Usefull Unix Commands

[http://3.bp.blogspot.com/-oojHLNYijZc/VbzqXA1yQtI/AAAAAAAANSE/hUODGCKf4Xw/s1600/Tux.png]

UNIX Assignments

Day1

Concept: Basic commands in UNIX, Filters, Pipes

Objective: At the end of the assignment, participants will be able to:

- Execute Basic Unix commands
- Implement the concepts of Pipes and Filters
- Work with vi editor

Problems:

Section 1:

1. List all the files and sub directories of the directory /bin.

ls ../../bin

2. List all the files including hidden files in your current directory.

ls -a

3. List all the files starting with letter 'r' in your current directory.

Ls r*

4. List all the files having three characters in their names, from your current directory.

ls [a-z][a-z][a-z]

5. List all the files with extension .doc in your current directory.

```
Ls *.doc
```

6. List all the files having the first letter of their name within the range ‘l’ to‘s’, from
 your current directory.

```
ls [l-s]*
```

7. Create a file text1 and read its input from keyboard.

```
cat >text1
rakesh
kumar
suthar
^Z
[3]+  Stopped                 cat > text1
cat text1
rakesh
kumar
suthar
```

8. Copy the contents of file text1 to another file text2.

```
cp text1 text2
```

9. Append the contents of file text2 to file text1.

```
cat text2 >> text1
```

10. Count the number of files in the current directory.

```
ls | wc -l
```

11. Display the output of command ls –l to a file and on the output screen.

```
ls -l >list.lst
cat list.lst
```

12. From file text1 print all lines starting from 10th line.

```
tail --line=+3 myfile


tail --line 1 text1           --shows last 1 line of text1
head –line 1 text1             --show first 1 line of text1
```

13. Find the number of users currently logged on to the system.

```
users|wc -w
```

or

who|wc-l

14. Delete all the files with their names starting with “tmp”.

rm tmp*

Section 2:

1. Display your current working directory.

   pwd

2. Create following directory structure under your Home directory
   (Note: Your home directory is where you login to.)

[https://www.blogger.com /blogger.g?blogID=1726508733578403491]

[https://www.blogger.com /blogger.g?blogID=1726508733578403491]

[https://www.blogger.com /blogger.g?blogID=1726508733578403491]

[https://www.blogger.com /blogger.g?blogID=1726508733578403491]

[https://www.blogger.com /blogger.g?blogID=1726508733578403491]

[https://www.blogger.com /blogger.g?blogID=1726508733578403491]

[https://www.blogger.com /blogger.g?blogID=1726508733578403491]

[https://www.blogger.com /blogger.g?blogID=1726508733578403491]

[https://www.blogger.com /blogger.g?blogID=1726508733578403491]

```
SYSTEM (SUB DIRECTORY)
  HARDWARE
              INPUT
    OUTPUT
  SOFTWARE
    APPL

    SYS
```

```
mkdir -p system/hardware/input
mkdir -p system/hardware/output
mkdir -p system/software/appl
mkdir -p system/software/sys

          ls -R system/

system/:
hardware  software

system/hardware:
input  output

system/hardware/input:
mouse  scanner

system/hardware/output:
printer  vdu

system/software:
appl  sys

system/software/appl:
lotus  wordstar

system/software/sys:
linkers  os
```

3. List detailed information about all the files and directories of Hardware directory while your current directory is still the home directory.

ls -R system/hardware/

```
system/hardware/:
input  output

system/hardware/input:
mouse  scanner

system/hardware/output:
printer  vdu
```

4. Change your current directory to SYS and list the names of all files and subdirectories in the directory sub tree starting that starts from your home directory.

ls -R ../../dac227

5. Copy the file SCANNER to directory SYSTEM while your current directory is APPL.

```
cp hardware/input/scanner
scanner
```

6. Rename the file SCANNER to SCAN.

```
mv scanner scan
```

7. Read some text form Keyboard and append it to the file SCAN.

```
cat scan
```

8. Remove the directory sub tree starting from SYSTEM in one go.
   (Note: This command is potentially dangerous)

```
rm -r system
```

Section 3:

1. Count the total number of words in file text1.

```
cat text1|wc -w
```

2. List the contents of ls command page wise.

3. Create a file FILE2 with some text in it. Increase the no. of hard links to the file FILE2 to 3 and check the inode number and link count for those names.

```
ln file2 file1
ln file2 file2

ls -i
2472656 file1  2472656 file2  2472656 file3
```

4. Using one single command, display the output of “who” and “pwd” commands.

```
pwd;who
```

5. Display the system date in following format:

   Today is Friday, 17 May 96

```
printf "today is %s, %s \n" $(date +%A)
$(date +%Y)

date +"Today is "%A," "%B" "%y
```

echo -n "Today is ";date +%A," "%B" "%y

echo -n 'Today is ';date +%A,;date +%B;date +%y

6. Display the following text message on the monitor screen.

   Deposited $100 to you account

echo 'Deposited $100 to you account'

7. Display the following message on the monitor.

   The long listing of my home dir ………… is …………

   (Hint: Use ls – l and pwd commands)

8. Use **find** command to locate the following within your home directory tree:

   a) Files with extension .c or .pl

   find assignment1 *.c
   find assignment1 *.pl

   b) Directories having permission 755

   find . -perm -755

   c) Files having permission 655

   find . -perm -655

   d) Files having inode number 12122

   find . -inum -12122

   e) Files which have not been accessed for more than a year and save the list in Old_File

   man find . -atime

   f) Files whose size is greater than 1024 bytes

   find . -size 1k --k form 1024 c for 1 byte b for 512 blocks

Section 4 :

1. Using vi editor:

   a) Create a file "Data1.txt

   touch Data1.txt

   b) Save the file and exit from the vi editor.

:wq

c) Open the vi editor without specifying a file name

vi

d) Write some text and and save it to a file “MyData2.txt”

:wq MyData2.txt

e) Repeat point (c) but after writing some text don’t save and just exit “vi”

:q!

2. Create a file using vi editor and enter the following text in it:

Unix Unix Unix Unix Unix

Unix is multi user operating system, Unix is multi tasking o\perating system
Everything on Unix is a file.
Unix File structure is hierarchical like an upside down tree.

Regular files cannot contain another file, or directory
Directory File Contains directory(s) and/or file(s) within it
Device files are used to represent physical devices.
Symbolic link is an indirect pointer to a file

a) Save the file without exiting vi.

:w

b) Display the line number from within vi

:set nu

:set nonu

c) Move first three lines of the file to the end of the file.

Go to the top most of file

Press esc key

Press 3yy

Go to end

Press p

d) Copy 5th line and paste above the first line.

Go to fifth line by :5

Press yy

Go to 1st line

Press p

e) Search the word ***Unix*** in forward direction

/Unix

f) Search the word ***Unix*** in backward direction

?Unix

g) Replace all the occurrences of the word ***Unix*** with ***UnixOS***

%s/Unix/UnixOS

Day 2

Concept: Shell Scripting, filters

Objective: At the end of the assignment, participants will be able to understand and implement:

- Regular expressions and grep
- Shell fundamentals
- Using basic UNIX commands and filters as building blocks
- The commands as conditions for decision making in shell scripts
- Shell Scripting constructs

Problems:

1. List only the directories in your current directory

```
ls -p|grep '/'
ls -p|grep '/'|wc –l
ls -d */
```

2. Display the name and count of the directories in the current directory.

```
ls -d */ |tee output|wc -l >> output

ls -d */;ls -d */|wc -l
```

3. Find out whether the users with a pattern“itp9” in their names have logged in ?

```
echo total users are `who | grep 'itp9'`
```

4. Find out whether a particular user “itp9” has logged in ?

```
who|cut -d' ' -f1|grep -F 'itp9'
```

5. Assign a value “Black” to var1 and then display the following message on the terminal using this variable.

var1="Black"

echo $var1 belt is associated with karate

Black belt is associated with karate

6. Create the file employee.txt having colon (:) separated fields. The fields of the record are: enumber, ename, eposition, esal, edoj, edept.
   And now answer the following:

```
enumber :      ename   :eposition      :esal   :       edoj
:edept
1000    :      hemant  :manager        :45000  :       01-10-1990
:it
2000    :      rakesh  :manager        :45000  :       02-10-1990      :it
```

   a. List all the employees along with a row number

   ```
   cut -d':' -f1,3 emp.lst
   cat -n emp.lst
   ```

   b. Sort the file as per the names

   ```
   cut -d':' -f2 emp.lst
   sort -t':' -k 2 emp.lst
   ```

   c. List top three salaried employees

   ```
   cut -d':' -f4 emp.lst|sort|head -n 3
   ```

   d. Remove duplicate records from the file

   ```
   cat emp.lst|uniq –u
   cut -d':' -f4 emp.lst|sort -u
   ```

   e. List dept. no along with no. of employees working in each dept.

   ```
   cut -d':' -f1,2 emp.lst|sort -u
   ```

   f. Sort the file in descending order of salary

   ```
   sort -t':' -k 4,4 emp.lst
   cut -d':' -f4 emp.lst|sort -r
   ```

   use –r for reverse order and –t for delimiter –k 4,4 means start from 4th column to 4th column only so sorted only based on 4th column .

7. Accept a file name and a number (x). Display x lines from the top of the file.
   Check if the file exists and is readable. The value of x should not

exceed the total number of lines in the files. Display suitable messages in case an error is encountered.

```
read -p "enter file name to create" fname
clear
if [ -f $fname ]
then
        echo "file found"
        if [ `ls -l $fname|cut -c2`==r ]
        then
               echo "file is readable your file is
\n"
            head $fname -n 2
    fi
fi
```

8. Write a menu based script which displays the following options :

    1. Make a file.
    2. Display contents
    3. Copy the file
    4. Rename the file
    5. Delete the file
    6. Exit

    Enter your option:

    If the user selects option 1, accept a file name from the user. If the file exists, then display an error message pass the control to the menu. If the file does not exist, then allow the user to enter some data. Pressing ^D would save the contents and display the menu.

    If the user selects option 2, then accept a file name from the user. If the file exists, then display the contents of the file. If the file does not exist, then display suitable error message. After this process, display the menu to accept another option.

    Selecting Option 3 allows the user to accept the source file and target file. If the source file exists and is readable, then accept the target file name. If the source file does not exist, then display suitable error message. If the target file does not exist, then copy the contents of the source file to the target file. If the target file exists, then display suitable message and go back to the menu.

Option 4 is similar to option 3 but rename the file instead of copying.

Selecting option 5 allows the user to enter a file name. If the file exists, then check to see if it is writable. If so, then delete the file with confirmation from the user. If the file does not exist, then display suitable error message.

```
 while [ 1 ]
 do
 echo "press 1 for make a file"
 echo "press 2 for display content"
 echo "press 3 for copy a file"
 echo "press 4 form rename a file"
 echo "press 5 for delete a file"
 echo "press 6 for exit"
read -p "choicd: " ch
case $ch in
        1)read -p "file name: " fname
            vi $fname ;;
        2)read -p "file name: " fname
            if [ -f $fname ]
                then
                cat $fname
            else
                echo "file not exist"
             fi;;

        3)read -p "source name: " sname
             if [ ! -f $sname ]
             then
                 echo "file not exist"
             else
                 read -p "target name: " tname
                 cp $sname $tname
             fi;;
        4)read -p "source name: " sname
           if [ -f $sname ]
         then
            read -p "save as: " tname
            mv $sname $tname
         else
              echo "file not exist"
              fi;;

        5)read -p "file to delete" fname
            if [ -f $fname ]
            then
            rm $fname
            else
            echo "no file  exist for deletion"
          fi;;
```

```
    6)break;;
esac
done
```

9. Write a menu based script which displays the following options :
   Accept roll number, name, marks in sub1, sub2 and sub3. Calculate total, percentage, grade & class and enter the record in a file "student". The marks are out of 100 in each subject. Allow the user to enter any number of records.

   Grade is found out as follows :

Percentage	Grade
< 35	Fail
>= 35 & < 50	Third
>= 50 & < 60	Second
>= 60 & < 75	First
>= 75	Distn

   If the student secures < 35 in any one of the subjects, then class = "fail". Otherwise class="pass".

10. Accept roll number from the user at the command line. Search it in the "student" file. If it is present, then display name, percentage, grade and class of the student. If the roll number is not present, then display a message "not found". Allow the user to enter any number of queries.
11. Accept the roll number from the user. Search it in the file. If it is present, then delete the respective record from the "student" file. Get the confirmation from the user before deleting the record from the file. If the roll number is not present, then display suitable error message.
12. Write a menu based shell script which will perform arithmetic operations on two numbers which are inputted by user. Menu should display following operations

```
   Menu
 --------------
1: Addition
2: Substraction
3: Multiplication
4: Division
5: Exit
```

```
while [ 1 ]
do
```

```
        clear
        read -p "ENTER FIRST NO." fno
        read -p "ENTER FIRST NO." sno
        echo "Menu
                --------------
                1: Addition
                2: Substraction
                3: Multiplication
                4: Division
                5: Exit"
        read ch
        case $ch in
                1)let res=$fno+$sno
                        echo "addition is: " $res;;
                2)let res=$fno-$sno
                        echo "Substraction is: "
$res;;
                3)let res=$fno*$sno
                        echo "Multiplication is: "
$res;;
                4)let res=$fno/$sno
                        echo "Division is: " $res;;
                5)
                        exit ;;
                *)echo "unpridicatable
output";;
        esac
        if [ $dead -eq 0 ]
        then
                break;
        fi
        read x
read x
done
```

13. Write a shell script that will remove a file taken as command line argument after taking the proper backup of file in /home/user1

/backup directory

```
if [ $# -eq 1 ]
then
        mkdir -p home/user1/backup
        mv $1 home/user1/backup
else
        echo "please give one file name to
take bkp and delete"
fi
```

14. Write a shell script that will accept a string from the user. Copy all contents of the file to other file without that string. Also display number of characters, lines, and words.

```
echo "ENTER STRING TO BE LEFT FROM
FILE"
read str
echo "ENTER FILE NAME"
read fname
cat $fname|tr -d $str
```

15. Write a shell script which will generate the O/P as follows

```
        echo"
           *
 * * *  * * * * * *
           *
               "
```

16. Write a shell script which will accept three nos. from the keyboard and finds the largest of them.

```
read -p "enter first no " fn
read -p "enter second no " sn
read -p "enter third no " tn
if [ $fn -gt $sn ]
then
        if [ $fn -gt $tn ]
        then
                echo $fn "is greatest"
        else
```

```
                echo $tn "is greatest"
        fi
else
         if [ $sn -gt $tn ]
         then
                 echo $sn "is greatest"
         else
                 echo $tn "is greatest"
         fi
fi
```

17. Write a shell script which will accept three nos. from the keyboard and finds the smallest of them.

```
read -p "enter first no " fn
read -p "enter second no " sn
read -p "enter third no " tn
if [ $fn -lt $sn ]
then
        if [ $fn -lt $tn ]
        then
                echo $fn "is smallest"
        else
                echo $tn "is smallest"
        fi
else
         if [ $sn -lt $tn ]
         then
                 echo $sn "is smallest"
         else
                 echo $tn "is smallest"
         fi
fi
```

18. Write a shell script which will calculate the factorial of an integer entered from the keyboard.

```
read -p "enter the no. to find factorial" no
fact=1;
for((i=no;i>0;i--))
do
```

```
        let fact=fact*i
done
echo $fact
```

19. Count total number of files in the current directory containing the text "UNIX" in them.
    (Note: Prepare a few files with some text as well as the text "UNIX" in them to test it)

```
touch Unix
touch UNIX.LST
touch MYUNIX.LST
ls -a|grep -c "UNIX"
```

20. Write a shell script to copy one file to another line by line.
    (Hint: May use head/tail filters)

```
echo "enter file name to be coppied"
read fname
echo "enter target filename"
read tname
if [ ! -f $fname ]
then
        echo "file not found"
else
        echo "enter how many lines to be copied"
        read nol
        head --line=$nol $fname|tee $tname
fi
```

21. Write a shell script that displays the name of all ordinary files in the current directory having execute permission for the owner.

```
echo "total ordinary files are" `ls -l|grep -c "^-"`        for regular files

echo "total ordinary files are" `ls -l|grep -c "^l"`        for softlink files

echo "total ordinary files are" `ls -l|grep -c "^d"`        for directory files

echo "total ordinary files are" `ls -l|grep -c
```

"^b"` for blocak special files

echo "total ordinary files are" `ls -l|grep -c "^c"` for char special files

22. Write a shell script which allows a menu based selection of whether a user wants to:
    a. List the contents of his HOME directory
    b. List the contents of another user's HOME directory
    c. Display only the login name along with terminal addresses of all the users currently logged on to the system.
    d. Exit the script.

```
 echo "
            1.    List the contents of his HOME
directory
             2.     List the contents of another
user.s  HOME directory
             3.    Display only the login name
along      with      terminal      addresses      of
                        all the users currently
logged on to the system.
          4.    Exit the script.
          "
     read ch
     case $ch in
          1)ls .;;
          2)ls ~user.s;;
          3)who| cut -c 1-22;;
          4)exit 0;;
     esac
```

To run
sh filename.sh

Grep ending with
ls -p|grep 'cpp$'

grep starting with

ls -p|grep '^a'

tee
ls |tee temp
It store result in both stdout and temp file

Unix command to display java settings on machine

java -XshowSettings

Posted 27th January 2015 by rksuthar

0 Add a comment