Delhi Technical Campus, Greater Noida

(Affiliated to GGSIPU, New Delhi)

Lab Manual

Linux Environment

(Bachelors of Computer Application)

Department of Computer Science and Engineering

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

Submitted by: Kartikey Raghuvanshi
00718002020
BCA 3rd Year

INDEX

Teacher's Signature:

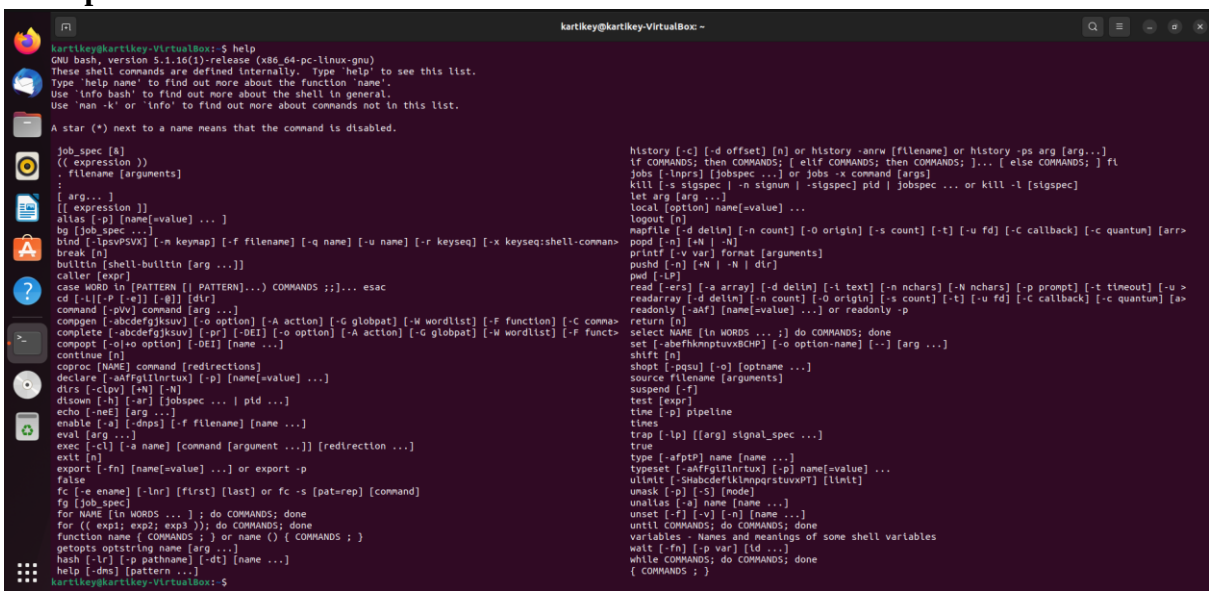
Program – 01

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]…

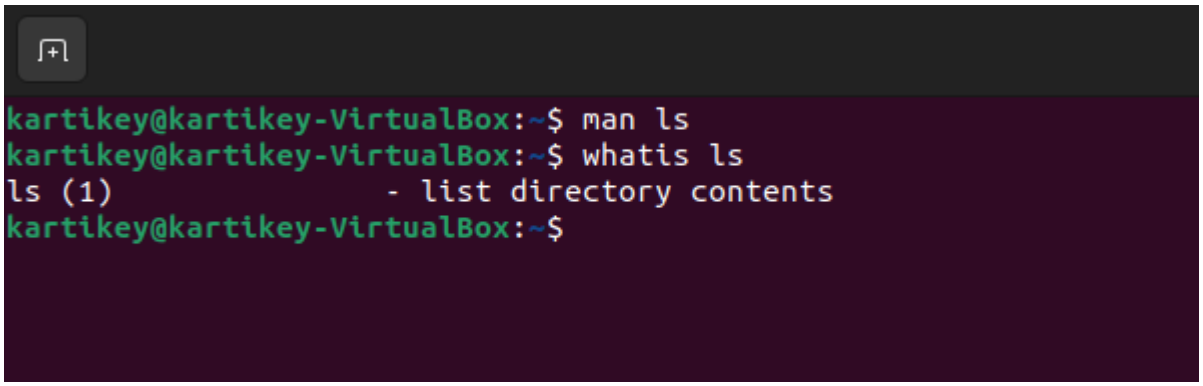
Various options:
help-d: It displays the short description for each topic.
help-m: It displays usage in pseudo manpage format.
help-s: It shows output only a short synopsis.

Example:



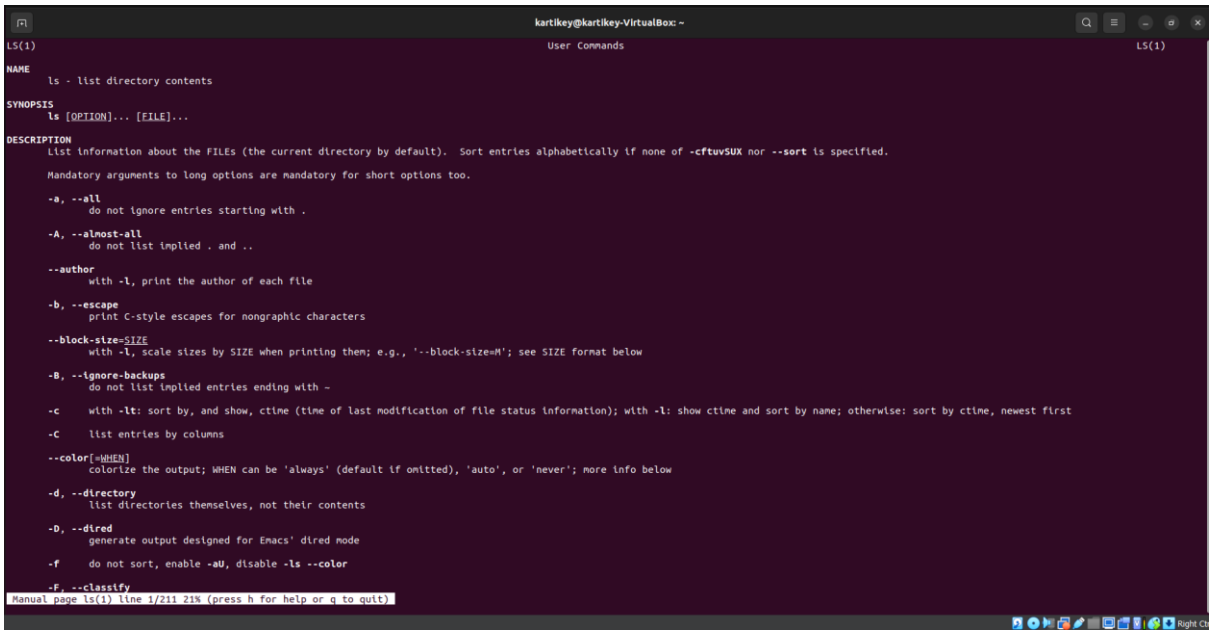
ii) **whatis** - whatis command in Linux is used to get a one-line manual page description.

Syntax: whatis [option]… [command_name]…



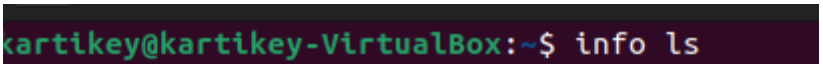
iii) **man** - man command in Linux is used to display the user manual of any command that we can run on the terminal. It provides a detailed view of the command.

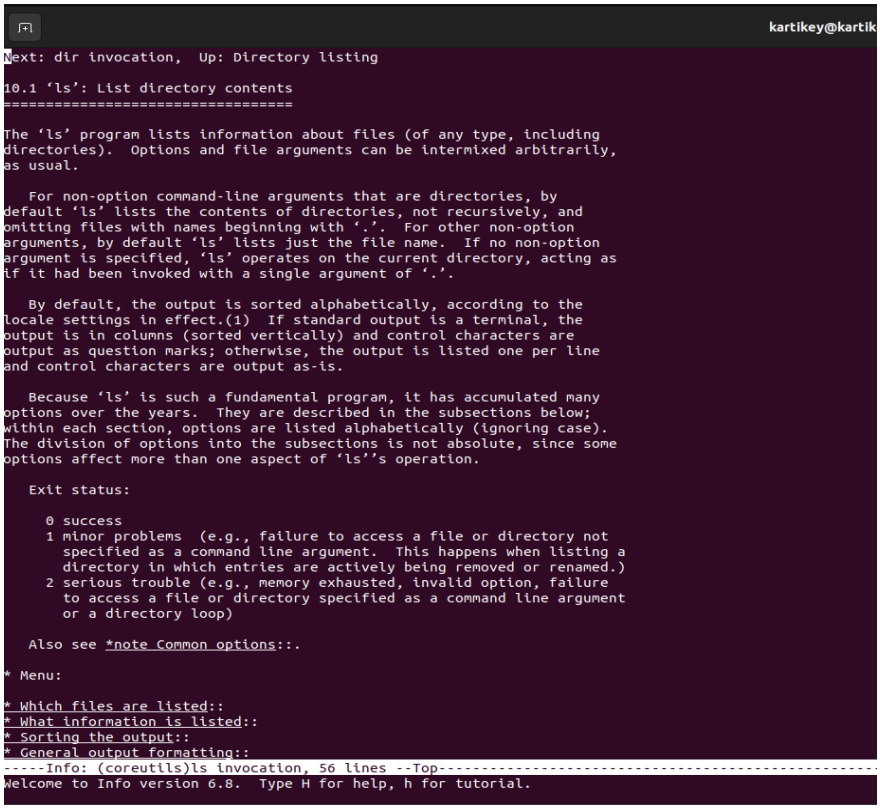
Syntax: $man [option]... [command_name]...



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

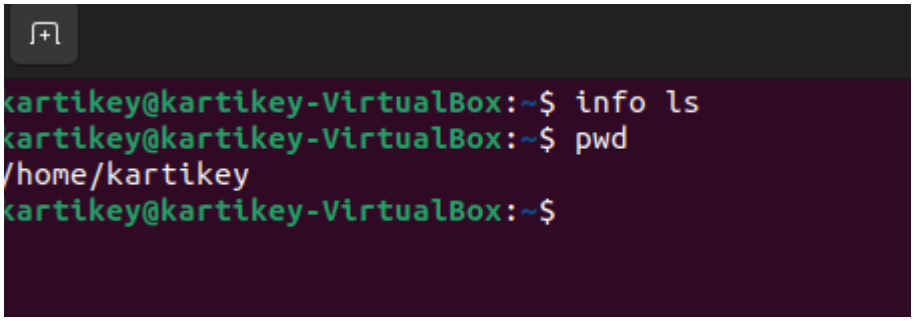
Syntax: info [option]... [menu-item]…





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

Syntax: $pwd



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.

```
kartikey@kartikey-VirtualBox:~$ whoami
kartikey
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: ~
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 un-deleted.
-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:~$
```

```
kartikey@kartikey-VirtualBox: ~
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: ~
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.

```
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.

```
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…

```
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 '|'.

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
```