

## ITW1 ASSIGNMENT-2

NAME :- TARUN ARORA

ROLL NO. :- 20075092

BRANCH :- CSE

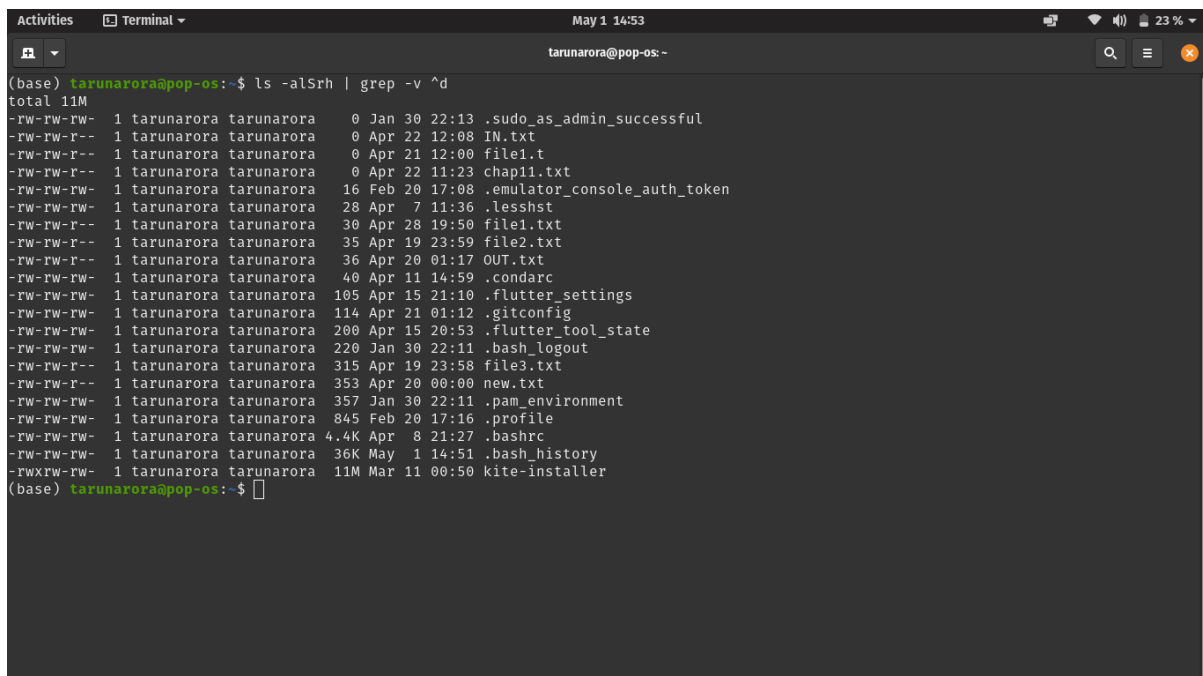
Q1. Write a command to find the list of all the files in the current directory in long format, sorted by size, and smallest first.

Solution:-

Code:-

```
ls -alSh | grep -v ^d
```

(to display only files)



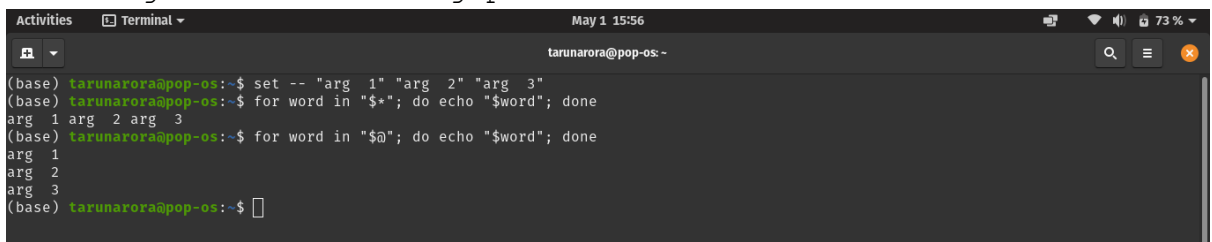
```
Activities Terminal May 1 14:53 tarunarora@pop-os: ~
(base) tarunarora@pop-os:~$ ls -alSh | grep -v ^d
total 11M
-rw-rw-rw- 1 tarunarora tarunarora  0 Jan 30 22:13 .sudo_as_admin_successful
-rw-rw-r-- 1 tarunarora tarunarora  0 Apr 22 12:08 IN.txt
-rw-rw-r-- 1 tarunarora tarunarora  0 Apr 21 12:00 file1.t
-rw-rw-r-- 1 tarunarora tarunarora  0 Apr 22 11:23 chap11.txt
-rw-rw-rw- 1 tarunarora tarunarora 16 Feb 20 17:08 .emulator_console_auth_token
-rw-rw-rw- 1 tarunarora tarunarora 28 Apr  7 11:36 .lessshst
-rw-rw-r-- 1 tarunarora tarunarora 30 Apr 28 19:50 file1.txt
-rw-rw-r-- 1 tarunarora tarunarora 35 Apr 19 23:59 file2.txt
-rw-rw-r-- 1 tarunarora tarunarora 36 Apr 20 01:17 OUT.txt
-rw-rw-rw- 1 tarunarora tarunarora 40 Apr 11 14:59 .condarc
-rw-rw-rw- 1 tarunarora tarunarora 105 Apr 15 21:10 .flutter_settings
-rw-rw-rw- 1 tarunarora tarunarora 114 Apr 21 01:12 .gitconfig
-rw-rw-rw- 1 tarunarora tarunarora 200 Apr 15 20:53 .flutter_tool_state
-rw-rw-rw- 1 tarunarora tarunarora 220 Jan 30 22:11 .bash_logout
-rw-rw-r-- 1 tarunarora tarunarora 315 Apr 19 23:58 file3.txt
-rw-rw-r-- 1 tarunarora tarunarora 353 Apr 20 00:00 new.txt
-rw-rw-rw- 1 tarunarora tarunarora 357 Jan 30 22:11 .pam_environment
-rw-rw-rw- 1 tarunarora tarunarora 845 Feb 20 17:16 .profile
-rw-rw-rw- 1 tarunarora tarunarora 4.4K Apr  8 21:27 .bashrc
-rw-rw-rw- 1 tarunarora tarunarora 36K May  1 14:51 .bash_history
-rwxrwxrwx 1 tarunarora tarunarora 11M Mar 11 00:50 kite-installer
(base) tarunarora@pop-os:~$
```

**Q2. What is the difference between \$\* and \$@?  
How do you find whether your system is 32  
bit or 64 bit ?**

`$*` Stores all the arguments that were entered on the command line (`$1 $2 ...`).

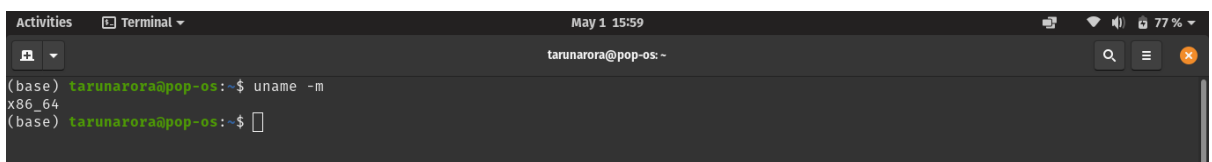
`"$@"` Stores all the arguments that were entered on the command line, individually quoted (`"$1" "$2" ...`).

They are special parameters that allow accessing all the command-line arguments at once. `$*` and `$@` both will act the same unless they are enclosed in double quotes, `" "`. However, the `"$*"` special parameter takes the entire list as one argument with spaces between and the `"$@"` special parameter takes the entire list and separates it into separate arguments, i.e. `$*` takes value as a string whereas `$@` has a value of array. `$*` treats command line arguments as a single string. `$@` treats each argument as a string and uses string polarisation .

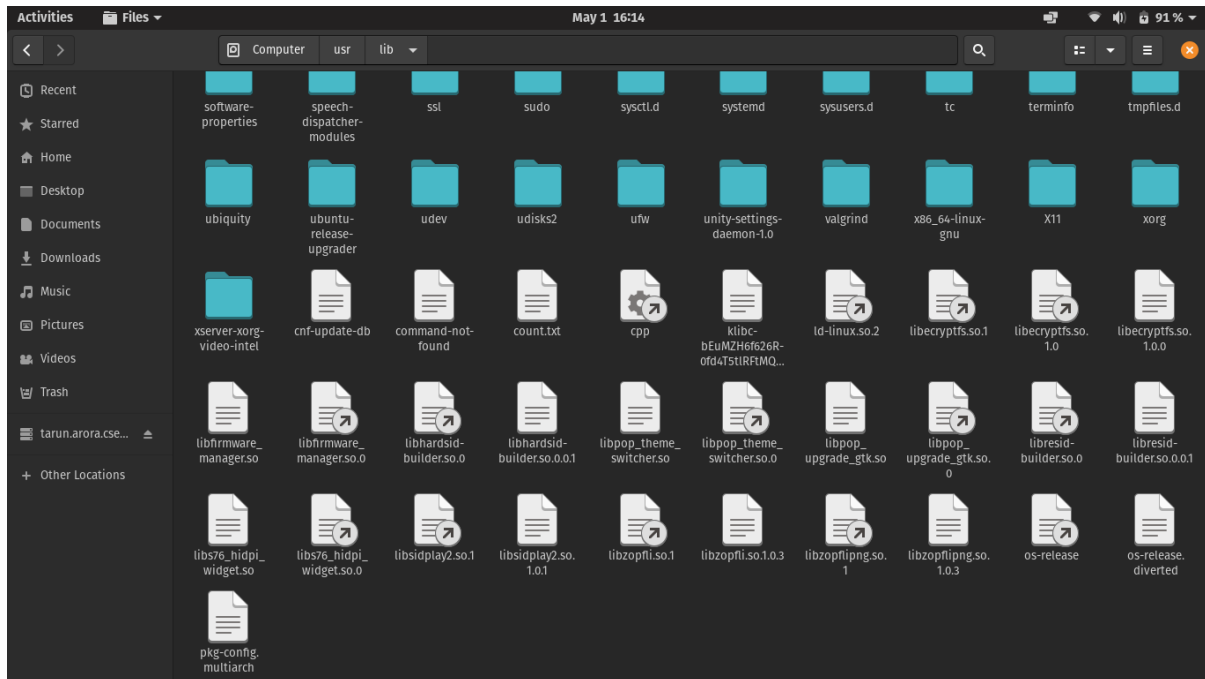
A terminal window titled 'tarunarora@pop-os -' showing a series of commands and their outputs. The commands are: 1. 'set -- "arg 1" "arg 2" "arg 3"' which sets three arguments. 2. 'for word in "\$\*"; do echo "\$word"; done' which iterates over the arguments as a single string, outputting 'arg 1 arg 2 arg 3'. 3. 'for word in "\$@"; do echo "\$word"; done' which iterates over the arguments as individual strings, outputting 'arg 1', 'arg 2', and 'arg 3' on separate lines. The prompt is '(base) tarunarora@pop-os:~\$'.

Code to get find whether system is 32/64 bit.

**`uname -m`**

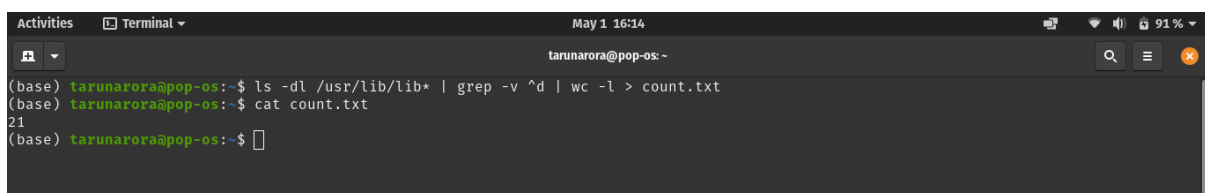
A terminal window titled 'tarunarora@pop-os -' showing the command 'uname -m' being executed. The output is 'x86\_64'. The prompt is '(base) tarunarora@pop-os:~\$'.

**Q3.** Count the total number of files in the directory `/usr/lib` starting with 'lib' and print output in a `count.txt` file.



Code to count number of "files" in `/usr/lib`:-

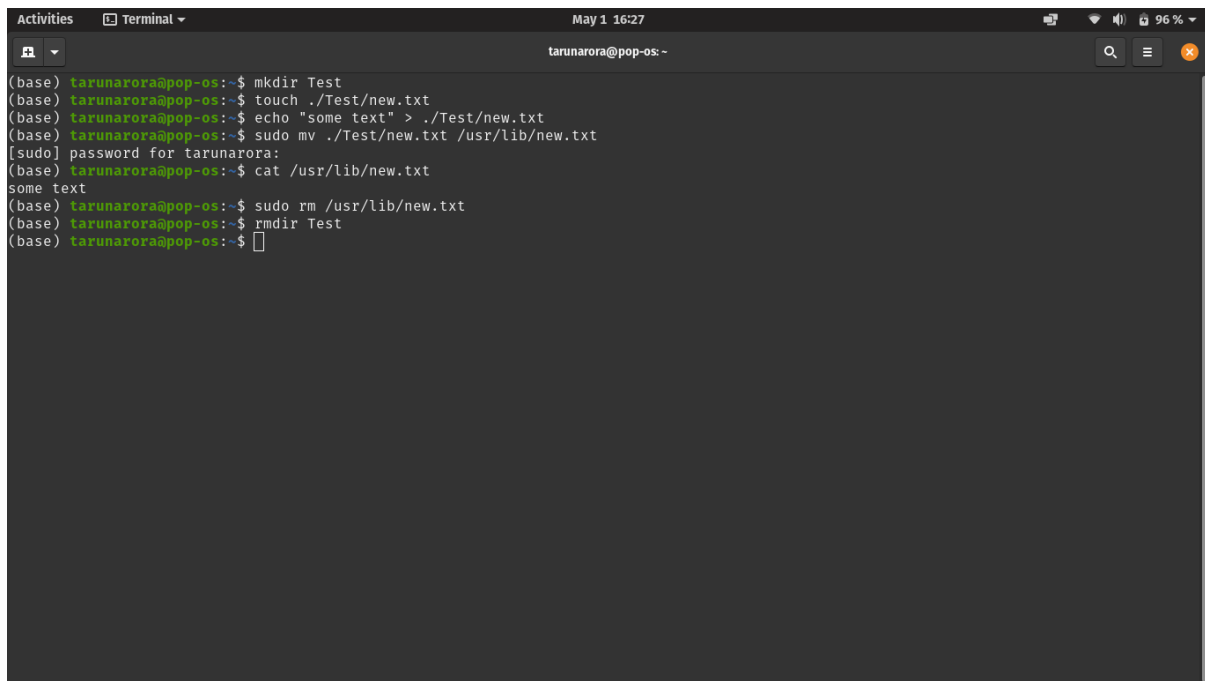
```
ls -dl /usr/lib/lib* | grep -v ^d | wc -l > count.txt
```



**Q4.** From your home directory, create a directory named "Test". Create a new file in Test directory named 'new.txt' and write some text in it. Then, move the file from there to "/usr/lib/". Then, display the content of that file on the terminal and then delete the file "new.txt" and the folder "Test".

**Solution:-**

Code is attached in the screenshot below.

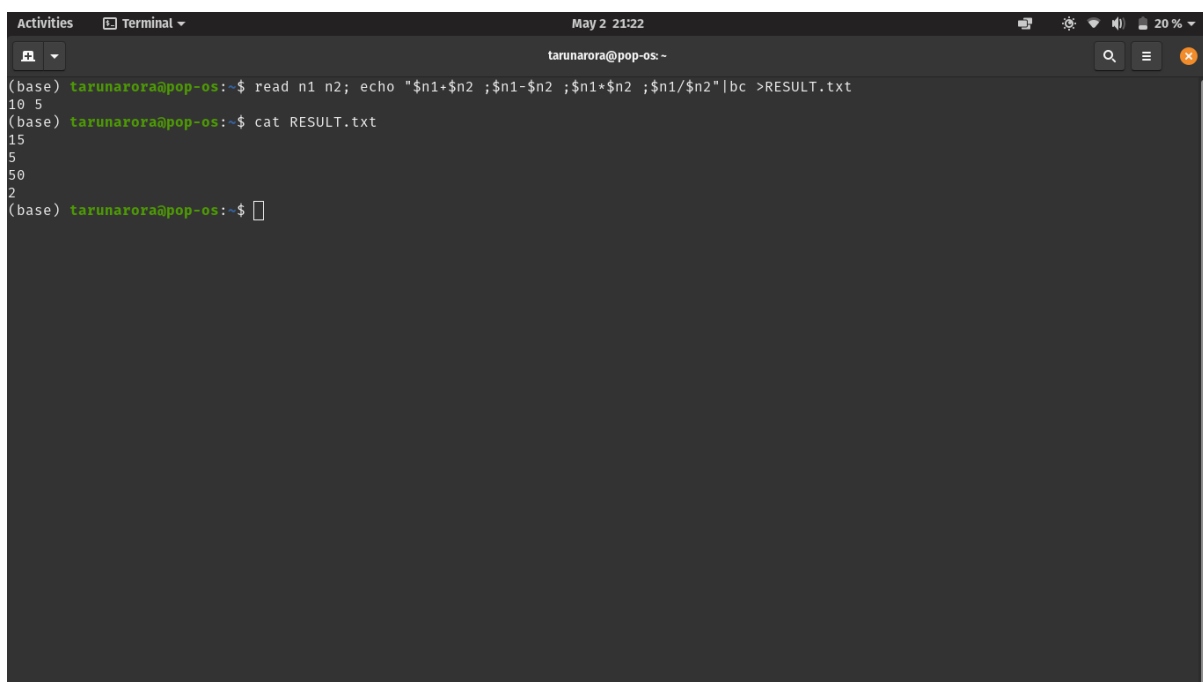


```
(base) tarunarora@pop-os:~$ mkdir Test
(base) tarunarora@pop-os:~$ touch ./Test/new.txt
(base) tarunarora@pop-os:~$ echo "some text" > ./Test/new.txt
(base) tarunarora@pop-os:~$ sudo mv ./Test/new.txt /usr/lib/new.txt
[sudo] password for tarunarora:
(base) tarunarora@pop-os:~$ cat /usr/lib/new.txt
some text
(base) tarunarora@pop-os:~$ sudo rm /usr/lib/new.txt
(base) tarunarora@pop-os:~$ rmdir Test
(base) tarunarora@pop-os:~$
```

Q5. Read two numbers and perform addition, subtraction, multiplication and division. Also save the output of the operations in RESULT.txt.

Solution:-

Code is attached in the screenshot below.

A terminal window titled 'Terminal' with a date and time of 'May 2 21:22'. The user is 'tarunarora@pop-os'. The terminal shows a sequence of commands and their outputs. First, a command is entered: `(base) tarunarora@pop-os:~$ read n1 n2; echo "$n1+$n2 ;$n1-$n2 ;$n1*$n2 ;$n1/$n2"|bc >RESULT.txt`. The output is: `10 5`. Then, the command `(base) tarunarora@pop-os:~$ cat RESULT.txt` is entered, and the output is: `15`, `5`, `50`, `2`. The prompt `(base) tarunarora@pop-os:~$` is shown again at the end.

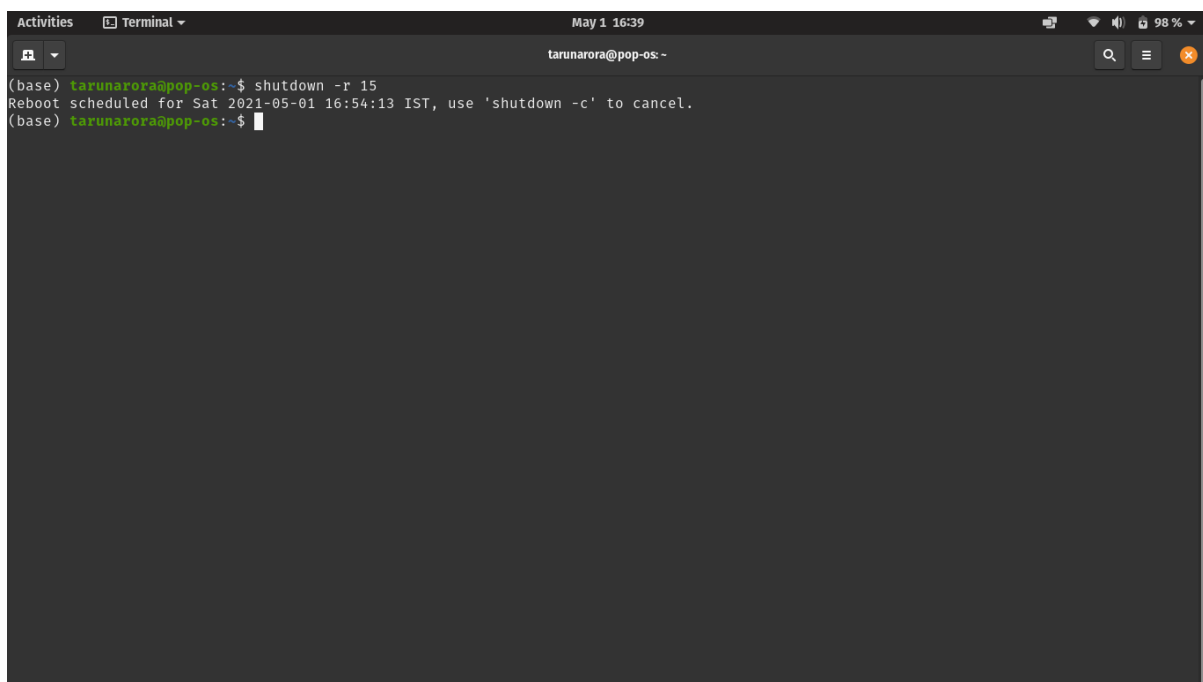
```
(base) tarunarora@pop-os:~$ read n1 n2; echo "$n1+$n2 ;$n1-$n2 ;$n1*$n2 ;$n1/$n2"|bc >RESULT.txt
10 5
(base) tarunarora@pop-os:~$ cat RESULT.txt
15
5
50
2
(base) tarunarora@pop-os:~$
```

**Q6.** Write a command that will allow a UNIX system to shut down in 15 minutes, after which it will perform a reboot.

**Solution:-**

**Code:-**

```
shutdown -r 15
```

A screenshot of a Linux terminal window. The window title is "Terminal" and it shows the user "tarunarora@pop-os" at the prompt. The user has entered the command "shutdown -r 15". The terminal output shows the command being executed and a confirmation message: "Reboot scheduled for Sat 2021-05-01 16:54:13 IST, use 'shutdown -c' to cancel." The prompt returns to the user, indicating the command was successful.

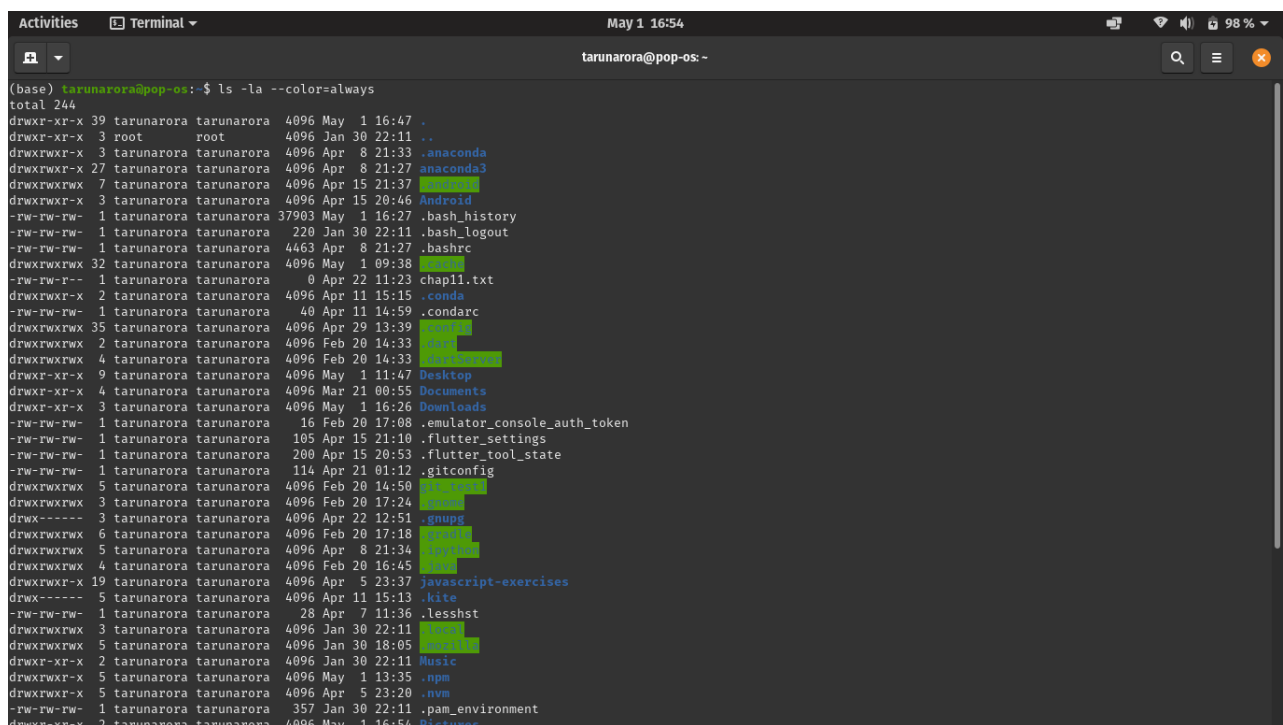
```
(base) tarunarora@pop-os:~$ shutdown -r 15
Reboot scheduled for Sat 2021-05-01 16:54:13 IST, use 'shutdown -c' to cancel.
(base) tarunarora@pop-os:~$
```

**Q7.** Write a command that will display files in the current directory, in a colored, long format.

**Solution:-**

**Code:-**

```
ls -la -color=always
```



The image shows a terminal window titled 'Terminal' with the date 'May 1 16:54' and the user 'tarunara@pop-os'. The command '(base) tarunara@pop-os:~\$ ls -la --color=always' has been executed. The output is a long-format listing of files and directories in the current directory, with permissions, owner, group, size, date, and filename. Files are color-coded: directories are blue, executables are green, and regular files are white. The output is as follows:

```
total 244
drwxr-xr-x 39 tarunara tarunara 4096 May 1 16:47 .
drwxr-xr-x 3 root root 4096 Jan 30 22:11 ..
drwxrwxr-x 3 tarunara tarunara 4096 Apr 8 21:33 .anaconda
drwxrwxr-x 27 tarunara tarunara 4096 Apr 8 21:27 anaconda3
drwxrwxr-x 7 tarunara tarunara 4096 Apr 15 21:37 android
drwxrwxr-x 3 tarunara tarunara 4096 Apr 15 20:46 Android
-rw-rw-rw- 1 tarunara tarunara 37903 May 1 16:27 .bash_history
-rw-rw-rw- 1 tarunara tarunara 220 Jan 30 22:11 .bash_logout
-rw-rw-rw- 1 tarunara tarunara 4463 Apr 8 21:27 .bashrc
drwxrwxr-x 32 tarunara tarunara 4096 May 1 09:38 .cache
-rw-rw-r-- 1 tarunara tarunara 0 Apr 22 11:23 chap11.txt
drwxrwxr-x 2 tarunara tarunara 4096 Apr 11 15:15 .conda
-rw-rw-rw- 1 tarunara tarunara 40 Apr 11 14:59 .condarc
drwxrwxr-x 35 tarunara tarunara 4096 Apr 29 13:39 .conda
drwxrwxr-x 2 tarunara tarunara 4096 Feb 20 14:33 .dev
drwxrwxr-x 4 tarunara tarunara 4096 Feb 20 14:33 .dev
drwxr-xr-x 9 tarunara tarunara 4096 May 1 11:47 Desktop
drwxr-xr-x 4 tarunara tarunara 4096 Mar 21 00:55 Documents
drwxr-xr-x 3 tarunara tarunara 4096 May 1 16:26 Downloads
-rw-rw-rw- 1 tarunara tarunara 16 Feb 20 17:08 .emulator_console_auth_token
-rw-rw-rw- 1 tarunara tarunara 105 Apr 15 21:10 .flutter_settings
-rw-rw-rw- 1 tarunara tarunara 200 Apr 15 20:53 .flutter_tool_state
-rw-rw-rw- 1 tarunara tarunara 114 Apr 21 01:12 .gitconfig
drwxrwxr-x 5 tarunara tarunara 4096 Feb 20 14:50 .github
drwxrwxr-x 3 tarunara tarunara 4096 Feb 20 17:24 .gnupg
drwx----- 3 tarunara tarunara 4096 Apr 22 12:51 .gnupg
drwxrwxr-x 6 tarunara tarunara 4096 Feb 20 17:18 .gnupg
drwxrwxr-x 5 tarunara tarunara 4096 Apr 8 21:34 .gnupg
drwxrwxr-x 4 tarunara tarunara 4096 Feb 20 16:45 .gnupg
drwxrwxr-x 19 tarunara tarunara 4096 Apr 5 23:37 javascript-exercises
drwx----- 5 tarunara tarunara 4096 Apr 11 15:13 .kite
-rw-rw-rw- 1 tarunara tarunara 28 Apr 7 11:36 .lessht
drwxrwxr-x 3 tarunara tarunara 4096 Jan 30 22:11 .lessht
drwxrwxr-x 5 tarunara tarunara 4096 Jan 30 18:05 .lessht
drwxr-xr-x 2 tarunara tarunara 4096 Jan 30 22:11 Music
drwxrwxr-x 5 tarunara tarunara 4096 May 1 13:35 .npm
drwxrwxr-x 5 tarunara tarunara 4096 Apr 5 23:20 .npm
-rw-rw-rw- 1 tarunara tarunara 357 Jan 30 22:11 .pam_environment
drwxr-xr-x 2 tarunara tarunara 4096 May 1 16:54 Pictures
```

**Q8.** What is the behavioral difference between "cmp" and "diff" commands? Justify with an example.

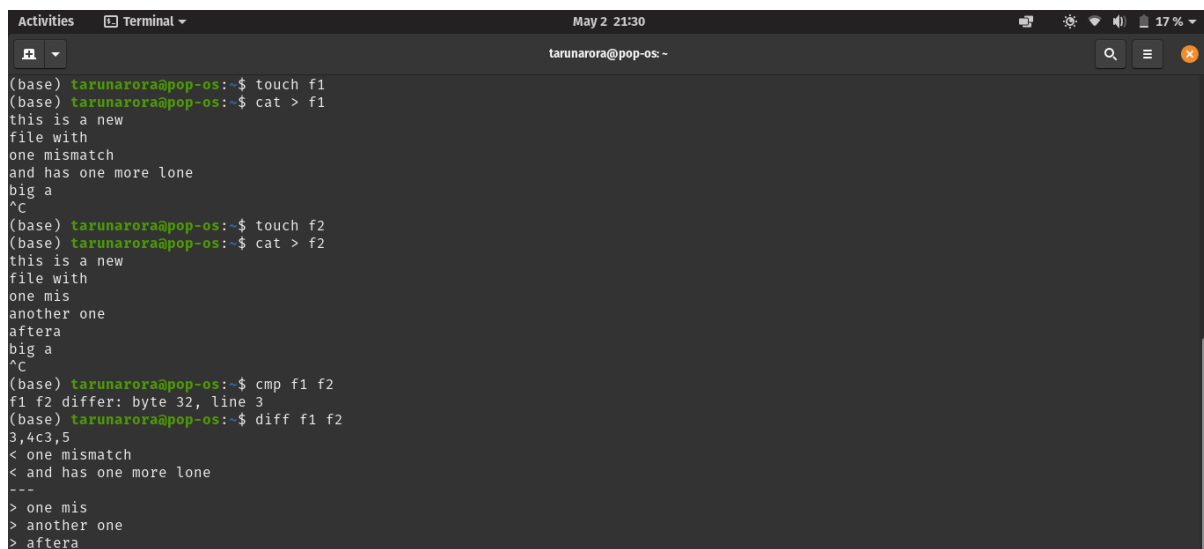
**Solution:-**

→ "cmp" command :

- "cmp" command is a simple file comparison which compares two files bytes by bytes and tells if the file are identical or not.
- If mismatch occurs it returns the byte at which first mismatch occur else if both the file are same then no output occurs and prompt is returned

→ "diff" command:

- "diff" command is compares two files line by line and tells which line needs to be changed in order to make the two files equal.
- If mismatch occurs it reports all of the mismatched lines else if both the files are same then no output is given and prompt is returned.



```
(base) tarunarora@pop-os:~$ touch f1
(base) tarunarora@pop-os:~$ cat > f1
this is a new
file with
one mismatch
and has one more lone
big a
^C
(base) tarunarora@pop-os:~$ touch f2
(base) tarunarora@pop-os:~$ cat > f2
this is a new
file with
one mis
another one
aftera
big a
^C
(base) tarunarora@pop-os:~$ cmp f1 f2
f1 f2 differ: byte 32, line 3
(base) tarunarora@pop-os:~$ diff f1 f2
3,4c3,5
< one mismatch
< and has one more lone
---
> one mis
> another one
> aftera
```



**Q9.** Write a command to searches the line which does not start with # or single quote (') or double front slashes (//) in a given file (file lines are star with any of [A-Z], [a-z], [0-9], and [#, ', //]).

**Solution:-**

**Code:-**

```
grep "^//" f1 -v | grep -v "^[`#]"
```

A terminal window titled "Terminal" with a timestamp of "May 2 23:26" and a user "tarunarora@pop-os: ~". The terminal shows the following commands and output:

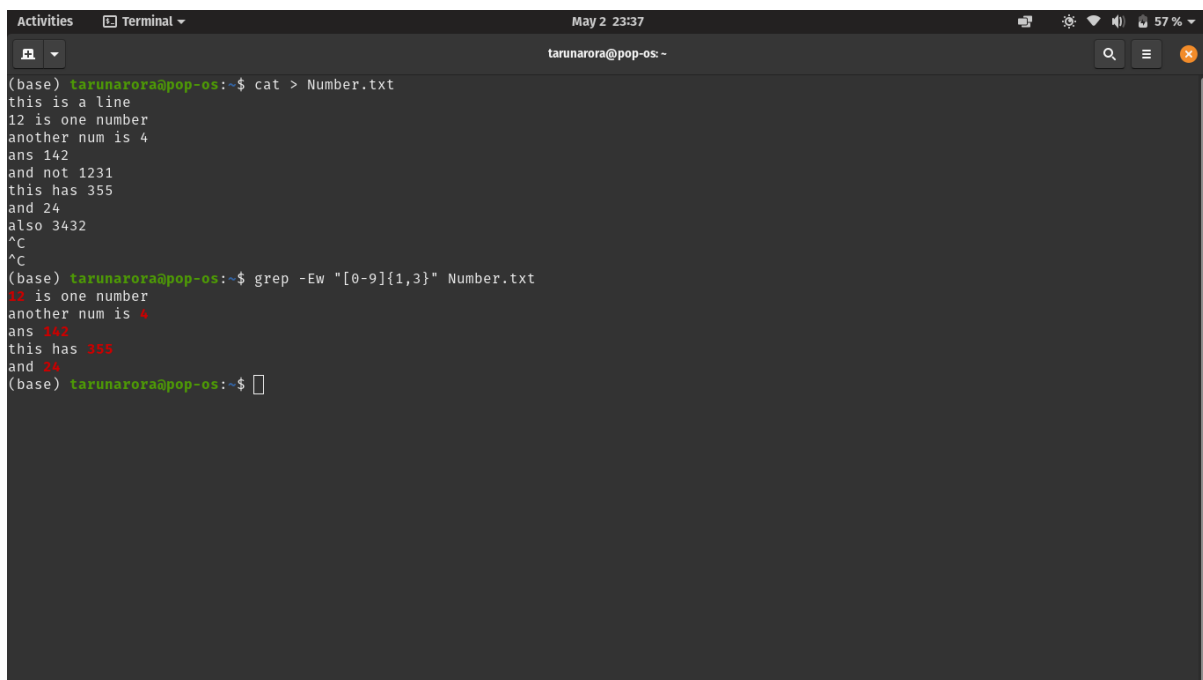
```
(base) tarunarora@pop-os:~$ cat > f1
hello
//slash line
#hash line
'quote line
normal
/hi
^C
(base) tarunarora@pop-os:~$ grep "^//" f1 -v | grep -v "^[`#]"
hello
normal
/hi
(base) tarunarora@pop-os:~$
```

**Q10.** Write a command to prints the line if its in the range of 0 to 999 from a file Number.txt. The file Number.txt contain digits in multiple lines.

**Solution:-**

**Code:-**

```
grep -Ew "[0-9]{1,3}" Number.txt
```

A terminal window titled 'Terminal' with a timestamp of 'May 2 23:37' and a user prompt 'tarunarora@pop-os: ~'. The terminal shows the following commands and output:

```
(base) tarunarora@pop-os:~$ cat > Number.txt
this is a line
12 is one number
another num is 4
ans 142
and not 1231
this has 355
and 24
also 3432
^C
^C
(base) tarunarora@pop-os:~$ grep -Ew "[0-9]{1,3}" Number.txt
12 is one number
another num is 4
ans 142
this has 355
and 24
(base) tarunarora@pop-os:~$
```

The output of the grep command shows the lines from the file that contain one or more digits, with the numbers themselves highlighted in red.

Q11. Write a command print all lines containing a vowel (a, e, i, o, or u) followed by a single character followed by the same vowel again from a file (file must have word like evening/adam) .

Solution:-

Code:-

```
grep -E "a.a|e.e|i.i|o.o|u.u" f1
```

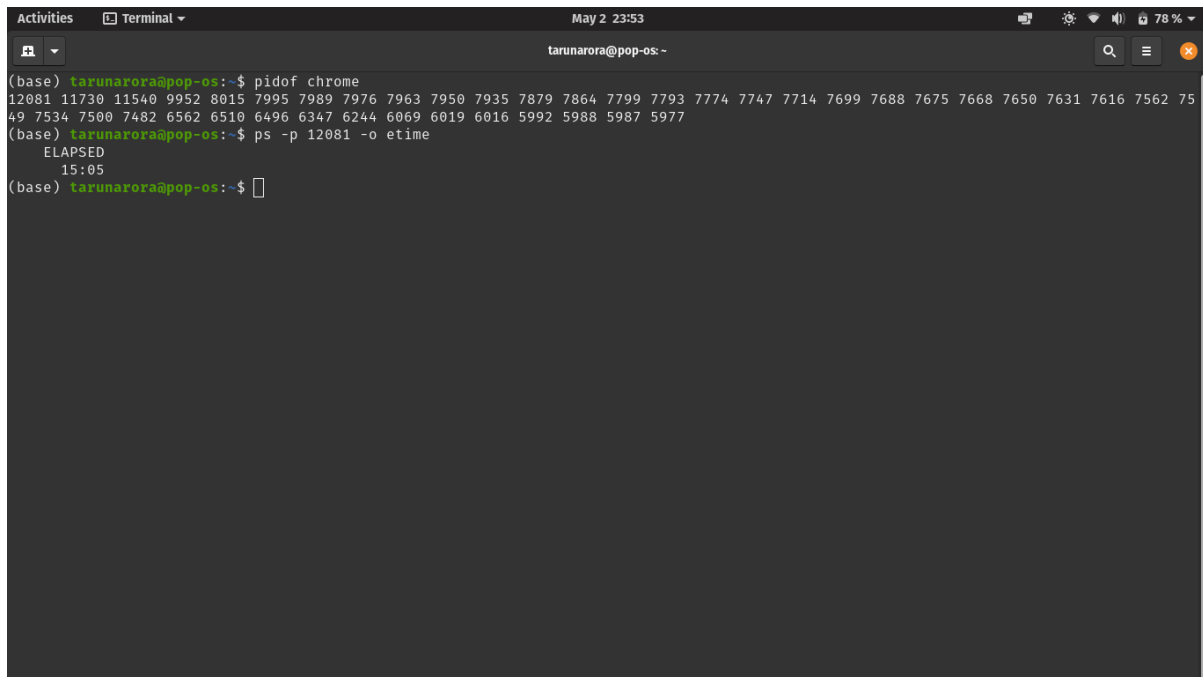
A terminal window titled 'Terminal' with a timestamp of 'May 2 23:43' and a user prompt 'tarunarora@pop-os: ~'. The terminal shows the following sequence of commands and output:  
(base) tarunarora@pop-os:~\$ cat > f1  
this line has evening  
are you awake?  
this line has nothing to say  
^C  
(base) tarunarora@pop-os:~\$ grep -E "a.a|e.e|i.i|o.o|u.u" f1  
this line has eve~~n~~ning  
are you ~~a~~awake?  
(base) tarunarora@pop-os:~\$  
The output shows that the grep command successfully identified the words 'evening' and 'awake' in the file 'f1', with the matching vowels and the following character being highlighted in red in the original image.

**Q12.** Write a command which provides the elapsed time since the process was started, in the form dd:hh:mm:ss.

**Solution:-**

**Code:-**

```
ps -p {pid} -o etime
```



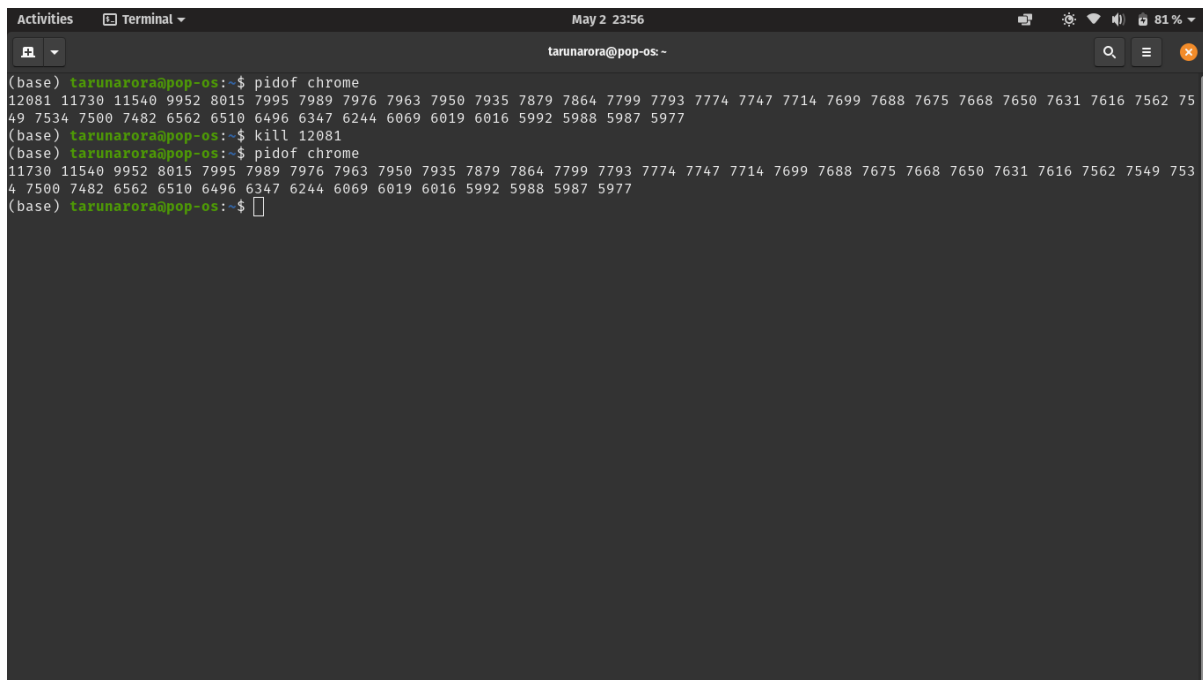
```
Activities Terminal May 2 23:53 tarunarora@pop-os: ~
(base) tarunarora@pop-os:~$ pidof chrome
12081 11730 11540 9952 8015 7995 7989 7976 7963 7950 7935 7879 7864 7799 7793 7774 7747 7714 7699 7688 7675 7668 7650 7631 7616 7562 7549 7534 7500 7482 6562 6510 6496 6347 6244 6069 6019 6016 5992 5988 5987 5977
(base) tarunarora@pop-os:~$ ps -p 12081 -o etime
ELAPSED
15:05
(base) tarunarora@pop-os:~$
```

**Q13.** Write a command to kill the specified process which are selected from all the running processes.

**Solution:-**

**Code:-**

```
kill {pid}
```

A terminal window titled 'Terminal' with a timestamp of 'May 2 23:56' and a battery level of '81%'. The user 'tarunarora@pop-os' is at the prompt. The terminal shows the following sequence of commands and outputs:  
(base) tarunarora@pop-os:~\$ pidof chrome  
12081 11730 11540 9952 8015 7995 7989 7976 7963 7950 7935 7879 7864 7799 7793 7774 7747 7714 7699 7688 7675 7668 7650 7631 7616 7562 7549 7534 7500 7482 6562 6510 6496 6347 6244 6069 6019 6016 5992 5988 5987 5977  
(base) tarunarora@pop-os:~\$ kill 12081  
(base) tarunarora@pop-os:~\$ pidof chrome  
11730 11540 9952 8015 7995 7989 7976 7963 7950 7935 7879 7864 7799 7793 7774 7747 7714 7699 7688 7675 7668 7650 7631 7616 7562 7549 7534 7500 7482 6562 6510 6496 6347 6244 6069 6019 6016 5992 5988 5987 5977  
(base) tarunarora@pop-os:~\$

**Q14.**How can you find out how long the system has been running? How to check the status of the password for the user named ITW1 ?


**Solution:-**

**Code:-**

```
uptime
```

```
sudo chage -l {username}
```

**// Note here username=ITW1**

A terminal window titled 'Terminal' with a dark background. The prompt is '(base) tarunarora@pop-os:~'. The user enters 'uptime', and the output is '23:58:34 up 2:50, 1 user, load average: 0.38, 0.45, 0.45'. The user then enters 'uptime -p', and the output is 'up 2 hours, 50 minutes'. Next, the user enters 'sudo chage -l tarunarora'. The prompt changes to '[sudo] password for tarunarora:'. The user enters a password (indicated by a square box). The output shows password expiration details: 'Last password change : Jan 30, 2021', 'Password expires : never', 'Password inactive : never', 'Account expires : never', 'Minimum number of days between password change : 0', 'Maximum number of days between password change : 99999', and 'Number of days of warning before password expires : 7'. The prompt returns to '(base) tarunarora@pop-os:~'.

**Q15.** What will the output of the command:

**`$ ps -t dev/console.`**

**Explain.**

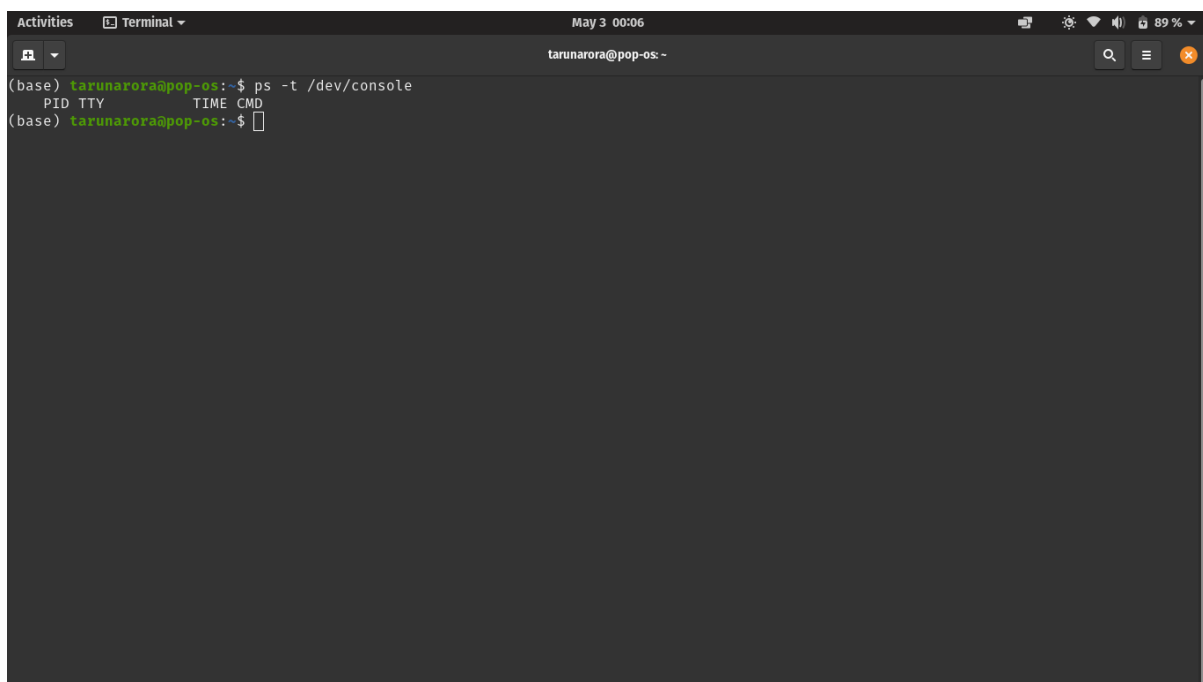
**Solution:-**

ps command gives the process status for viewing information related to processes running on the system and on using -t option it selects the processes associated with the terminals given in the tty list.

/dev/console is the system console which can be pointed to a variety of devices.

Kernel messages are logged to /dev/console where a login prompt can be launched.

As currently there are no processes linked with the /dev/console running therefore the list is empty

A screenshot of a terminal window. The title bar shows 'Activities', 'Terminal', and the date 'May 3 00:06'. The terminal content shows a user prompt '(base) tarunarora@pop-os:~\$' followed by the command 'ps -t /dev/console'. The output is a table with headers 'PID TTY' and 'TIME CMD'. Below the headers, there is a single line of output: '(base) tarunarora@pop-os:~\$' followed by a cursor. The terminal window has a dark background and a light-colored text color.

```
(base) tarunarora@pop-os:~$ ps -t /dev/console
PID TTY          TIME CMD
(base) tarunarora@pop-os:~$
```

**Q16.** Write a command to display the 10 th line from a file. Also display the first 5 lines of the file.

**Solution:-**

**Code:-**

**To display:- 10<sup>th</sup> line**

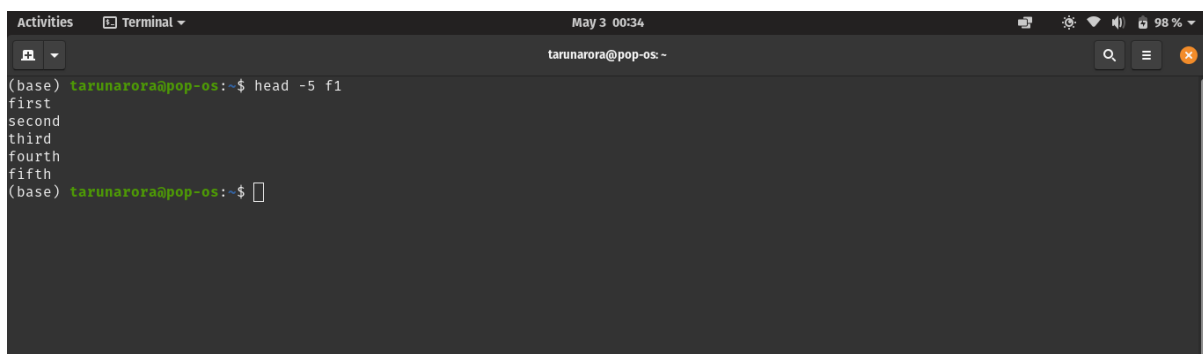
`head -10 fileName | tail -1`



```
Activities Terminal May 3 00:33 tarunarora@pop-os: -
(base) tarunarora@pop-os:~$ cat > f1
first
second
third
fourth
fifth
sixth
seventh
eighth
ninth
tenth
eleventh
^C
(base) tarunarora@pop-os:~$ head -10 f1 | tail -1
tenth
(base) tarunarora@pop-os:~$
```

**To display first 5 lines of the file:-**

`Head -5 fileName`



```
Activities Terminal May 3 00:34 tarunarora@pop-os: -
(base) tarunarora@pop-os:~$ head -5 f1
first
second
third
fourth
fifth
(base) tarunarora@pop-os:~$
```



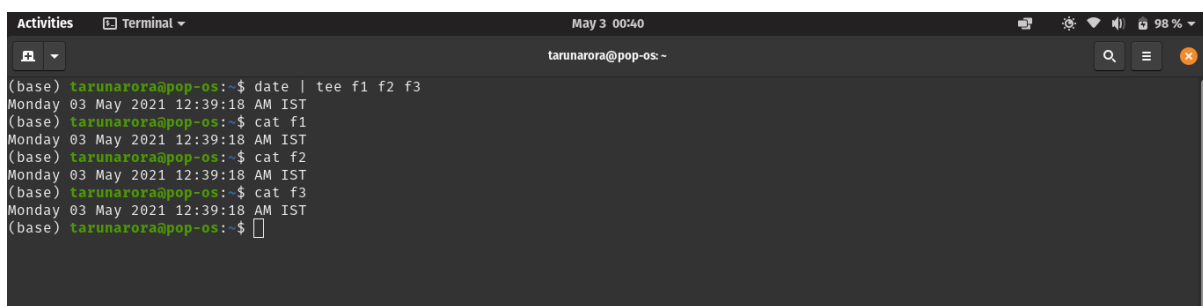
**Q17.** Write a command to redirect the output of date command to multiple files. Also how do you list the hidden files in current directory?

**Solution:-**

**Code:-**

**Redirecting output to multiple files**

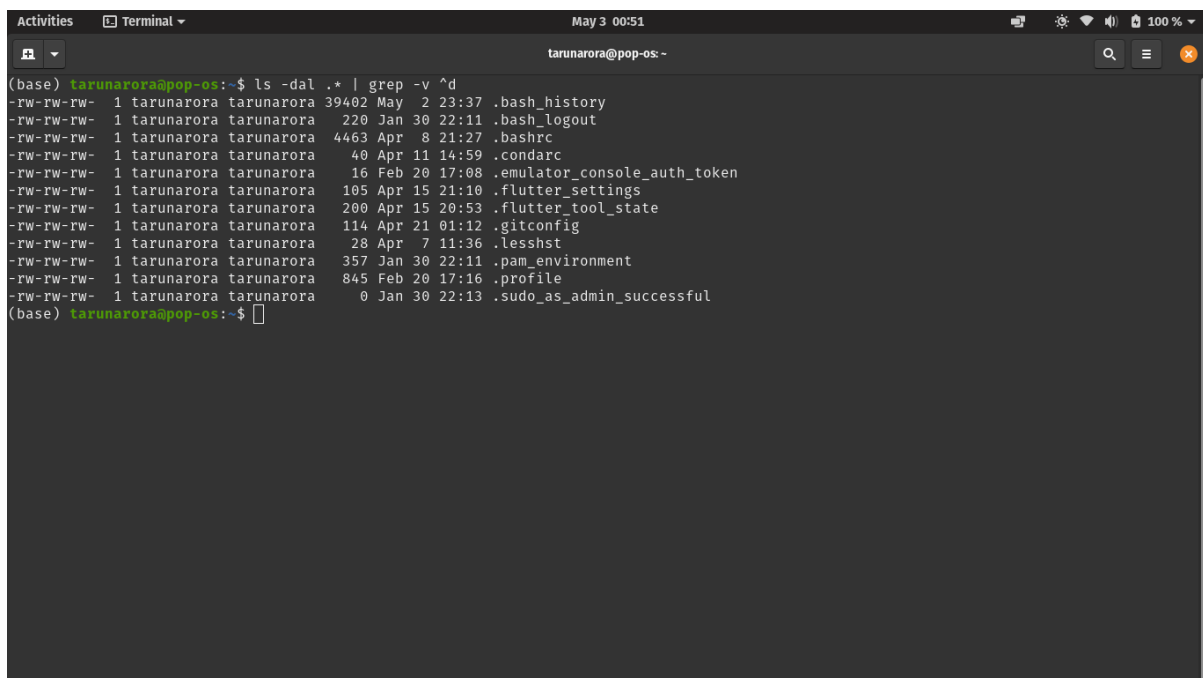
```
date | tee f1 f2 f3
```



```
Activities Terminal May 3 00:40 tarunarora@pop-os:~  
(base) tarunarora@pop-os:~$ date | tee f1 f2 f3  
Monday 03 May 2021 12:39:18 AM IST  
(base) tarunarora@pop-os:~$ cat f1  
Monday 03 May 2021 12:39:18 AM IST  
(base) tarunarora@pop-os:~$ cat f2  
Monday 03 May 2021 12:39:18 AM IST  
(base) tarunarora@pop-os:~$ cat f3  
Monday 03 May 2021 12:39:18 AM IST  
(base) tarunarora@pop-os:~$
```

**Displaying only hidden "files"**

```
ls -dal .* | grep -v ^d
```



```
Activities Terminal May 3 00:51 tarunarora@pop-os:~  
(base) tarunarora@pop-os:~$ ls -dal .* | grep -v ^d  
-rw-rw-rw- 1 tarunarora tarunarora 39402 May  2 23:37 .bash_history  
-rw-rw-rw- 1 tarunarora tarunarora  220 Jan 30 22:11 .bash_logout  
-rw-rw-rw- 1 tarunarora tarunarora  4463 Apr  8 21:27 .bashrc  
-rw-rw-rw- 1 tarunarora tarunarora   40 Apr 11 14:59 .condarc  
-rw-rw-rw- 1 tarunarora tarunarora   16 Feb 20 17:08 .emulator_console_auth_token  
-rw-rw-rw- 1 tarunarora tarunarora  105 Apr 15 21:10 .flutter_settings  
-rw-rw-rw- 1 tarunarora tarunarora  200 Apr 15 20:53 .flutter_tool_state  
-rw-rw-rw- 1 tarunarora tarunarora  114 Apr 21 01:12 .gitconfig  
-rw-rw-rw- 1 tarunarora tarunarora   28 Apr  7 11:36 .lessht  
-rw-rw-rw- 1 tarunarora tarunarora  357 Jan 30 22:11 .pam_environment  
-rw-rw-rw- 1 tarunarora tarunarora  845 Feb 20 17:16 .profile  
-rw-rw-rw- 1 tarunarora tarunarora    0 Jan 30 22:13 .sudo_as_admin_successful  
(base) tarunarora@pop-os:~$
```

Q18. Write a command to print the file names in a directory that does not contain the word "july"?

Solution:-

To display "filenames" not containing "july"

Code:-

```
ls -p --ignore=*july* | grep -v /
```

A terminal window titled "Terminal" with a date and time of "May 3 17:22". The user is logged in as "tarunarora@pop-os". The terminal shows the command "ls" being executed, which lists various files and directories including "anaconda3", "Downloads", "rock-paper-scissor-game", "Number.txt", "Android", "f1", "javascript-exercises", "okjuly", "Desktop", "f2", "julyisgood", "Pictures", "Templates", "Documents", "f3", "Music", "Public", and "Videos". The user then executes the command "ls -p --ignore=\*july\* | grep -v /", which filters out any file names containing "july" and removes the trailing slashes from directory names. The output of this command is "f1", "f2", "f3", and "Number.txt".

```
(base) tarunarora@pop-os:~$ ls
anaconda3  Downloads  rock-paper-scissor-game  Number.txt
Android    f1         javascript-exercises     okjuly
Desktop    f2         julyisgood              Pictures
Documents  f3         Music                   Public
Videos

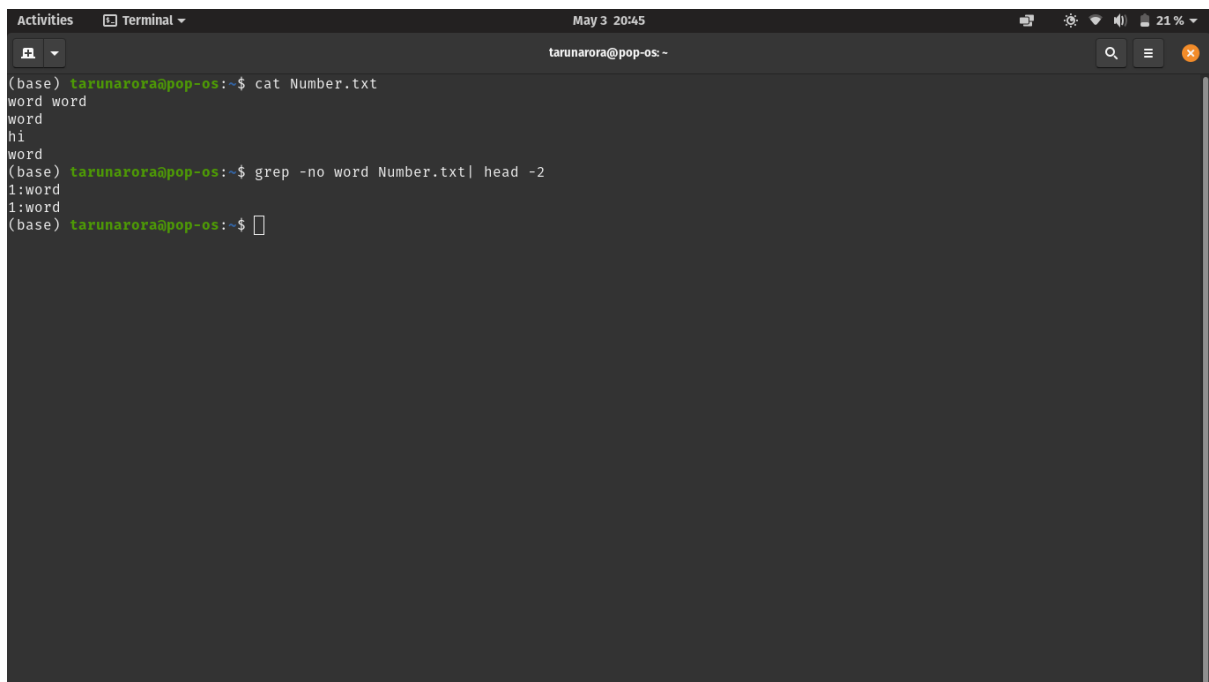
(base) tarunarora@pop-os:~$ ls -p --ignore=*july* | grep -v /
f1
f2
f3
Number.txt
(base) tarunarora@pop-os:~$
```

**Q19.** Write a command to find the first two instances of a "word" from a file. The must have multiple instances of the "word".

**Solution:-**

**Code:-**

```
grep -no word Number.txt| head -2
```



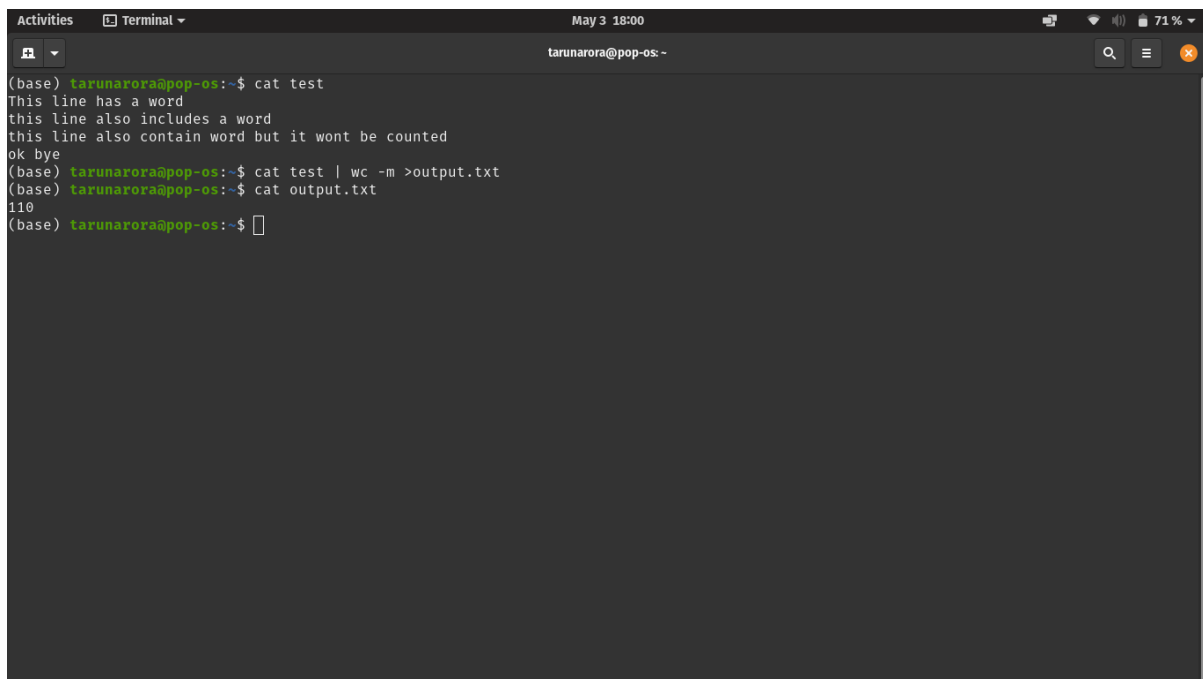
```
Activities Terminal May 3 20:45 tarunarora@pop-os: ~
(base) tarunarora@pop-os:~$ cat Number.txt
word word
word
hi
word
(base) tarunarora@pop-os:~$ grep -no word Number.txt| head -2
1:word
1:word
(base) tarunarora@pop-os:~$
```

**Q20.** Write a command to count number of characters in our file and save the output to new text file at the same time.

**Solution:-**

**Code:-**

```
cat test | wc -m > output.txt
```



```
Activities Terminal May 3 18:00 71%
tarunarora@pop-os: ~
(base) tarunarora@pop-os:~$ cat test
This line has a word
this line also includes a word
this line also contain word but it wont be counted
ok bye
(base) tarunarora@pop-os:~$ cat test | wc -m >output.txt
(base) tarunarora@pop-os:~$ cat output.txt
110
(base) tarunarora@pop-os:~$
```

\*\*\*\*\*EOF\*\*\*\*\*