Nanjiah Lingammal Polytechnic College,Mettupalayam -301

Department of Computer Engg

Name :

Sem : III

Year : II

Subject : Linux Practical

LAB EXERCISES

PART – A LINUX COMMANDS	
Write down the syntax and usage of the following exercise with all options. **Check the commands with the system**	
1	(a) Logon to LINUX and logoff. (b) Usage of directory management commands: ls, cd, pwd, mkdir, rmdir (c) Usage of File Management commands: cat, chmod, cp, mv, rm, more, file commands
2	Use the general purpose commnds: wc, od, lp, cal , date, who , tty ,ln
3	Using the simple filters: pr, head, tail, cut, paste, nl, sort
4	Advanced filters : Search for a pattern using grep, egrep & fgrep
5	To know the details of process status- pscommand , Process management commands: &,nohup, kill, nice
6	Communication Commands: news, write, mail, wall, calendar
7	Device pattern using meta character to match each of the following situation:- a. All two character filenames. b. All filenames consisting of two lowercase letters. c. All filenames ending with c. d. All filenames beginning with a c and ending with a digit. e. All filenames beginning with p and having at somewhere.
PART – B SHELL SCRIPTS	
1	Write a shell-script that accepts a numerical value N. Then display the decrementing value of N till it reaches 0.
2	Write a shell-script that takes three command line arguments. The first argument is the name of the destination file and the other two arguments are names of files to be placed in the destination file.
3	Write a Shell script to print contents of file from given line number to next given number of lines
4	a)Shell script to say Good morning/Afternoon/Evening as you log in to system b)Write a shell-script that print out date information in this order: time, day of the week, day number, year – that is like this. 21:18:00 IST Thu 4 Feb 2016
5	Write a shell-script that tells you its name and PID
6	Develop a Basic math Calculator using case statement

7	Write a shell-script that presents a multiple-choice question, gets the user's answer and report back whether the answer is right, wrong or not one of the choices.
8	a) Write script to determine whether given file exist or not, file name is supplied as command line argument, also check for sufficient number of command line argument
	b)Write a shell-script that takes a command line argument and reports on whether it is a directory, a file or something else.

Ex.No :1 Login,Logout ,Directory Management and File Management Commands

Date :

Aim:

To write the syntax and usage of login, logout, Directory Management and File Management commands

HARDWARE REQUIREMENTS:

1. Computer with Pentium processor.

SOFTWARE REQUIREMENTS:

1. Server – Red Hat Linux Enterprise Server release 6.4

PROCEDURE

1.a) Logon to Linux and Logoff

1.Logon to linux

2.Select user and type password

3.Now, Selected user is logged on to linux

4.To Logout from linux ,

Select System → logout option

1.b) Usage of Directory Commands

1. Command : ls

Syntax : ls [options]

Usage : It is used to display files in the current working directory

Options :

-l - list the files in the long format
-a - list all entries ,including the hidden files
-d - list the directory files instead of its contents
-t - lists in the order of last modification time

Eg :

```
# ls -l
# ls -a
# ls -d
# ls -t
```

2. Command : pwd

Syntax : pwd

Usage : It is used to display current working directory

Eg:

```
# pwd
```

3. Command : mkdir

Syntax : mkdir directory name

Usage : It is used to create a new directory

Eg:

```
# mkdir ex1
# ls -l ex1
```

4. Command : cd

Syntax : cd [directory name]

Usage : It is used to change directory from current working directory

Eg:

```
# cd ex1
```

5. Command : rmdir

Syntax : rmdir directory name

Usage : It is used to remove a directory specified in the directory name

Eg:

```
# cd
# rmdir ex1
# ls -l ex1
```

1.c) Usage of File Management Commands

1. Command : cat

Syntax : cat [option] filename

Usage : It is used to display file contents

Options : -s warning about non existing file

Eg:

```
# cat > test
        This is a test file
```

Note :press ctrl+z to save contents

```
# cat test
```

2. Command : chmod options permissions filename

Syntax : chmod [option] mode filename

Usage : It is used to change the file permissions

Options : -f suppress most error messages
-c report only when a change is made

Permissions : read (r)– 4 , write(w) -2 , execute(x) – 1

Eg:

```
# chmod u=rwx,g=rx,o=r test
# ls -l test
# chmod 774 test
# ls -l test
```

3. Command : cp

Syntax : cp source target

Usage : It is used to copy contents from source file to destination file

Eg:

```
# cp test test1
# ls -l test*
```

4. Command : mv

Syntax : for moving, mv[options] source target
for renaming, mv[options] oldname newname

Usage : It is used to move or rename files

Options : -f will move the file(s) without prompting even if it is writing over an

existing target.

-i prompts before overwriting another file.

Eg:

```
# mv test .\Desktop
# ls -l test*
# mv test1 test2
# ls -l test*
```

5. Command : rm

Syntax : rm[options] filename

Usage : It is used to remove files and directories

Options : -r prompts before removing directory or file

-f removes file without prompting

Eg:

```
# rm -r test*
# ls -l test*
```

6. Command : more

Syntax : more [options] [-num *lines*] [*file*...]

Usage : It is used to display long text file one screen at a time

Options : **-d** option display a user message like this **"[Press space to continue, 'q' to quit.]"** .

Eg:

```
# more -12 -d test
```

7. Command : file

Syntax : `file [option(s)] object_name(s)`

Usage : It is used to determine a file's type

Options : `*` - gives information about all files in the current directory

Eg :

```
# file *
# file test.txt
# file .\Desktop\*
```

Result

Thus the above commands has been executed and verified successfully

Ex.No :2 General Purpose Commands

Date :

Aim:

To write the syntax and usage of general purpose commands

HARDWARE REQUIREMENTS:

1. Computer with Pentium processor.

SOFTWARE REQUIREMENTS:

1.Server – Red Hat Linux Enterprise Server release 6.4

PROCEDURE

1. Command : wc

   Syntax : wc [option] file name

   Usage : It is used to count words,bytes and newline

Options :

-l - newline count -w - word count

-c - bytes count -m - character count

Eg :

```
# cat > test
        1 2 3 4 5
        a b c
 #  wc  test
```

2. Command : od

Syntax : od [option] file name

Usage : It is used to dump files in octal and other formats

Options :

-c - Output as ASCII characters or backslash escape sequence

-b - Output as octal bytes

-a - Output as named character

Eg :

```
 # od  -a  test
 # od  -b   test
 #  od  -c  test
```

3. Command : lp

Syntax : lp [option] file name

Usage : It is used to print files

Options :

-d - Print files to the default printer

-i - Specifies an existing job to modify

-o - Sets one or more job options

Eg:

```
#lp  test
#lp  -d  myprinter  –o  media=legal
```

To Print a text file with 12 characters per inch, 8 lines per inch, and a 1 inch left margin,

```
#lp -d myprinter -o cpi=12 -o lpi=8 -o page-left=72 filename
```

4. Command : cal

Syntax : cal [month] [year]

Usage : It is used to display calendar

Options :

-1 Display a single month. This is the default.

-3 Display three months: last month, this month, and next month

-s Display the calendar using Sunday as the first day of the week.

-m Display Monday as the first day of the week.

-j Display dates of the Julian calendar.

-y Display a calendar for the entire current year.

Eg:

```
# cal -y
```

5. Command : Date

Syntax : date [option] [+format]

Usage : It is used to display current date

Options :

-d displays day of the month

-y print the last two digits of the year

H,M,S – Hour,Minute,second respectively

-s sets the system date

Eg:

```
# date

# date –date="yesterday"

# date -d “04/21/2016”
```

6. Command : who

Syntax : who [option] [file] [am i]

Usage : It is used to display who is logged on to the system

Options :

-a displays all information

-H print a line of column headings

Eg:

```
# who

# who -a
```

7. Command : tty

Syntax : tty [option]

Usage : It is used to print the file name of the terminal connected to standard input

Options :

--version – displays this version and exit

--help displays this help and exit

-s returns exit status

Eg:

```
# tty

# tty --version
```

8. Command : ln

Syntax : ln [option] [target] [link name]

Usage : It is used to create either hard link or soft link

Soft link (*also referred to as* **symlink** – short for symbolic link) is a special type of file in Unix, which references another file or directory. Symlink contains the name for another file and contains no actual data.

Hard link is a pointer to a physical data. If you remove the original file, the data will not be lost as long as there's at least one hard link pointing to it. Both original and hard link file has same inode number .

The 1st character in each and every line of the ls command output indicates one the following file types. If the 1st character is l (lower case L), then it is a link file.

- **–**regular file
- **l** link file
- **d** directory
- **p** pipe
- **c** character special device
- **b** block special device

Options :

-n does not overwrite existing files

-s, used to create symbolic links

Eg:

```
# cat  > sample.txt
        this is a soft link
# ln -s sample.txt sample_lnk.txt
# ls -l
# ln  sample.txt  sample_hardlnk.txt
# ls -l
```

Result

Thus the above commands has been executed and verified successfully

Ex.No :3 Simple Filter Commands

Date :

Aim:

To write the syntax and usage of simple filter commands

HARDWARE REQUIREMENTS:

1. Computer with Pentium processor.

SOFTWARE REQUIREMENTS:

1. Server – Red Hat Linux Enterprise Server release 6.4

PROCEDURE

1. Command : pr

   Syntax : pr [option] file name

   Usage : It is used to convert text file for printing

   Options :

   -n prints lines with number
   -h prints heading

   Eg:

```
# cat > test
    This is a sample text file

# pr test
```

```
# pr -n  test
```

2. Command : head

   Syntax : head [option] file name

   Usage : It is used to print the first ‘n’ no of data of the given input

   Options :
   -n prints first ‘n’ no of lines
   -c prints ‘n’ no of bytes from a initial part of a file

   Eg:

```
# cat > test
    1
    2
      3
    4
    5
# head  test

# head -n2  test
```

3. Command : tail

   Syntax : tail [option] file name

   Usage : It is used to print last ‘n’ no of lines from the given input

   Options :
   -n prints last ‘n’ no of lines
   -c prints ‘n’ no of bytes from a initial part of a file

   Eg:

```
# cat > test
    1
    2
    3
    4
    5
# tail  test

# tail  -n2  test
```

4. Command : cut

   Syntax : cut [option] file name

   Usage : It is used to extract portion of a text from a file by selecting column

   Options :
   -c specifies the position of a character
   -d specifies the delimiter

   Eg:

```
# cat > test
    This is a sample file.
    Testing : cut : command

#cut -c2 test

#cut -c1-5 test

  #cut -c2- test
```

```
#cut -d ':' test
```

5. Command : paste

Syntax : paste [option] file list

Usage : It is used to merge the lines from multiple files

Options :

-c specifies the position of a character
-d specifies the delimiter
-s joins all lines in a file
- - merge a file by pasting the data into 2 columns

Eg:

```
# cat > test
    This is a sample file.
    Testing : cut : command

#paste -s test

#paste -d, -s test

#paste  - -  <test

#paste test  /etc/passwd
```

6. Command : nl

Syntax : nl [option] file name

Usage : It is used to number the lines in a file

Options :

-i specifies the increment value instead of printing line number in sequential order
-s specifies the string to be followed by a number

Eg:

```
# nl  test

#nl -i+5 test

#nl –s "nlp" test
```

7. Command : sort

Syntax : sort [option] file name

Usage : It is used to order the elements in a text

Options :

-r reversely arranged
-k sorting based on column or field

Eg:

```
# cat > test
    good    bb
    bad     xx
```

```
        apple   aa
#sort  test
#sort -r  test
#sort -k2  test
```

Result

Thus the above commands has been executed and verified successfully

Ex.No :4 Advanced Filter Commands

grep ,egrep and fgrep Date :

Aim:

To write the syntax and usage of Advanced filter commands

HARDWARE REQUIREMENTS:

1. Computer with Pentium processor.

SOFTWARE REQUIREMENTS:

1.Server – Red Hat Linux Enterprise Server release 6.4

PROCEDURE

grep

grep is an acronym that stands for "Global Regular Expressions Print". grep is a program which scans a specified file or files line by line, returning lines that contain a pattern. A pattern is an expression that specifies a set of strings by interpreting characters as meta-characters.

Syntax : **grep <flags>'<regular expression>'<filename>**

grep prints the search results to the screen (stdout) and returns the following exit values:

0 A match was found.
1 No match was found.
>1 A syntax error was found or a file was inaccessible (even if matches were found).

Some common flags are:

-c for counting the number of successful matches and not printing the actual matches,

-ito make the search case insensitive,

-n to print the line number before each match printout,

-v to take the complement of the regular expression (i.e. return the lines which don't match)

-l to print the file names of files with lines which match the expression

egrep

egrep is an acronym that stands for "Extended Global Regular Expressions Print". The 'E' in egrep means treat the pattern as a regular expression. "Extended Regular Expressions" abbreviated 'ERE' is enabled in egrep. egrep (which is the same as grep -E) treats +, ?, |, (, and) as meta-characters.

In basic regular expressions (with grep), the meta-characters ?, +, {, |, (, and) lose their special meaning. If you want grep to treat these characters as meta-characters, escape them \?, \+, \{, \|, \(, and \).

For example, here grep uses basic regular expressions where the plus is treated literally, any line with a plus in it is returned.

#**grep "+" myfile.txt**

egrep on the other hand treats the "+" as a meta character and returns every line because plus is interpreted as "one or more times".

#egrep "+" myfile.txt

Here every line is returned because the + was treated by egrep as a meta character. normal grep would have searched only for lines with a literal +.

fgrep

fgrep is an acronym that stands for "Fixed-string Global Regular Expressions Print". fgrep (which is the same as grep -F) is fixed or fast grep and behaves as grep but does NOT recognize any regular expression meta-characters as being special. The search will complete faster because it only processes a simple string rather than a complex pattern.

For example, if I wanted to search my .bash_profile for a literal dot (.) then using grep would be difficult because I would have to escape the dot because dot is a meta character that means 'wild-card, any single character':

#**grep "." myfile.txt**

The above command returns every line of myfile.txt. Do this instead:

#fgrep "." myfile.txt

Then only the lines that have a literal '.' in them are returned. fgrep helps us not bother escaping our meta characters.

Eg:

#cat > test

this is a testing page for egrep , egrep and fgrep commands.

flags are n,i,c,v.

#grep “for” test

#grep -iv “for” test

#grep -icv “for” test

cat > test1

aba

aabb

ssaa

wert

rraqa

abc

#egrep b{2} test1

#egrep "aba | abca" test1

grep -n "b$" test1

#fgrep "b" test1

Result

Thus the above commands has been executed and verified successfully

Ex.No :5 Process status and Management Commands

Date :

Aim:

HARDWARE REQUIREMENTS:

1. Computer with Pentium processor.

SOFTWARE REQUIREMENTS:

1. Server – Red Hat Linux Enterprise Server release 6.4

PROCEDURE

1. Command : ps
   Syntax : ps [options]
   Usage : It is used to list running processes
   Options :
   -e list all process
   -f full format list
   Eg

```
#ps

#ps -e

#ps -f
```

2. Command : &
   Syntax : processname &
   Usage : used to run the given process in the background
   Eg

```
# sleep 10 &
# jobs
```

3. Command : kill

Syntax : kill pid no

Usage : Used to kill the process

Eg

```
# ps
# kill -9 3696
```

4. Command : nohup

Syntax : nohup processname &

Usage : allows executing a background process even when the user logs off of the system

Eg

```
# cat > test
        while true ; do echo 'hello' ; sleep 1500; done

# nohup bash test & > nohup.out
```

logout current user and login again

cat nohup.out

5. Command : nice

Syntax : nice -n 'nice value' process name

Usage : used to set process priority

-n add integer N to the niceness (default 0 , range = -20 to 19)

Eg

```
# nice -n 11 sleep 140
  # ps -l
```

Result

Thus the above commands has been executed and verified successfully

Ex.No :6 Communication Commands

Date :

Aim:

HARDWARE REQUIREMENTS:

1. Computer with Pentium processor.

SOFTWARE REQUIREMENTS:

1. Server – Red Hat Linux Enterprise Server release 6.4

PROCEDURE

1. Command : news
   Syntax : **news [options] [news item]**]
   Usage : It is used to Displays message to all the users.
   Options :
   - a displays all of the news items
   -s displays the names of all of the news items.
   -c displays a count of all of the news items

2. Command : write
   Syntax : write <username> <terminal no>
   Usage : It is used to to send message to users logged on to that machine.

   Eg:

   #write root
   Hai
   Press ctrl +d

3. Command : wall
   Syntax : wall [message]
   Usage : It is used to broadcast your message to all login user
   Options : -n hide the sender identification

   Eg:

   #wall
   Hai
   Press ctrl +d

4. Command : mail

Syntax : mail [mail id]

Usage : It is used to read or send messages

Options :-s subject (The subject of the mail)

-c email-address (Mark a copy to this “email-address”, or CC)

-b email-address (Mark a blind carbon copy to this “email-address”, or BCC)

Eg:

To create a new mail,

#mail root

Subject: testing mail command

hai

Press ctrl +d to save mail

To read mail,

#mail -f /var/spool/mail/root

\ type the mail number at the ampersand prompt and press enter.

To quit mail reading, press ctrl +d (or) type q at the ampersand prompt

5. Command : calendar

Syntax : calendar [-ab] [-A *num*] [-B *num*] [-l *num*] [-w *num*] [-f *calendarfile*] [-t [[[*cc*]*yy*]*mm*]*dd*]

Usage : It is used to Displays appointments and reminders.

Result

Thus the above commands has been executed and verified successfully

Ex.No :7 Pattern using Meta Character

Date :

Aim:

To Device pattern using meta character to match each of the following

a. All two character file names
b. All File names consisting of two lowercase letters
c. All file names ending with c
d. All file names beginning with a c and ending with a digit
e. All file names beginning with p and having at somewhere

HARDWARE REQUIREMENTS:

1. Computer with Pentium processor.

SOFTWARE REQUIREMENTS:

1. Server – Red Hat Linux Enterprise Server release 6.4
2. Client -windows Xp

PROCEDURE

a. All two character file names

Command : ls ??

b. All File names consisting of two lowercase letters

Command : ls *[a-z]* | grep '[[:lower:]]'

c. All file names ending with c

Command : ls *c

d. All file names beginning with a c and ending with a digit

Command : ls c*[0-9]

f. All file names beginning with p and having at somewhere

Command : ls p*?p*

Result

Thus the above commands has been executed and verified successfully

Ex.No :8 Display the Decremented Numeric Value of N

Date :

Aim:

To write a shell script that accepts a numerical value N. Then display the decrementing value of N till it reaches 0

HARDWARE REQUIREMENTS:

1. Computer with Pentium processor.

SOFTWARE REQUIREMENTS:

1. Server – Red Hat Linux Enterprise Server release 6.4

PROCEDURE

1. Create a new vi editor file
2. Read integer value of N
3. Check if n > 0 then print value of N
4. Decrement value of N
5. Repeat thru step 3 until n =0

PROGRAM

```
echo " Enter the integer Value "
read n
while [ $n -ge 0 ]
do
echo "$n"
let n--
done
```

Result

Thus the shell program to display the decremented integer value was executed and verified successfully

Ex.No :9 File copying using Command Line Arguments

Date :

Aim:

To write a shell script that takes three command line arguments .the first argument is the name of the destination file and other two arguments are names of files to be placed in the destination file

HARDWARE REQUIREMENTS:

1. Computer with Pentium processor.

SOFTWARE REQUIREMENTS:

1. Server – Red Hat Linux Enterprise Server release 6.4

PROCEDURE

1. Create a new vi editor file
2. Read destination file , first source file and second source file
3. Concatenate two files and copied into destination file
4. Check file status . if status = 0 the print “file copied successfully “ otherwise print “problem in copying file”

PROGRAM

```
echo "Destination file name:$3"
echo "Source File name1: $1"
echo "Source file name2:$2"
cat $1  $2 > $3
status=$?
if [ $status -eq 0 ]
then
echo "Copying Successfully"
else
echo "Problem in copying a file"
fi
```

Result

Thus the shell program File Copying using command line arguments was executed and verified successfully.

Ex.No :10 Printing the File Content

Date :

Aim:

To write a shell script to print contents of file from given line number to next given number of lines

HARDWARE REQUIREMENTS:

1. Computer with Pentium processor.

SOFTWARE REQUIREMENTS:

1. Server – Red Hat Linux Enterprise Server release 6.4

PROCEDURE

1. Create a new vi editor file
2. If no argument is passed then print arguments are missing
3. If target file is missing then print target file does not exists . otherwise print the content from the given line number

PROGRAM

```
if [ $# -eq 0 ]
then
echo "Error : command Line Arguments are Missing"
exit 1
fi
if [ ! -f $3 ]
then
echo " Target File Does not Exists"
exit 2
else
tail -n+$1 $3 | head -n$2
fi
```

Result

Thus the shell program printing the File Content was executed and verified successfully.

Ex.No :11.a Login Message

Date :

Aim:

To write a shell script to say Good morning/Afternoon/Evening as you log in to system

HARDWARE REQUIREMENTS:

1. Computer with Pentium processor.

SOFTWARE REQUIREMENTS:

1. Server – Red Hat Linux Enterprise Server release 6.4

PROCEDURE

1. Create a new vi editor file
2. Set hour to current time hour
3. If hour >=0 and hour <12 then print “Good Morning “
4. If hour >=12 and hour <18 then print “Good Afternoon” .otherwise print “Good Evening”

PROGRAM

```
hour=$(date +"%H")
if [ $hour -ge 0 -a $hour -lt 12 ]
then
echo "Good morning , $USER"
elif [ $hour -ge 12 -a $hour -lt 18 ]
then echo "Good Afternoon ,$USER"
else
echo "Good Evening , $USER"
fi
```

Result

Thus the shell program Login Message was executed and verified successfully.

Ex.No :11.b Printing Date information

Date :

Aim:

To write a shell script to print out date information in the order of time zone, day of the weekday number, year

HARDWARE REQUIREMENTS:

1. Computer with Pentium processor.

SOFTWARE REQUIREMENTS:

1. Server – Red Hat Linux Enterprise Server release 6.4

PROCEDURE

1. Create a new vi editor file
2. Set now to date format %X %Z %a %d %b %y
3. Print now

PROGRAM

```
now=$(date +"%X %Z %a %d %b %Y")
echo $now
```


Result

Thus the shell program Printing Date Information was executed and verified successfully.

Ex.No :12 Printing Process Name and Id

Date :

Aim:

To write a shell script that tell you its name and PID

HARDWARE REQUIREMENTS:

1. Computer with Pentium processor.

SOFTWARE REQUIREMENTS:

1. Server – Red Hat Linux Enterprise Server release 6.4

PROCEDURE

1. Create a new vi editor file
2. Use ps and grep command to print user, pid , comm. Of bash shell process
3.

PROGRAM

```
#!/bin/bash
ps -eo user,pid,comm |grep bash
```

Result

Thus the shell program Printing Process name and ID of a specified process was executed and verified successfully.

Ex.No :13 Simple Arithmetic Calculator

Date :

Aim:

To write a shell script to develop basic calculator using case statement

HARDWARE REQUIREMENTS:

1. Computer with Pentium processor.

SOFTWARE REQUIREMENTS:

1. Server – Red Hat Linux Enterprise Server release 6.4

PROCEDURE

2. Create a new vi editor file
3. If no of argument is less than three then Print "Missing arguments"

4. Set operand1 = First argument $1
5. Set operand 2 =Third Argument $3
6. Set operator op = Second Argument $2
7. If $op = "+" then perform $op1 + $op2
8. If $op = "-" then perform $op1 - $op2
9. If $op = "x" then perform $op1 * $op2
10. If $op = "/" then perform $op1 / $op2
11. If $op = "*" then print Error

PROGRAM

```
op1=$1
op=$2
op2=$3
if [ $# -lt 3 ]
then
   echo "Missing no of Arguments"
exit 1
fi
case "$op" in

 +) echo $(( $op1 + $op2 ));;

 -) echo $(( $op1 - $op2 ));;

 x) echo $(( $op1 * $op2 ));;

 /) echo $(( $op1 / $op2 ));;

 *) echo "Error";;

esac
```

Result

Thus the shell program Simple calculator was executed and verified successfully.

Ex.No :14.a Checking the given File Exist or not

Date :

Aim:

To write a shell script to determine whether given file exist or not , file name is supplied as command line argument, also check for sufficient number of command line argument.

HARDWARE REQUIREMENTS:

1. Computer with Pentium processor.

SOFTWARE REQUIREMENTS:

1. Server – Red Hat Linux Enterprise Server release 6.4

PROCEDURE

1. Create a new vi editor file
2. If no argument is passed then print arguments are missing

3. If target file is found then print target file is found . otherwise print "not found "

PROGRAM

```
if [ $# -ne 1 ]
then
echo "Usage : $0 filename"
exit 1
fi
if [ -f $1 ]
then
echo " $1 is Found "
exit 2
else
echo "Not Found"
fi
```

Result

Thus the shell program Checking the given file exist or not using Command line arguments was executed and verified successfully.

Ex.No :14.b Finding file or directory using Command Line Arguments

Date :

Aim:

To write a shell script that takes a command line argument and report on whether it is a directory, a file or something else

HARDWARE REQUIREMENTS:

1. Computer with Pentium processor.

SOFTWARE REQUIREMENTS:

2. Server – Red Hat Linux Enterprise Server release 6.4

PROCEDURE

12. Create a new vi editor file
13. Print Given Argument
14. If given argument is file the print " Given Argument is a File "
15. If given argument is director, then print "Given Argument is a directory"

16.Otherwise report error message

PROGRAM

```
echo "Your Argument is :$1"
if [ -f $1 ]
then
echo "Given Argument is a File"
elif [ -d $1 ]
then
echo "Given Argument is a Directory"
else
echo "Not Exists"
fi
```

Result

Thus the shell program Finding File or directory using Command line arguments was executed and verified successfully.

Ex.No :15 Multiple - Choice Question

Date :

Aim:

To write a shell script that presents a multiple-choice question , gets the user's answer and report back whether the answer is right , wrong or not one of the choices.

HARDWARE REQUIREMENTS:

1. Computer with Pentium processor.

SOFTWARE REQUIREMENTS:

1 Server – Red Hat Linux Enterprise Server release 6.4

PROCEDURE

1. Create a new vi editor file
2. Print Question and options
3. Get Answer from User

4. Assign user answer in array a1 and Correct answer in array a2
5. Construct for loop to traverse array a1
6. Compare Each array element a1 with a2
7. If matches the print answer is correct. otherwise print answer is wrong

PROGRAM

```
echo "Which Command is used to remove files ? "
echo "A.dm  B.rm   C.delete    D.erase    E.None"
read ans
a1[0]=$ans
a2[0]="B"

echo " Which Command is used to count no.of words,lines,characters containes in a file ? "
echo  "A.count B.countp C.wc D.wcount E.None"
read ans
a1[1]=$ans
a2[1]="C"

echo " Which Command is used to remove directory ? "
echo "A.rdir   B.rmdir   C.rd    D.remove   E.None"
read ans
a1[2]=$ans
a2[2]="B"

echo " Which Command is used to sort the lines of data in a file  ? "
echo "A.sort    B.st    C.sh    D.sort -r    E.None"
read ans
a1[3]=$ans
a2[3]="A"

k=0
q=1
for  i  in  ${a1[@]}
 do
   if  [  $I  =  ${a2[$k]}  ]
   then
   echo "Answer for Question $q  is correct"
   else
   echo "Answer for Question $q  is Wrong"
   fi
  let k++
  let q++
 done
```

Result

Thus the shell program Multiple-choice question was executed and verified successfully.