for Interactive copying. With this option the system first warns the user before overwriting the destination file.
- b (backup): With this option cp command creates the backup of the destination file in the same folder with the different name and in different format.
- f (force): If the system is unable to open destination file for writing operation because the user doesn't have writing permission for this file then by using -f option with cp command, destination file is deleted first and then copying of content is done from source to destination file.
- r (recursive): Copying directory structure. With this option cp command shows its recursive behavior by copying the entire directory structure recursively.



```
kartikey@kartikey-VirtualBox:~$ ls
Desktop  Documents  Downloads  file  kartikey  kartikey1.txt  Music  Pictures  Public  snap  Templates  Videos
kartikey@kartikey-VirtualBox:~$ cp kartikey1.txt kartikey.txt
kartikey@kartikey-VirtualBox:~$ ls
Desktop  Documents  Downloads  file  kartikey  kartikey1.txt  kartikey.txt  Music  Pictures  Public  snap  Templates  Videos
kartikey@kartikey-VirtualBox:~$ ls kartikey
linux
kartikey@kartikey-VirtualBox:~$ cp kartikey.txt kartikey1.txt kartikey
kartikey@kartikey-VirtualBox:~$ ls kartikey
kartikey1.txt  kartikey.txt  linux
kartikey@kartikey-VirtualBox:~$
```

## Program – 04

**Ques 04:** Demonstrate the use of which, where is, locate cmd.

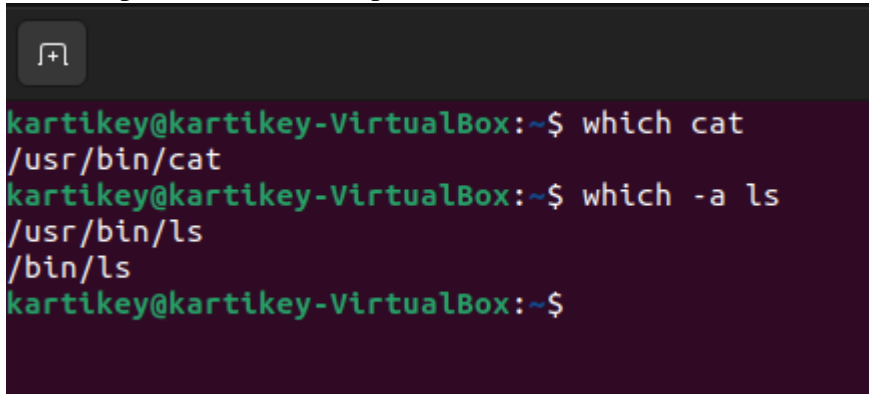
i) **which** - The which command allows users to search the list of paths in the \$PATH environment variable and outputs the full path of the command specified as an argument. The which command returns one of the following values that indicate its exit status:

- 0. All arguments were found and executable.
- 1. One or more arguments don't exist or aren't executable.
- 2. An invalid option has been specified.

**Syntax:** which [argument]

**Various options:**

-a: It is optional and used to print all the matches it finds.

A terminal window with a dark background and green text. The prompt is 'kartikey@kartikey-VirtualBox:~\$'. The first command is 'which cat', and the output is '/usr/bin/cat'. The second command is 'which -a ls', and the output is '/usr/bin/ls' followed by '/bin/ls' on a new line. The prompt returns to '~\$'.

```
kartikey@kartikey-VirtualBox:~$ which cat
/usr/bin/cat
kartikey@kartikey-VirtualBox:~$ which -a ls
/usr/bin/ls
/bin/ls
kartikey@kartikey-VirtualBox:~$
```

ii) **whereis** - whereis command is used to find the location of source/binary file of a command and manuals sections for a specified file in Linux system.

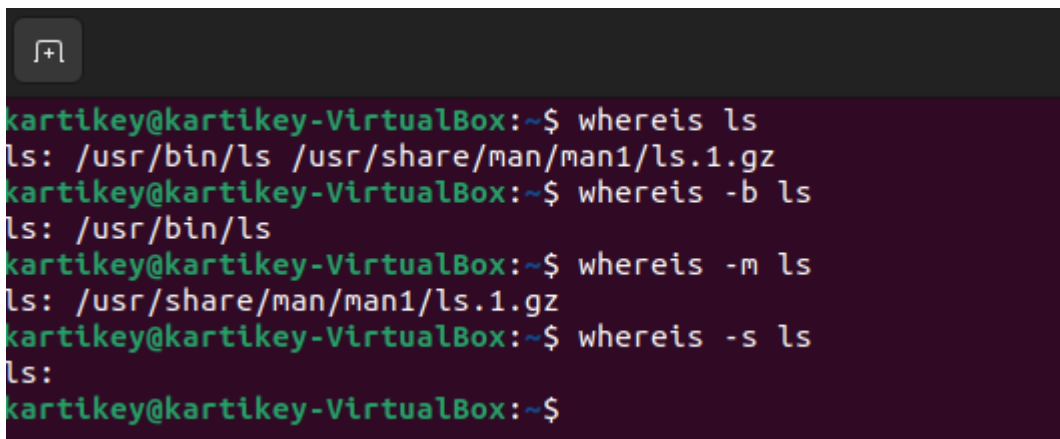
**Syntax:** whereis [options] filename...

**Various options:**

-b: This option is used when we only want to search for binaries.

-m: This option is used when we only want to search for manual sections.

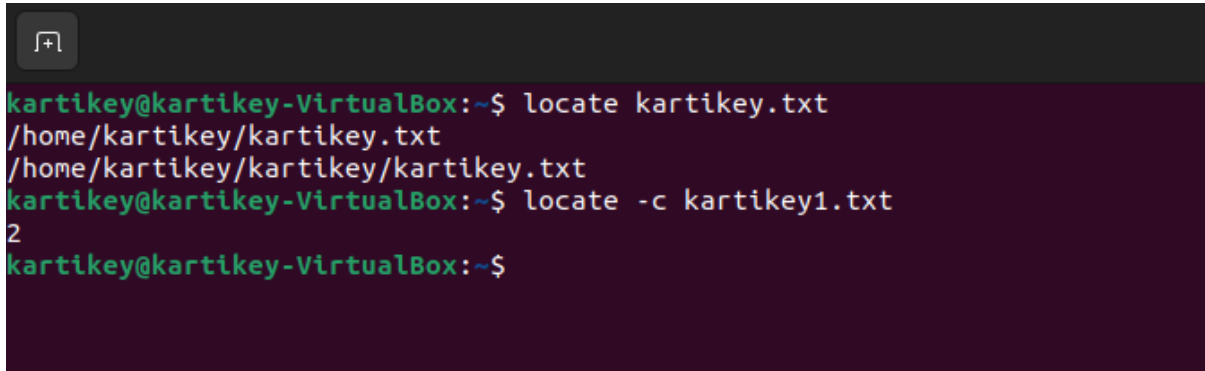
-s: This option is used when we only want to search for sources.

A terminal window with a dark background and green text. The prompt is 'kartikey@kartikey-VirtualBox:~\$'. The first command is 'whereis ls', and the output is 'ls: /usr/bin/ls /usr/share/man/man1/ls.1.gz'. The second command is 'whereis -b ls', and the output is 'ls: /usr/bin/ls'. The third command is 'whereis -m ls', and the output is 'ls: /usr/share/man/man1/ls.1.gz'. The fourth command is 'whereis -s ls', and the output is 'ls:'. The prompt returns to '~\$'.

```
kartikey@kartikey-VirtualBox:~$ whereis ls
ls: /usr/bin/ls /usr/share/man/man1/ls.1.gz
kartikey@kartikey-VirtualBox:~$ whereis -b ls
ls: /usr/bin/ls
kartikey@kartikey-VirtualBox:~$ whereis -m ls
ls: /usr/share/man/man1/ls.1.gz
kartikey@kartikey-VirtualBox:~$ whereis -s ls
ls:
kartikey@kartikey-VirtualBox:~$
```

iii) **locate** - locate command in Linux is used to find the files by name. This command will exit with status 0 if any specified match found. If no match founds or a fatal error encountered, then it will exit with status 1.

**Syntax:** locate [option]... pattern...

A terminal window with a dark background and light green text. The prompt is 'kartikey@kartikey-VirtualBox:~\$'. The first command is 'locate kartikey.txt', which returns two paths: '/home/kartikey/kartikey.txt' and '/home/kartikey/kartikey/kartikey.txt'. The second command is 'locate -c kartikey1.txt', which returns the number '2'. The prompt returns to '~\$' after the second command.

```
kartikey@kartikey-VirtualBox:~$ locate kartikey.txt
/home/kartikey/kartikey.txt
/home/kartikey/kartikey/kartikey.txt
kartikey@kartikey-VirtualBox:~$ locate -c kartikey1.txt
2
kartikey@kartikey-VirtualBox:~$
```

## Program – 05

**Ques 05:** Demonstrate the use of cat cmd in three ways - creating a file, display file content and appending a file (concatenation).

**cat** - Cat(concatenate) command is very frequently used in Linux. It reads data from the file and gives their content as output. It helps us to create, view, concatenate files.

### **Syntax:**

**To view a single file** – cat filename

**To create a file** – cat > filename (press enter to write content)

Ctrl+D to save and exit

**To append a file** – cat file1 >> file2 (will append the content of file1 into file2)

```
kartikey@kartikey-VirtualBox:~$ cat > program.txt
I am Kartikey Raghuvanshi and i am creating this file to start with linux
thank you for this.
█
```

*Creating a file*

```
kartikey@kartikey-VirtualBox:~$ cat program.txt
I am Kartikey Raghuvanshi and i am creating this file to start with linux
thank you for this.
kartikey@kartikey-VirtualBox:~$
```

*Displaying a file*

```
kartikey@kartikey-VirtualBox:~$ cat program.txt >> kartikey.txt
kartikey@kartikey-VirtualBox:~$ cat kartikey.txt
I am Kartikey Raghuvanshi and i am creating this file to start with linux
thank you for this.
kartikey@kartikey-VirtualBox:~$ █
```

*Appending a file*

## Program – 06

**Ques 06:** Demonstrate the use of wc, grep.

i) **wc** - wc stands for word count. As the name implies, it is mainly used for counting purpose. It is used to find out number of lines, word count, byte and characters count in the files specified in the file arguments. By default it displays four-columnar output. First column shows number of lines present in a file specified, second column shows number of words present in the file, third column shows number of characters present in file and fourth column itself is the file name which are given as argument.

**Syntax:** wc [option]... [file]...

### **Various options:**

- l: This option prints the number of lines present in a file.
- w: This option prints the number of words present in a file.
- c: This option displays count of bytes present in a file.

```
kartikey@kartikey-VirtualBox:~$ wc kartikey.txt
 2 18 94 kartikey.txt
kartikey@kartikey-VirtualBox:~$ wc -l kartikey.txt
2 kartikey.txt
kartikey@kartikey-VirtualBox:~$ wc -w kartikey.txt
18 kartikey.txt
kartikey@kartikey-VirtualBox:~$ wc -c kartikey.txt
94 kartikey.txt
kartikey@kartikey-VirtualBox:~$
```

ii) **grep** - The grep filter searches a file for a particular pattern of characters, and displays all lines that contain that pattern. The pattern that is searched in the file is referred to as the regular expression (grep stands for global search for regular expression and print out).

**Syntax:** grep [options] pattern [files]

### **Various options:**

- c: This prints only a count of the lines that match a pattern
- i: Ignores, case for matching
- l: Displays list of a filenames only.
- n: Display the matched lines and their line numbers.
- v: This prints out all the lines that do not matches the pattern.

```
kartikey@kartikey-VirtualBox:~$ cat program3.txt
 6      7      33
kartikey@kartikey-VirtualBox:~$ cat program1.txt
unix is a great os. unix was developed in Bell labs.
unix is use or ethical hacking.
kartikey@kartikey-VirtualBox:~$ grep -i "unix" program1.txt
unix is a great os. unix was developed in Bell labs.
unix is use or ethical hacking.
unix is base of linux and also unix is base for ios system.
kartikey@kartikey-VirtualBox:~$
```

```
kartikey@kartikey-VirtualBox:~$ grep -c "unix" program1.txt
3
```

```
kartikey@kartikey-VirtualBox:~$ grep -l "unix" *
grep: Desktop: Is a directory
grep: Documents: Is a directory
grep: Downloads: Is a directory
grep: file: Is a directory
grep: kartikey: Is a directory
grep: Music: Is a directory
grep: Pictures: Is a directory
program1.txt
grep: Public: Is a directory
grep: snap: Is a directory
grep: Templates: Is a directory
grep: Videos: Is a directory
```

```
kartikey@kartikey-VirtualBox:~$ grep -n "unix" program1.txt
1:unix is a great os. unix was developed in Bell labs.
2:unix is use or ethical hacking.
3:unix is base of linux and also unix is base for ios system.
```

```
kartikey@kartikey-VirtualBox:~$ grep -v "unix" program1.txt
kartikey@kartikey-VirtualBox:~$
```

## Program – 07

**Ques 07:** Demonstrate the use of pipes in at least 5 commands.

**Pipes** - A pipe is a form of redirection (transfer of standard output to some other destination) that is used in Linux and other Unix-like operating systems to send the output of one command/program/process to another command/program/process for further processing. The Unix/Linux systems allow the stdout of a command to be connected to the stdin of another command. You can make it do so by using the pipe character '|'.  
Example: `ls | grep program1.txt`

**Syntax:** `command_1 | command_2 | command_3 | .... | command_N`

```
kartikey@kartikey-VirtualBox:~$ ls
Desktop  Documents  Downloads  file  fresh.txt  kartikey  kartikey1.txt  kartikey.txt  Music  Pictures  program1.txt  program2.txt  program3.txt  program.txt  Public  snap  Templates  Videos
```

```
kartikey@kartikey-VirtualBox:~$ ls | grep program1.txt
program1.txt
```

```
kartikey@kartikey-VirtualBox:~$ cat program2.txt
Hello
bye
Hiii
Noo
Yes
Take care
kartikey@kartikey-VirtualBox:~$ cat program2.txt | sort
bye
Hello
Hiii
Noo
Take care
Yes
```

```
kartikey@kartikey-VirtualBox:~$ cat program2.txt | grep Hiii
Hiii
```

```
kartikey@kartikey-VirtualBox:~$ cat program2.txt | head -5
Hello
bye
Hiii
Noo
Yes
```

```
kartikey@kartikey-VirtualBox:~$ cat program2.txt | head -5 | tail -3
Hiii
Noo
Yes
```

```
kartikey@kartikey-VirtualBox:~$ cat program2.txt | wc
 6    7   33
```

```
kartikey@kartikey-VirtualBox:~$ cat program2.txt | wc > program4.txt
kartikey@kartikey-VirtualBox:~$ cat program4.txt
 6    7   33
```