Command	Meaning
command1 | command2	pipe the output of command1 to the input of command2
sort	sort data
cat file1 file2 > file0	concatenate file1 and file2 to file0
who	list users currently logged in
*	match any number of characters
?	match one character
man command	read the online manual page for a command
whatis command	brief description of a command
apropos keyword	match commands with keyword in their man pages
cp source destination	Copy a file or directory to same or different directory
rm file	Delete files and directories
date	Print the current date and time
cal	Displays calendar of the present month
grep pattern file	Searches a file for lines that have a certain pattern

Commands:

ls - list files and directories

$ ls - Document Download picture
Desktop Music Public

ls - a - list all files and directories

ls -a :- Download - Mozilla pubic
Documents music videos
.dbus .gconf .dd.txt Templates

Mkdir - Make a directory
$ mkdir rd.txt

cd directory - change to named directory
$ cd rd.txt

cd ~ - change to home directory
$ cd ~ cd ~

$d :- change to parent directory
$ cd :. $cd

Pwd - display the path of the current dictionary
$pwd :/ home/ students
students

command >file - redirect standard output to a file $ cat < add.file
dora
dd

command >> file : append standard output to a file $cat >> add.txt
dev
dharshi

command <file : redirect standard input from a file
deva
dev

dev
dHarshi

command 1 | command 2 => pipe the output of command to the input of command 2.

output : cat >kj.txt

dharshi
Guna
Priya

~ $ => cat kj.txt | dj.txt

dharshi
Guna
Priya

(13) sort - sort data

cat >> dd1.txt

30 90 50 60

sort dd1.txt

30 50 60 90

(14) cat file1 file2 > file0 - concatenate file1 & file2 to file0

cat > file.txt
12
34
56

cat > file2.txt
35
78

cat >file 3.txt
58
90
cat dd.txt
12
34
56
35
78
WHO - list users currently logged in
who.cd
who mkdir
=> list users currently logged in
txt class.cpp f1.txt
Pairs.cp.cn vab.sh
? => Match one charactor
cat f ?txt
f1.txt
f2.txt
f3.txt
Man : read manual page for a and Man txt
name
cat - Concatenate file &
Print Synopsis cat [option]. ... [file]

what is Command : brief description of a command
what is cat cat(1) - Concatenate file & print on the standard output
apropos Keyword - Match commands with keyword in their man pages
appropos cat
nam(7) - pluggable authetication Module for linux
aligned - allow(3) - allocated alligned memory
allocd(3) - allocate Memory
cp source destination - copy a file (or) directory to the same (or) different directory
cp1.txt f2.txt

O/p:

f1.txt	f2.txt
Karpagam	Karpagam
clg	clg
cbe	cbe

rm file - Delete files or directories myf.txt

date - print the current date and time

Tue JUL 2 09:32:29 IST 2019

Display Calender

JUL 2019

Su	Mon	Tue	Wed	Thu	Fri	sat
	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			

Grep pattern file :. searches a file for lines the have a certain pattern cat > f1. txt

Priya studied in USA

grep studied f1. txt

Priya studied in USA

Result:

Thus the linux command has been executed and the output has been verified successfully.

PREPARATION	30	
LAB PERFORMANCE	30	
REPORT	40	
TOTAL	100	
INITIAL OF THE FACULTY		

Ex.No.: 2 **SIMPLE SHELL SCRIPT FOR BASIC ARITHMETIC AND LOGICAL CALCULATIONS**

Date :

Aim:

To write a shell script for basic arithmetic and logical calculations.

Algorithm:

Step1: Start .

Step 2: Read the numbers a,b.

Step 3: Read the option.

Step 4: If the option is 1, addition operation c=`expr$a +$b`.

Step 5: If the option is 2, subtraction operation c= `expr $a + $b`.

Step 6: If the option is 3, multiplication operation c=`expr $a * $b`.

Step 7: If the option is 4, division operation c=`expr$a / $b`.

Step 8: If the option is 5, logical OR operation c= [$a –lt 20 –o $b –gt 100].

Step 9: If the option is 6, logical AND operation c= [$a –lt 20 –a $b –gt 100].

Step 10: If the option is 7, logical NOT operation c= [! false] is true.

Step 11: Print the result.

Step 12: Stop.

Program:

```
echo " enter the values"
read a and b
echo " enter the values"
read choice
$ choice in
1) echo " expr $a + $b' ;;
2) echo ' expr $a - $b' ;;
3) echo ' expr $a \* $b' ;;
4) echo ' expr $a / $b' ;;
5) if [$a - 40] || [$b - gt b] ; then
```

Output:

```
Enter the value
1 2
Enter the choice 1
enter value 2 3
Addition is 5
enter choice 2
enter value 2 3
subtraction 1
Enter the choice 5
True
Enter the choice 6
False
Enter the choice 7
Taile
Enter the choice 8
Invalid number
```

```
    echo " true"
  else
    echo "false"
  fi ;;
7) if [[ ! [ $a -eq $b]];
  then
    echo "a and b are not equal"
  else
    echo "a and b are equal
  *) echo " Invalid number")
  esac
```

Result:

Thus the simple shell script for basic arithmatic and logical exhalation.

PREPARATION	30	
LAB PERFORMANCE	30	
REPORT	40	
TOTAL	100	
INITIAL OF THE FACULTY		

Ex.No. 3 SHELL SCRIPTS TO CHECK VARIOUS ATTRIBUTES OF FILES

Date : AND DIRECTORIES

Aim:

To write a shell scripts to check various attributes of files and directories.

Algorithm:

Step 1: Start .

Step 2: Read the file name.

Step 3: Check the file has write permission or not using [-w filename].

Step 4: Check the file has read permission or not using [-r filename].

Step 5: Check the file has execute permission or not using [-x filename].

Step 6: Checking file is a directory [-d filename].

Step 7: Stop.

Program:

```
echo " Enter the file name'
read f
echo ' checking for write permission
if [-w $f]; then
echo "The file has write permission"
else
    echo " The file has no write permission'
fi ;;
    echo" checking for read permission"
if [-r $f ]; then
    echo ' The file has read permission
```

Ouput:

```
Enter the filename :
  8.sh
checking for write permission
The file has write permission
checking for record permission
The file has read permission
checking for execute permission
The file has execute permission

checking file is a directory
The file is not a directory
```

```
    else
    echo " The file has no read
fi ;;
    echo " checking for execute permission"
if [ -x $f ]; then
    echo " The file has execute permission"
else
    echo " The file has no execute permission"
echo
    echo " checking file is a directory
    if [ -d $ f ]; then
    echo " The file is a directory
    else
        echo " The file is not a directory
    fi;;
    echo.
```

Result:

Thus the various attributes of files and the directions, output was verified successfully.

PREPARATION	30	
LAB PERFORMANCE	30	
REPORT	40	
TOTAL	100	
INITIAL OF THE FACULTY		

Ex.No. : 4 **SHELL SCRIPTS TO PERFORM VARIOUS OPERATIONS ON GIVEN STRING**

Date :

Aim:

To write a shell scripts to perform various operations on given string.

Algorithm:

Step1: Start .

Step 2: Enter the choice.

Step 3: Read two strings s1,s2

Step 4: If option 1, string comparison [$s1=$s2].

Step 5: If option 2, concatenation using two strings $s1|$s2.

Step 6: If option 3, the length of the string t= `echo $s | wc –c `.

Step 7: If option 4 , reversing the string.)

Step 8: Print the result.

Step 9: Stop.

Program:

```
echo " read value" a and b"
read a
read b
echo " enter the choice
read choice
  case $choice in
1) if ["$a" = = $ b"]; then
  echo " string match"
else
  echo" string doesn't match"
```

Output:

```
Read value a and b
2
2
enter the choice :1
String match
Read value a and b
Harini
Tamil
String doesnt match
Read value a and b
Harini
String length.
Enter the choice :4
String reverse :
Enter 1st String:
Jaja
ajgej
```

```
fi ;;
2) c= $a / $b
   echo $c;;
3) f=' echo $a |wc -c'
   echo , $1 ;;
4) echo $a | mv ;;
esac
```

Result:

Thus the shell script the perform the string operation on the string and the output was verified successfully

PREPARATION	30	
LAB PERFORMANCE	30	
REPORT	40	
TOTAL	100	
INITIAL OF THE FACULTY		

Ex.No. :5 **SHELL SCRIPTS TO EXPLORE SYSTEM VARIABLES SUCH AS PATH, HOME ETC**

Date :

Aim:

To write a shell scripts to explore system variables such as PATH, HOME etc.

Algorithm:

Step 1: Start .

Step 2: echo $PATH which displays the path name.

Step 3: echo $HOME to display the user's home directory.

Step 4: echo $USER to display current logged in.

Step 5: echo $PWD to display present working directory.

Step 6: echo $TEMP to display the path to where processes can store temporary files.

Step 7: echo $HOSTNAME to display the system's host name.

Step 8: Stop .

Program:

```
Read choice
  case $ choice in
1) echo $ path
    ;;
2) echo $ home
    ;;
3) echo $ user
    ;;
4) echo $ pwd
    ;;
5) echo $ temp
    ;;
```

Output:

/user/local/sbin:/usr/local/bin:/usr/sbin:/sbin:/bin:/user/games:/usor/local/games

2

/home/students

3

students

4

/home/students/Desktop

5

AT-33

6

/temp/ssh-v2.TK/n.33r/8/agent.2371

```
6) echo $ HOST NAME
;;
esac
```

Result:

Thus the shell script to explore System variables such as path, home has been executed and the output has been verified.

PREPARATION	30	
LAB PERFORMANCE	30	
REPORT	40	
TOTAL	100	
INITIAL OF THE FACULTY		

Ex.No.: 6 SHELL SCRIPT TO CHECK AND LIST ATTRIBUTES OF PROCESSES

Date :

Aim:

To write a shell script to check and list attributes of processes.

Algorithm:

Step 1: Start .

Step 2: ps command is used to display active processes.

Step 3: ps[option] to display the list of attributes of processes.

Step 4: echo $ps –a to display all processes on a terminal, with the exception of group leaders.

Step 5: echo $ps –j to display the process group ID and session ID.

Step 6: echo $ps –c to displays scheduler data.

Step 7: echo $ps –f to display a full listing.

Step 8: echo $ps –plist to display data for the list of process IDs.

Step 9: echo $ps –ulist to display data for the list of usernames.

Step 10: Stop .

Program:

```
read choice
case $choice in
1) ps
   ;;
2) ps-c
   ;;
3) ps-j
   ;;
```

Output:

1

PID	TTY	Time	CMD
3217	pts/0	00:00:00	bash
3267	pts/0	00:00:00	bash
3267	pts/0	00:00:00	bash

2

PID	LLS	PRI	TTY	Time	CMD
3217	TS	19	pts/0	00:00:00	bash
3266	TS	19	pts/0	00:00:00	bash
3267	TS	19	pts/0	00:00:00	bash

3

PID	PGID	SID	TTY	Time	CMD
3217	3417	3217	pts/0	00:00:00	bash
3268	3268	3217	pts/0	00:00:00	bash
3267	3268	3217	pts/0	00:00:00	ps

4

PID	TTY	Time	CMD
3270 /	pts/0	00:00:00	bash
3271 /	pts/0	00:00:00	ps

```
4) ps -a
   ;;
5) ps -f
   ;;
6) ps - p32117
   ;;
7) ps - usstudents
   ;;
esac
```

5

UID	PID	PPID	C	STIME	TTY	TIME CMD
student	3217	3207	0	09:28	pts/0	00:00:00 bash
student	3272	3217	0	09:29	pts/0	00:00:00 bash
student	3273	3272	0	09:29	pts/0	00:00:00 ps

6

PID	TTY	TIME CMD
3217	pts/0	00:00:00 bash

7

PID	TTY	TIME CMD
1435	?	00:00:00 gnome-keyring-d
1438	?	00:00:00 init

Result:

Thus the shell script to check and list attribute of process

PREPARATION	30	
LAB PERFORMANCE	30	
REPORT	40	
TOTAL	100	
INITIAL OF THE FACULTY		

Ex.No.: 7 EXECUTION OF VARIOUS SYSTEM ADMINISTRATIVE COMMANDS

Date:

Aim:

To execute various system administrative commands.

Algorithm:

Step 1: Start .

Step 2: echo $groups gives the group names.

Step 3: echo $chown changes the ownership of a files.

Step 4: echo $usermod changes may be made to password, group membership, expiration date and other attributes of a given user's account.

Step 5:echo $useradd for adding user account to the system and creates a home directory for that particular user.

Step 6: echo $chgrp changes the group ownership of a file or files.

Step 7: Stop.

Commands:

```
echo "chown"
ls -e group.sh
chown unix group.sh
ls -l group.sh
chgrp root group.sh
echo "chmod"
ls -l c group.sh
chmod u+x c group.sh
ls -l group.sh
chmod ugo+x go+x cgroup.sh
ls-cgroup.sh.
```

Output:

Chown

-rw-rw-r--1 students 2 JUL 19.11.32 group.sh

-rw-rw-r--1 students 2 JUL 19:11:32 group.sh

chgrp

-rw-rw-r--1 students 2 JUL 19 11:39 group sh

-rw-r-r--1 students root 8 JUL 19 11:34 group. sh

Chmod

-rw-rw-r--1 students 2 JUL 19 10:30 cgroup.sh

-rw-rw-r--1 students 2 JUL 19 10:30 cgroup.sh

-rw-rw-r--1 students 2 JUL 19 10:30 cgroup.sh

-rw-rw-r--1 students 2 JUL 19 10:30 cgroup.sh

Result:

Thus the execution of various System administrative commands are executed and output is successfully verified.

PREPARATION	30	
LAB PERFORMANCE	30	
REPORT	40	
TOTAL	100	
INITIAL OF THE FACULTY		

Ex.No.: 8 WRITE AWK SCRIPT THAT USES ALL OF ITS FEATURES

Date :

Aim:

To write awk script that uses all of its features.

Algorithm:

Step 1: Start .

Step 2: Create an employee.txt file and enter the datas as employee id, name, designation, salary.

Step 3: Use awk '{print;}' employee.txt.

Step 4: Display the result.

Step 5: Stop .

Program:

```
echo " --- Employee details ---"
cat employee.txt
echo "--- Details ---"
awk " {print $' employee.txt
echo "--- Salary details ---"
awk '{print $1, $2, $3, $4 ; $5 ,$6}'
echo " --- only staffs ---"
awk $5 =" staff '{print $, $2}' employee.
                                txt
```

Output:

--- Employee details ---

1.	Guna	Manager	550	850	15000
2.	Raja	Manager	450	750	10000
3.	Asha	Staff	700	450	9500

Glory details

1.	Guna	Manager	16000
2.	Abi	Manager	11200
3.	Asha	Staff	10650

--- only staffs ---

1.	Asha	Staff	10650

Result:

Thus the program to write awk script that use all of its features is executed and output is verified successfully.

PREPARATION	30	
LAB PERFORMANCE	30	
REPORT	40	
TOTAL	100	
INITIAL OF THE FACULTY		

Ex.No.: 9 **USE SED INSTRUCTION TO PROCESS /ETC/PASSWORDFILE**

Date:

Aim:

To write a program to use sed instruction to process /etc/password file.

Algorithm:

Step 1: Start.

Step 2: $cat /etc/password | sed this cat command dumps the contents of etc/passwprd to sed through the pipe into sed' s pattern space. The patern space is the internal work buffer that sed uses for its operations.

Step 3: Delete all lines with sed using $cat /etc/password | sed 'd'.

Step 4: Prints the line (-p).

Step 5: s / pattern1 / pattern2 is used to substitutes the first occurrence of pattern1 with pattern2.

Step 6: sed addresses: $cat / etc / password | sed 'id| more instructs the sed to perform the editing command on the first line of the file, the sed will delete the first line of / etc / password and print the rest of the line.

Step 7: Stop.

Program:

```
echo " line addressing"
# cat /etc | passwd.
Sed -n -e 1p -e 2p /etc | passwd

Sed -n -e 5p -e 6p -e 7p -e 8p -e 9p
/etc
Passwd
```

Output:

LINE ADDRESSING

root : x : 0 : 0 : root : /root : /bin/bash

daemon : x : 1 : 1 : daemon : /usr/sbin/nologin

sync : x : 4 : 65534 : sync : /bin : /bin/sync

games : x : 5 : 60 : games : usr/ games /usr/sbin/ nologin

man : x : 6 : 12 : man /var/cache/man /usr/sbin/ nologin

CONTENT ADDRESSING

root : x : 0 : 0 : root : /root : /bin/bash

bin : x : 2 : 2 : bin / bin : /usr/sbin/ nologin

syn : x : 3 : 3 : sync / bin : /bin / sync

```
echo " CONTENT ADDRESSING"
sed -n '/bin/p' /etc/passwd
```

Result:

Thus the output of use sed Instruction to process let c /passwd file is processed and verified successfully.

PREPARATION	30	
LAB PERFORMANCE	30	
REPORT	40	
TOTAL	100	
INITIAL OF THE FACULTY		

Ex.No.10 WRITE A SHELL SCRIPT TO DISPLAY LIST OF USERS CURRENTLY LOGGED IN

Date :

Aim:

To write a shell script to display list of users currently logged in.

Algorithm:

Step 1: Start .

Step 2: Enter the choice.

Step 3:Option 1: echo $users to display the user names of users currently logged in.

Step 4: Option 2: echo $who shows currently logged in users with time details.

Step 5: Option 3: echo $whoami: Print the user name associated with the current effective user ID.

Step 6: Option 4: Exit.

Step 7: Stop

Program:

```
read ch
case $ch in
1) echo "your username : $(echo $user)";;
2) echo "currently logged on user";
   who;
3) who ami ;;
4) echo "list user who are logged in"
   who -u;;
```

Output:

1

your username : students

2

currently logged on users

Student :0 2019-07-03 09:00(:0)

student pts/0 2019-07-03 09:23(:0)

3

Students

4

students -kzkT8A

students -kz k7SA @ AT-49

5

list user who commonly used are logged in

09:43:46 up 52min, 2, user, load averag

: 0.07 0.08 0.12

user	TTY	FORM	LOGIN@	IDLE	Icpu	PCPU	WTTAI
Student-U2	:0	:0	08:59	? xam?	5.08	0.025	iii.-0
Student: U8	PTS/6	:0	09:43	2.008	0.438	0.003	co

```
5) echo " list user who are logged in since
file created " last -a ;
esac.
```

Result:

Thus the shell script to display list of users currently logged in is executed and output is verified successfully.

PREPARATION	30	
LAB PERFORMANCE	30	
REPORT	40	
TOTAL	100	
INITIAL OF THE FACULTY		

Ex.No.11 WRITE A SHELL SCRIPT TO DELETE ALL THE TEMPORARY FILES

Date:

Aim:

To write a shell script to delete all the temporary files.

Algorithm:

Step1: Start .

Step2: The command 'find . -name' '*.temp' -print 0.

Step 3: find . -name "temp" | rm for deleting the files.

Step4: Stop.

Program:

```
find - name '*. temp - print 0
find - name "temp" | rm
echo "temporay files are deleted"
```

Output:

Temporary files are deleted

Result:

Thus the shell script to delete all the temporary files are executed and verified Successfully

PREPARATION	30	
LAB PERFORMANCE	30	
REPORT	40	
TOTAL	100	
INITIAL OF THE FACULTY		

Program:

```
echo " Enter the limit"
read n
echo " Enter the numbers"
for (i=0; i<n; i++)
do
read m
a[i]=$m
done
if [${a[$j]} -gt {a[$j+1]}]
then
t=${a[$j]}
a[$j]=${a[$j+1]}
a[$j+1]=$t
fi
done
echo " Enter the element to be
searched":
read s
l=0
c=0
u=$(c$n-1))
while [$l - le $u]
do
```

Output:

```
Enter the limit :4
Enter the number
9
4
7
16
Sorted array is
4
7
9
16
Enter the element to be searched :
4
"Element found at position 1"
```

```
mid= $(((($d+$u))/2))
if [ $s -eq ${a[$mid]} ]
then
   u= $(( $mid -1))
   else
     l= $(( $mid+1))
     fi
     done
     if [ $c -eq 1 ]
   then
     echo "Element found at position
$(( $mid +1))
     else
     echo" element not found"
       fi
```

Result:

Thus the shell script to search an element from an array using binary search and the output is verified successfully.

PREPARATION	30	
LAB PERFORMANCE	30	
REPORT	40	
TOTAL	100	
INITIAL OF THE FACULTY		

Ex.No. 13 **WRITE AWK SCRIPT TO PRINT THE SUM OF N NUMBERS USING LOOP**

Date :

Aim:

To write awk script that print the sum of n numbers using loop.

Algorithm:

Step 1: Start

Step 2: Read the value of n.

Step 3: $awk `{ for(1=1;i<=n;i++)

Step 4: total=total +$i.

Step 5: END {print total}`.

Step 6: Stop the program.

Program:

```
echo -n" Enter number"
read n
sd=0
sum=0
while [$n -gt 0]
do
  sd= $ (( $n-1. 10))
   n = $(( $n/10))
   sum = $ (( $sum + $sd))
done
  echo "sum of all digits is $sum"
```

Output:

Sum of all digits :15

Result:

Thus the awk script to print the sum of n numbers using loop is verified and executed successfully.

PREPARATION	30	
LAB PERFORMANCE	30	
REPORT	40	
TOTAL	100	
INITIAL OF THE FACULTY		