Prog 1: To connect to linux server and understand basic directory structure of linux.

Connect to Linux Server using SSH:

1.Open a Terminal on your local machine:

On Linux or macOS, you can use the built-in terminal.

On Windows, you can use a terminal emulator like PuTTY or the Windows Subsystem for Linux.

2.Use the ssh command to connect:

ssh username@server_ip

example:-

ssh john@example.com

Basic Linux Directory Structure:

Linux follows a hierarchical directory structure. Here are some key directories:

1./ (Root Directory): The top-level directory. Everything on the system is under this directory.

2./home: Home directories for regular users. Each user has a subdirectory here with their name.

3./etc: Configuration files for the system and various applications.

4./bin and /usr/bin: Essential system binaries (commands) are stored here.

5./sbin and /usr/sbin: System binaries that are generally reserved for superuser (root) are stored here.

6./var: Variable files like logs, databases, etc.

7./tmp: Temporary files.

8./dev: Device files, representing hardware devices.

9./proc: A virtual filesystem containing information about processes and kernel parameters.

10./lib and /usr/lib: Essential system libraries.

11./boot: Contains files needed to boot the system, including the kernel.

12./mnt and /media: Directories for temporarily mounting filesystems.

13./opt: Typically used for installing third-party software.

14./srv: Data for services provided by the system.

Prog 2: To understand help commands like man, info ,help ,whatis ,apropos :

1.man - Manual Pages:

man stands for manual, and it's used to display the manual pages for various commands and utilities.

Example: man ls will display the manual page for the ls command, providing information on its usage, options, and more.

Navigation in man pages: Use arrow keys or the keyboard shortcuts described at the bottom (e.g., q to quit, / to search).

2.info - GNU Info System:

info is another system for providing documentation. It often contains more detailed and structured information than man pages.

Example: info ls will provide information about the ls command in an info format.

Navigation in info: Similar to man, but with some additional commands. Use the spacebar to move forward, b to move backward, and q to quit.

3.help - Shell Built-in Help:

The help command is a shell built-in command that provides information about built-in shell commands.

Example: help cd will provide information about the cd command, which is a built-in shell command for changing directories.

4.whatis - Display One-line Manual Page Descriptions:

whatis is used to display a one-line description of a command.

Example: whatis ls will display a short description of the ls command.

5.apropos - Search the Manual Page Names and Descriptions:

apropos is used for searching the manual page names and descriptions for a keyword.

Example: apropos text editor will display a list of commands related to text editors.

```
NAME
       man - an interface to the system reference manuals

SYNOPSIS
       man [man options] [[section] page ...] ...
       man -k [apropos options] regexp ...
       man -K [man options] [section] term ...
       man -f [whatis options] page ...
       man -l [man options] file ...
       man -w|-W [man options] page ...

DESCRIPTION
       man  is the system's manual pager.  Each page argument given to man is normally the name of
       a program, utility or function.  The manual page associated with each of these arguments is
       then  found  and  displayed.   A section, if provided, will direct man to look only in that
       section of the manual.  The default action is to search in all of  the  available  sections
       following  a  pre-defined order (see DEFAULTS), and to show only the first page found, even
       if page exists in several sections.

       The table below shows the section numbers of the manual followed by the types of pages they
       contain.

       1   Executable programs or shell commands
       2   System calls (functions provided by the kernel)
       3   Library calls (functions within program libraries)
       4   Special files (usually found in /dev)
       5   File formats and conventions, e.g. /etc/passwd
       6   Games
       7   Miscellaneous  (including  macro  packages  and  conventions),  e.g.  man(7), groff(7),
           man-pages(7)
       8   System administration commands (usually only for root)
       9   Kernel routines [Non standard]

       A manual page consists of several sections.

       Conventional section names include NAME,  SYNOPSIS,  CONFIGURATION,  DESCRIPTION,  OPTIONS,
       EXIT STATUS,  RETURN VALUE,  ERRORS,  ENVIRONMENT,  FILES,  VERSIONS, CONFORMING TO, NOTES,
       BUGS, EXAMPLE, AUTHORS, and SEE ALSO.

       The following conventions apply to the SYNOPSIS section and can be used as a guide in other
       sections.

       bold text          type exactly as shown.
       italic text        replace with appropriate argument.
       [-abc]             any or all arguments within [ ] are optional.
       -a|-b              options delimited by | cannot be used together.
       argument ...       argument is repeatable.
       [expression] ...   entire expression within [ ] is repeatable.

       Exact  rendering  may  vary depending on the output device.  For instance, man will usually
       not be able to render italics when running in a terminal, and will typically use underlined
       or coloured text instead.

 Manual page man(1) line 2 (press h for help or q to quit)
```

```
              poor alignment, which are unsightly and generally  confusing  when  displayed  along
              with  the  manual  page.   However,  some  users  want  to  see  them anyway, so, if
              $MAN_KEEP_STDERR is set to any non-empty value, error output will  be  displayed  as
              usual.

       MAN_DISABLE_SECCOMP
              On Linux, man normally confines subprocesses that handle untrusted data using a sec-
              comp(2) sandbox.  This makes it safer to run complex  parsing  code  over  arbitrary
              manual  pages.   If  this goes wrong for some reason unrelated to the content of the
              page being displayed, you can set $MAN_DISABLE_SECCOMP to  any  non-empty  value  to
              disable the sandbox.

       PIPELINE_DEBUG
              If  the  $PIPELINE_DEBUG environment variable is set to "1", then man will print de-
              bugging messages to standard error describing each subprocess it runs.

       LANG, LC_MESSAGES
              Depending on system and implementation, either or both  of  $LANG  and  $LC_MESSAGES
              will  be interrogated for the current message locale.  man will display its messages
              in that locale (if available).  See setlocale(3) for precise details.

FILES
       /etc/manpath.config
              man-db configuration file.

       /usr/share/man
              A global manual page hierarchy.

SEE ALSO
       apropos(1), groff(1), less(1), manpath(1), nroff(1), troff(1), whatis(1), zsoelim(1),  man-
       path(5), man(7), catman(8), mandb(8)

       Documentation for some packages may be available in other formats, such as info(1) or HTML.

HISTORY
       1990, 1991 - Originally written by John W. Eaton (jwe@che.utexas.edu).

       Dec  23  1992:  Rik  Faith  (faith@cs.unc.edu) applied bug fixes supplied by Willem Kasdorp
       (wkasdo@nikhefk.nikef.nl).

       30th April 1994 - 23rd February 2000: Wilf. (G.Wilford@ee.surrey.ac.uk) has been developing
       and maintaining this package with the help of a few dedicated people.

       30th  October 1996 - 30th March 2001: Fabrizio Polacco <fpolacco@debian.org> maintained and
       enhanced this package for the Debian project, with the help of all the community.

       31st March 2001 - present day: Colin Watson <cjwatson@debian.org>  is  now  developing  and
       maintaining man-db.

BUGS
       https://gitlab.com/cjwatson/man-db/-/issues
       https://savannah.nongnu.org/bugs/?group=man-db

2.10.2                                    2022-03-17                                        MAN(1)
 Manual page man(1) line 557/610 (END) (press h for help or q to quit)
```

```
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::
* Formatting file timestamps::
* Formatting the file names::

   ---------- Footnotes ----------

   (1) If you use a non-POSIX locale (e.g., by setting 'LC_ALL' to
'en_US'), then 'ls' may produce output that is sorted differently than
-----Info: (coreutils)ls invocation, 56 lines --Top-----------------------------------------------------------------------------------
Welcome to Info version 6.8.  Type H for help, h for tutorial.
```

```
anurag@anurag-Nitro-AN515-57:~$ help cd
cd: cd [-L|[-P [-e]] [-@]] [dir]
    Change the shell working directory.

    Change the current directory to DIR.  The default DIR is the value of the
    HOME shell variable.

    The variable CDPATH defines the search path for the directory containing
    DIR.  Alternative directory names in CDPATH are separated by a colon (:).
    A null directory name is the same as the current directory.  If DIR begins
    with a slash (/), then CDPATH is not used.

    If the directory is not found, and the shell option `cdable_vars' is set,
    the word is assumed to be  a variable name.  If that variable has a value,
    its value is used for DIR.

    Options:
      -L        force symbolic links to be followed: resolve symbolic
                links in DIR after processing instances of `..'
      -P        use the physical directory structure without following
                symbolic links: resolve symbolic links in DIR before
                processing instances of `..'
      -e        if the -P option is supplied, and the current working
                directory cannot be determined successfully, exit with
                a non-zero status
      -@        on systems that support it, present a file with extended
                attributes as a directory containing the file attributes

    The default is to follow symbolic links, as if `-L' were specified.
    `..' is processed by removing the immediately previous pathname component
    back to a slash or the beginning of DIR.

    Exit Status:
    Returns 0 if the directory is changed, and if $PWD is set successfully when
    -P is used; non-zero otherwise.
```

```
anurag@anurag-Nitro-AN515-57:~$ whatis ls
ls (1)               - list directory contents
anurag@anurag-Nitro-AN515-57:~$
```

```
gslp (1)             - Format and print text using ghostscript
HTML::Filter (3pm)   - Filter HTML text through the parser
HTML::FormatText (3pm) - Format HTML as plaintext
iconv (1)            - convert text from one character encoding to another
jfs_debugfs (8)      - shell-type JFS file system editor
Locale::gettext (3pm) - message handling functions
lzless (1)           - view xz or lzma compressed (text) files
lzmore (1)           - view xz or lzma compressed (text) files
mawk (1)             - pattern scanning and text processing language
md5sum.textutils (1) - compute and check MD5 message digest
nano (1)             - Nano's ANOther editor, inspired by Pico
nawk (1)             - pattern scanning and text processing language
ngettext (1)         - translate message and choose plural form
ngettext (3)         - translate message and choose plural form
nm-connection-editor (1) - network connection editor for NetworkManager
nmtui (1)            - Text User Interface for controlling NetworkManager
nmtui-connect (1)    - Text User Interface for controlling NetworkManager
nmtui-edit (1)       - Text User Interface for controlling NetworkManager
nmtui-hostname (1)   - Text User Interface for controlling NetworkManager
pam_echo (8)         - PAM module for printing text messages
pam_selinux (7)      - PAM module to set the default security context
pdftotext (1)        - Portable Document Format (PDF) to text converter (version 3.03)
pico (1)             - Nano's ANOther editor, inspired by Pico
pkttyagent (1)       - Textual authentication helper
pod2text (1)         - Convert POD data to formatted ASCII text
pr (1)               - convert text files for printing
pygettext3 (1)       - Python equivalent of xgettext(1)
pygettext3.10 (1)    - Python equivalent of xgettext(1)
red (1)              - line-oriented text editor
runcon (1)           - run command with specified security context
rview (1)            - Vi IMproved, a programmer's text editor
rvim (1)             - Vi IMproved, a programmer's text editor
sed (1)              - stream editor for filtering and transforming text
select-editor (1)    - select your default sensible-editor from all installed editors
sensible-editor (1)  - sensible editing
software-properties-gtk (1) - Software Sources List editor
sort (1)             - sort lines of text files
spd-say (1)          - send text-to-speech output request to speech-dispatcher
tc-pedit (8)         - generic packet editor action
tc-skbmod (8)        - user-friendly packet editor action
Text::CharWidth (3pm) - Get number of occupied columns of a string on terminal
Text::Iconv (3pm)    - Perl interface to iconv() codeset conversion function
Text::WrapI18N (3pm) - Line wrapping module with support for multibyte, fullwidth, and combining characters and languages without whitespaces between words
textdomain (3)       - set domain for future gettext() calls
troff (1)            - the troff processor of the groff text formatting system
vi (1)               - Vi IMproved, a programmer's text editor
view (1)             - Vi IMproved, a programmer's text editor
vim (1)              - Vi IMproved, a programmer's text editor
xedit (1)            - simple text editor for X
xmore (1)            - plain text display program for the X Window System
xzless (1)           - view xz or lzma compressed (text) files
xzmore (1)           - view xz or lzma compressed (text) files
zless (1)            - file perusal filter for crt viewing of compressed text
zmore (1)            - file perusal filter for crt viewing of compressed text
anurag@anurag-Nitro-AN515-57:~$
```

```
ls (1)               - list directory contents
anurag@anurag-Nitro-AN515-57: $ apropos text editor
apt-transport-http (1) - APT transport for downloading via the Hypertext Transfer Protocol (HTTP)
atobm (1)            - bitmap editor and converter utilities for the X Window System
awk (1)              - pattern scanning and text processing language
bind_textdomain_codeset (3) - set encoding of message translations
bindtextdomain (3)   - set directory containing message catalogs
bitmap (1)           - bitmap editor and converter utilities for the X Window System
bmtoa (1)            - bitmap editor and converter utilities for the X Window System
bzless (1)           - file perusal filter for crt viewing of bzip2 compressed text
bzmore (1)           - file perusal filter for crt viewing of bzip2 compressed text
chcon (1)            - change file security context
csplit (1)           - split a file into sections determined by context lines
dcgettext (3)        - translate message
dcngettext (3)       - translate message and choose plural form
dgettext (3)         - translate message
dngettext (3)        - translate message and choose plural form
Dpkg::Gettext (3perl) - convenience wrapper around Locale::gettext
echo (1)             - display a line of text
ed (1)               - line-oriented text editor
editor (1)           - Nano's ANOther editor, inspired by Pico
editres (1)          - a dynamic resource editor for X Toolkit applications
ex (1)               - Vi IMproved, a programmer's text editor
file2brl (1)         - Translate an xml or a text file into an embosser-ready braille file
fmt (1)              - simple optimal text formatter
gedit (1)            - text editor for the GNOME Desktop
gettext (1)          - translate message
gettext (3)          - translate message
Glib::OptionContext (3pm) - defines options accepted by the commandline option parser
gnome-text-editor (1) - text editor for the GNOME Desktop
gparted (8)          - GNOME Partition Editor for manipulating disk partitions.
gsbj (1)             - Format and print text for BubbleJet printer using ghostscript
gsdj (1)             - Format and print text for DeskJet printer using ghostscript
gsdj500 (1)          - Format and print text for DeskJet 500 BubbleJet using ghostscript
gslj (1)             - Format and print text for LaserJet printer using ghostscript
gslp (1)             - Format and print text using ghostscript
HTML::Filter (3pm)   - Filter HTML text through the parser
HTML::FormatText (3pm) - Format HTML as plaintext
iconv (1)            - convert text from one character encoding to another
jfs_debugfs (8)      - shell-type JFS file system editor
Locale::gettext (3pm) - message handling functions
lzless (1)           - view xz or lzma compressed (text) files
lzmore (1)           - view xz or lzma compressed (text) files
mawk (1)             - pattern scanning and text processing language
md5sum.textutils (1) - compute and check MD5 message digest
nano (1)             - Nano's ANOther editor, inspired by Pico
nawk (1)             - pattern scanning and text processing language
ngettext (1)         - translate message and choose plural form
ngettext (3)         - translate message and choose plural form
nm-connection-editor (1) - network connection editor for NetworkManager
nmtui (1)            - Text User Interface for controlling NetworkManager
nmtui-connect (1)    - Text User Interface for controlling NetworkManager
nmtui-edit (1)       - Text User Interface for controlling NetworkManager
nmtui-hostname (1)   - Text User Interface for controlling NetworkManager
pam_echo (8)         - PAM module for printing text messages
pam_selinux (7)      - PAM module to set the default security context
```

Prog 3: To understand basic directory navigation commands like cat ,cd ,mv ,cp ,rm ,mkdir ,rmdir ,file ,pwd command

1.cd - Change Directory:

Use cd to change your current working directory.

Example: cd /path/to/directory or simply cd directory if it's in the current directory.

2.ls - List Directory Contents:

Use ls to list the contents of a directory.

Example: ls, ls /path/to/directory.

3.pwd - Print Working Directory:

Use pwd to print the current working directory.

Example: pwd.

4.mkdir - Make Directory:

Use mkdir to create a new directory.

Example: mkdir new_directory.

5.rmdir - Remove Directory:

Use rmdir to remove an empty directory.

Example: rmdir empty_directory.

6.cp - Copy:

Use cp to copy files or directories.

Example: cp file.txt /path/to/destination, cp -r directory /path/to/destination (for directories).

7.mv - Move/Rename:

Use mv to move files or directories, or to rename them.

Example: mv file.txt /path/to/destination, mv old_name.txt new_name.txt, mv directory /path/to/destination (for directories).

8.rm - Remove/Delete:

Use rm to remove files or directories.

Example: rm file.txt, rm -r directory (for directories, be cautious with the -r option).

9.cat - Concatenate and Display:

Use cat to display the contents of a file or concatenate files.

Example: cat file.txt.

10.file - Determine File Type:

Use file to determine the type of a file.

Example: file document.pdf.

```
anurag@anurag-Nitro-AN515-57:~$ cd Downloads/
anurag@anurag-Nitro-AN515-57:~/Downloads$
```

```
anurag@anurag-Nitro-AN515-57:~$ cd Downloads/
anurag@anurag-Nitro-AN515-57:~/Downloads$ ls
discord-0.0.36.deb  testcli
anurag@anurag-Nitro-AN515-57:~/Downloads$ pwd
/home/anurag/Downloads
anurag@anurag-Nitro-AN515-57:~/Downloads$ mkdir new_folder
anurag@anurag-Nitro-AN515-57:~/Downloads$ ls
discord-0.0.36.deb  new_folder  testcli
anurag@anurag-Nitro-AN515-57:~/Downloads$ rmdir new_folder
anurag@anurag-Nitro-AN515-57:~/Downloads$ ls
discord-0.0.36.deb  testcli
anurag@anurag-Nitro-AN515-57:~/Downloads$ touch test.txt
anurag@anurag-Nitro-AN515-57:~/Downloads$ cp test.txt
cp: missing destination file operand after 'test.txt'
Try 'cp --help' for more information.
anurag@anurag-Nitro-AN515-57:~/Downloads$ pwd
/home/anurag/Downloads
anurag@anurag-Nitro-AN515-57:~/Downloads$ cp /home/anurag/Downloads/test.txt
cp: missing destination file operand after '/home/anurag/Downloads/test.txt'
Try 'cp --help' for more information.
anurag@anurag-Nitro-AN515-57:~/Downloads$ cp test.txt /home/anurag/Downloads
cp: 'test.txt' and '/home/anurag/Downloads/test.txt' are the same file
anurag@anurag-Nitro-AN515-57:~/Downloads$
```

```
anurag@anurag-Nitro-AN515-57:~$ cd Downloads/
anurag@anurag-Nitro-AN515-57:~/Downloads$ mkdir cpcomm
anurag@anurag-Nitro-AN515-57:~/Downloads$ cp file.txt cpcomm
anurag@anurag-Nitro-AN515-57:~/Downloads$ cd cpcomm
anurag@anurag-Nitro-AN515-57:~/Downloads/cpcomm$ pwd
/home/anurag/Downloads/cpcomm
anurag@anurag-Nitro-AN515-57:~/Downloads/cpcomm$ bash file.txt
file.txt: line 1: this: command not found
file.txt: line 2: ignore: command not found
anurag@anurag-Nitro-AN515-57:~/Downloads/cpcomm$
```

```
anurag@anurag-Nitro-AN515-57:~$ cd Downloads/
anurag@anurag-Nitro-AN515-57:~/Downloads$ cat file.txt
this is for demo
ignore this.


anurag@anurag-Nitro-AN515-57:~/Downloads$ file file.txt
file.txt: ASCII text
anurag@anurag-Nitro-AN515-57:~/Downloads$
```

Prog 4: To understand basic commands like:-

date,cal,echo,bc,ls,who,whoami,hostname,uname,tty,aliase

- date:
  - Displays the current date and time.
- cal:
  - Displays a calendar for the current month or a specified month/year.
- echo:
  - Prints text or variables to the terminal.
- bc (Basic Calculator):
  - Provides a command-line calculator.
- ls (List Files and Directories):
  - Lists the files and directories in the current directory.
- who:

    - Displays information about users currently logged in.
- whoami:
    - Prints the username of the current user.
- hostname:
    - Displays the name of the current host (computer).
- uname:
    - Prints system information.
- tty:
    - Prints the file name of the terminal connected to the standard input.

- alias:
    - Creates a shortcut or alias for a command.

```
vboxuser@ubuntu:~/Desktop/jarvis/New$ date
Saturday 02 December 2023 04:35:04 PM IST
vboxuser@ubuntu:~/Desktop/jarvis/New$ echo "Hello"
Hello
vboxuser@ubuntu:~/Desktop/jarvis/New$ bc
bc 1.07.1
Copyright 1991-1994, 1997, 1998, 2000, 2004, 2006, 2008, 2012-2017 Free Software Foundation, Inc.
This is free software with ABSOLUTELY NO WARRANTY.
For details type `warranty'.
10+2
12
^C
(interrupt) use quit to exit.
quit
vboxuser@ubuntu:~/Desktop/jarvis/New$ ls
fileone.txt
vboxuser@ubuntu:~/Desktop/jarvis/New$ who
vboxuser tty2         2023-12-02 16:00 (tty2)
vboxuser@ubuntu:~/Desktop/jarvis/New$ whoami
vboxuser
vboxuser@ubuntu:~/Desktop/jarvis/New$ hostname
ubuntu
vboxuser@ubuntu:~/Desktop/jarvis/New$ uname -a
Linux ubuntu 6.2.0-37-generic #38~22.04.1-Ubuntu SMP PREEMPT_DYNAMIC Thu Nov  2 18:01:13 UTC 2 x86_64 x86_64 x86_64 GNU/Linux
vboxuser@ubuntu:~/Desktop/jarvis/New$ tty
/dev/pts/0
vboxuser@ubuntu:~/Desktop/jarvis/New$ alias ll='ls -l'
```

Prog 5: To understand vi basics, Three modes of vi Editor, how to write, save, execute a shell script in vi editor.

Vi is a text editor that is commonly used in Unix and Linux operating systems. It is a versatile and powerful tool for creating, editing, and managing text files. Vi operates in different modes, allowing users to perform various tasks efficiently.

Vi Modes:

Normal Mode:

Use arrow keys to

navigate. Press i

to enter Insert

mode.

Press : to enter Command-line mode.

Insert Mode:

Press Esc to return to Normal mode.

In Insert mode, you can type and edit text.

Command-line Mode:

Press : in Normal mode to enter

Command-line mode. Execute

commands like saving, quitting, etc.

Example: :w to save, :q to quit, :wq to save and quit.

Writing and Saving a Shell Script in Vi:

Open Vi:

"vi script.sh" type this command

Switch to Insert Mode:

Press i in Normal mode.

Write Your Script:

Type or paste your shell script.

Save and Exit:

Press Esc to enter Normal mode.

Type :wq to save and exit, then press Enter.

Example Vi Commands:

- **Navigation in Normal Mode:**
  - h: Move left
  - j: Move down
  - k: Move up
  - l: Move right

Editing in Insert Mode:

- Type or edit text as needed.

Saving and Quitting in Command-line Mode:

- :w: Save (write)
- :q: Quit (exit)
- :wq: Save and quit

Prog 6: To understand process related commands like: - ps, top, pstree, nice, renice in Linux.

- ps (Process Status):
  - Displays information about currently running processes.
- top:
  - Provides a dynamic real-time view of the running system.
- pstree:
  - Displays processes in a tree structure, showing their parent-child relationships.
- nice:
  - Adjusts the priority of a process, making it more or less favorable to the CPU scheduler.
- renice:
  - Changes the priority of a running process.

```
vboxuser@ubuntu:~/Desktop/jarvis/New$ ps aux
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root           1  0.0  0.4 167796 12852 ?        Ss   15:59   0:01 /sbin/init splash
root           2  0.0  0.0      0     0 ?        S    15:59   0:00 [kthreadd]
root           3  0.0  0.0      0     0 ?        I<   15:59   0:00 [rcu_gp]
root           4  0.0  0.0      0     0 ?        I<   15:59   0:00 [rcu_par_gp]
root           5  0.0  0.0      0     0 ?        I<   15:59   0:00 [slub_flushwq]
root           6  0.0  0.0      0     0 ?        I<   15:59   0:00 [netns]
root          10  0.0  0.0      0     0 ?        I<   15:59   0:00 [mm_percpu_wq]
root          11  0.0  0.0      0     0 ?        I    15:59   0:00 [rcu_tasks_kthread]
root          12  0.0  0.0      0     0 ?        I    15:59   0:00 [rcu_tasks_rude_kthread
root          13  0.0  0.0      0     0 ?        I    15:59   0:00 [rcu_tasks_trace_kthrea
root          14  0.0  0.0      0     0 ?        S    15:59   0:00 [ksoftirqd/0]
root          15  0.0  0.0      0     0 ?        I    15:59   0:01 [rcu_preempt]
root          16  0.0  0.0      0     0 ?        S    15:59   0:00 [migration/0]
root          17  0.0  0.0      0     0 ?        S    15:59   0:00 [idle_inject/0]
root          19  0.0  0.0      0     0 ?        S    15:59   0:00 [cpuhp/0]
root          20  0.0  0.0      0     0 ?        S    15:59   0:00 [cpuhp/1]
root          21  0.0  0.0      0     0 ?        S    15:59   0:00 [idle_inject/1]
root          22  0.0  0.0      0     0 ?        S    15:59   0:00 [migration/1]
```

```
vboxuser@ubuntu:~/Desktop/jarvis/New$ top

top - 16:38:45 up 39 min,  1 user,  load average: 0.01, 0.05, 0.07
Tasks: 188 total,   1 running, 187 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.7 us,  0.6 sy,  0.0 ni, 98.4 id,  0.1 wa,  0.0 hi,  0.2 si,  0.0 st
MiB Mem :   2967.2 total,    860.2 free,    819.6 used,   1287.4 buff/cache
MiB Swap:   2680.0 total,   2680.0 free,      0.0 used.   1955.6 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
   1524 vboxuser  20   0 5084716 386076 143424 S   6.0  12.7   2:21.10 gnome-shell
   2691 vboxuser  20   0  874580  66708  50404 S   1.3   2.2   0:08.68 gnome-terminal-
     64 root      20   0       0      0      0 I   0.7   0.0   0:02.97 kworker/u6:5-events_freezable_power_
      1 root      20   0  167796  12852   8244 S   0.0   0.4   0:01.90 systemd
      2 root      20   0       0      0      0 S   0.0   0.0   0:00.02 kthreadd
      3 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_gp
      4 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_par_gp
      5 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 slub_flushwq
      6 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 netns
     10 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 mm_percpu_wq
     11 root      20   0       0      0      0 I   0.0   0.0   0:00.00 rcu_tasks_kthread
```

```
vboxuser@ubuntu:~/Desktop/jarvis/New$ nice
0
vboxuser@ubuntu:~/Desktop/jarvis/New$ renice
```

```
vboxuser@ubuntu:~/Desktop/jarvis/New$ pstree
systemd─┬─ModemManager───2*[{ModemManager}]
        ├─NetworkManager───2*[{NetworkManager}]
        ├─accounts-daemon───2*[{accounts-daemon}]
        ├─acpid
        ├─avahi-daemon───avahi-daemon
        ├─colord───2*[{colord}]
        ├─cron
        ├─cups-browsed───2*[{cups-browsed}]
        ├─cupsd
        ├─dbus-daemon
        ├─gdm3─┬─gdm-session-wor─┬─gdm-wayland-ses─┬─gnome-session-b───2*[{gnome-session-b}]
        │      │                 │                 └─2*[{gdm-wayland-ses}]
        │      │                 └─2*[{gdm-session-wor}]
        │      └─2*[{gdm3}]
        ├─gnome-keyring-d───3*[{gnome-keyring-d}]
        ├─irqbalance───{irqbalance}
        ├─2*[kerneloops]
        ├─networkd-dispat
        ├─packagekitd───2*[{packagekitd}]
        ├─polkitd───2*[{polkitd}]
        ├─power-profiles-───2*[{power-profiles-}]
        ├─rsyslogd───3*[{rsyslogd}]
        ├─rtkit-daemon───2*[{rtkit-daemon}]
        ├─snapd───14*[{snapd}]
        ├─switcheroo-cont───2*[{switcheroo-cont}]
        ├─systemd─┬─(sd-pam)
        │         ├─at-spi2-registr───2*[{at-spi2-registr}]
        │         ├─dbus-daemon
        │         ├─dconf-service───2*[{dconf-service}]
```

Prog 7: To understand how to examine and change File permissions.

In Linux, file permissions determine who can access a file or directory and what actions they can perform (read, write, execute). Here's how you can examine and change file permissions using the chmod command:

Common Permission Values:

- Read (r):
  - Allows reading the contents of the file.
- Write (w):
  - Allows modifying the file or creating new files in a directory.
- Execute (x):
  - Allows running the file as a program or entering a directory.

Examining File Permissions:

Viewing Permissions:

Use the ls -l command to list files with detailed information. The

output will display the permissions along with other details. ls -l

filename

Example output:

-rw-r--r-- 1 user user 1234 Dec 1 10:00

filename The permissions are represented by

rw-r--r--.

The first character (- in the example) indicates the type of file (regular file).

The next three characters (rw-) represent the owner's permissions.

The next three characters (r--) represent the group's permissions.

The last three characters (r--) represent others' (everyone else) permissions.

In this example, the owner can read and write, the group can read, and others can read.

Changing File Permissions:

Using Numeric Representation:

The chmod command can be used with numeric

representations. chmod 644 filename

The numeric representation 644 corresponds to the permissions rw-r--r--,

where: Owner has read and write (6).

Group has read (4).

Others have read

(4).

Using Symbolic Representation:

The chmod command can also use symbolic

representation. chmod u+x filename

This adds execute permission to the

owner. chmod o-r filename

This removes read permission from others.

Important Flags:

u: User/Owner

g: Group

o: Others

+: Add permission

-: Remove permission

=: Set permission explicitly

Example:

chmod u=rw,g=r,o=r filename

This sets explicit permissions for the owner, group, and others.

Prog 8: Set a file to be read-only with the chmod command. Interpret the file CO4 Bachelor of Computer Applications permissions displayed by the ls -l command.

```
onworks@onworks-Standard-PC-i440FX-PIIX-1996:~$ chmod 444 fileone.txt
onworks@onworks-Standard-PC-i440FX-PIIX-1996:~$ ls -l fileone.txt
 -r--r--r-- 1 vboxuser vboxuser 9382 Dec  2 16:31 fileone.txt
```

Interpretation:

The first character (-) indicates that this is a regular file.
The next three characters (r--) represent read-only permissions for the owner.
The following three characters (r--) represent read-only permissions for the group. The last three characters (r--) represent read-only permissions for others.
The number 1 indicates that there is one hard link to this file.
"user" is the owner of the file, and "group" is the group associated with the file.
The file size is 9382 bytes.
The file was last modified on Dec 2 at 16:31.

Q9 Delete one or more directories with the rmdir command. See what happens if the directory is not empty. Experiment (carefully!) with the rm -r command to delete a directory and its content.

```
lynx@lynx:~/test$ ls
empty-dir  file.txt  non-empty-dir
lynx@lynx:~/test$ rmdir non-empty-dir/
rmdir: failed to remove 'non-empty-dir/': Directory not empty
lynx@lynx:~/test$ rmdir empty-dir/
lynx@lynx:~/test$
```

Q10 Change your directory to the directory exercises. Create a file in that directory, named the file as example using the cat command containing the following text: water, water everywhere and all the boards did shrink; water, water everywhere, no drop to drink.

```
lynx@lynx:~/test$ mkdir exercises
lynx@lynx:~/test$ cat > exercises/example1 << EOF
water, water everywhere and all the boards did shrink;
water, water everywhere, no drop to drink.
EOF
lynx@lynx:~/test$ cat exercises/example1
water, water everywhere and all the boards did shrink;
water, water everywhere, no drop to drink.
lynx@lynx:~/test$
```

11. Write basic shell script to display the table of a number.

```
#!/bin/bash

# Check if the number of arguments is correct
if [ "$#" -ne 1 ]; then
    echo "Usage: $0 <number>"
    exit 1
fi

# Get the number from command line argument
number=$1

# Display the table header
echo "Multiplication Table of $number:"

# Use a loop to calculate and display the table
for i in {1..10}; do
    result=$((number * i))
    echo "$number x $i = $result"
done
```

```
Multiplication Table of 12:
12 x 1 = 12
12 x 2 = 24
12 x 3 = 36
12 x 4 = 48
12 x 5 = 60
12 x 6 = 72
12 x 7 = 84
12 x 8 = 96
12 x 9 = 108
12 x 10 = 120
```

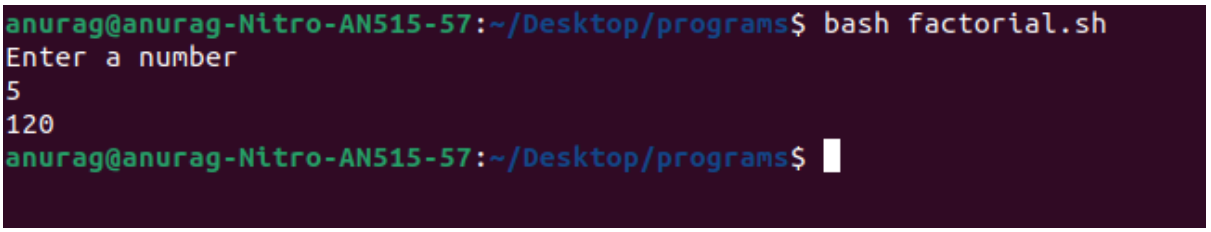
12.Write basic shell script to input a character from user and then check whether it is uppercase, lowercase or digit

```
#!/bin/bash

# Prompt user for input
read -p "Enter a character: " char

# Check if the input is a single character
if [ ${#char} -ne 1 ]; then
    echo "Please enter a single character."
    exit 1
fi

# Check if the character is an uppercase letter
if [[ $char =~ ^[A-Z]$ ]]; then
    echo "The entered character is an uppercase letter."
# Check if the character is a lowercase letter
elif [[ $char =~ ^[a-z]$ ]]; then
    echo "The entered character is a lowercase letter."
# Check if the character is a digit
elif [[ $char =~ ^[0-9]$ ]]; then
    echo "The entered character is a digit."
# If none of the above, it is something else
else
    echo "The entered character is neither uppercase nor lowercase letter nor a
digit."
fi
```

```
Enter a character: t
The entered character is a lowercase letter.
```

Prog 13: Write basic shell script to calculate factorial of a number.

```
echo "Enter a number"
read num
fact=1
while [ $num -gt 1 ]
do
  fact=$((fact * num))  #fact = fact * num
  num=$((num - 1))      #num = num - 1
done
echo $fact
```



Q.14) Write basic shell script to input the month number and generate corresponding calendar

read months

month=$1

cal $1

```
~/Desktop/os
❯ ./11v2.sh
12
   December 2023
Su Mo Tu We Th Fr Sa
                  1  2
 3  4  5  6  7  8  9
10 11 12 13 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28 29 30
31
```

Q.15) Write basic shell script to list all directories

directories=$(ls -ld */)

echo "List of Directories:"

echo "$directories"

```
~/Desktop                                                    at 16:11:09
❯ ./12.sh
List of Directories:
drwx------ 3 siddharth siddharth 4096 Mar  6  2023 BaseTools/
drwxrwxr-x 4 siddharth siddharth 4096 Nov 14  2022 C++_Programs/
drwxrwxr-x 4 siddharth siddharth 4096 Jan 16  2023 LibreOffice_Bug_Patches/
drwxrwxr-x 3 siddharth siddharth 4096 Feb 23  2023 LibreOffice_GSoC_Ideas/
drwxrwxr-x 6 siddharth siddharth 4096 Apr 28  2023 misc1/
drwxrwxr-x 2 siddharth siddharth 4096 Apr 27  2023 misc2/
drwxrwxr-x 3 siddharth siddharth 4096 Mar 23  2023 miscellanous/
drwxrwxr-x 2 siddharth siddharth 4096 Dec  4 16:09 os/
drwxrwxr-x 5 siddharth siddharth 4096 Jan 10  2023 re-lab/
drwxrwxr-x 9 siddharth siddharth 4096 Nov 13  2022 simian/
```

Q.16) Write basic shell script to display greatest of three numbers

```
echo -n "Enter the first number: "
read num1
```

```
echo -n "Enter the second number: "
read num2
echo -n "Enter the third number: "
read num3
# Using if statements to find the greatest number
if [ $num1 -gt $num2 ] && [ $num1 -gt $num3 ]; then
    greatest=$num1
elif [ $num2 -gt $num1 ] && [ $num2 -gt $num3 ]; then
    greatest=$num2
else
    greatest=$num3
fi
# Displaying the greatest number
echo "The greatest number is: $greatest"
```

Q.17)Write basic shell script to check whether number entered by user is prime or not

```
# Prompt the user to enter a number
echo -n "Enter a number: "
read user_number
# Validate if the input is a positive integer
if ! [[ "$user_number" =~ ^[1-9][0-9]*$ ]]; then
    echo "Invalid input. Please enter a positive integer."
    exit 1
fi
# Function to check if a number is prime
is_prime() {
    num=$1
    if [ $num -lt 2 ]; then
```

```
        echo "$num is not a prime number."
        return
    fi
    for ((i = 2; i <= num / 2; i++)); do
        if [ $((num % i)) -eq 0 ]; then
            echo "$num is not a prime number."
            return
        fi
    done
    echo "$num is a prime number."
}
# Call the is_prime function with the user's input
is_prime $user_number
```