School of Engineering and Applied Science
Ahmedabad University
Operating Systems Lab
Submission of Lab Assignment – 01

Roll no: 201501021
Charmi Chokshi

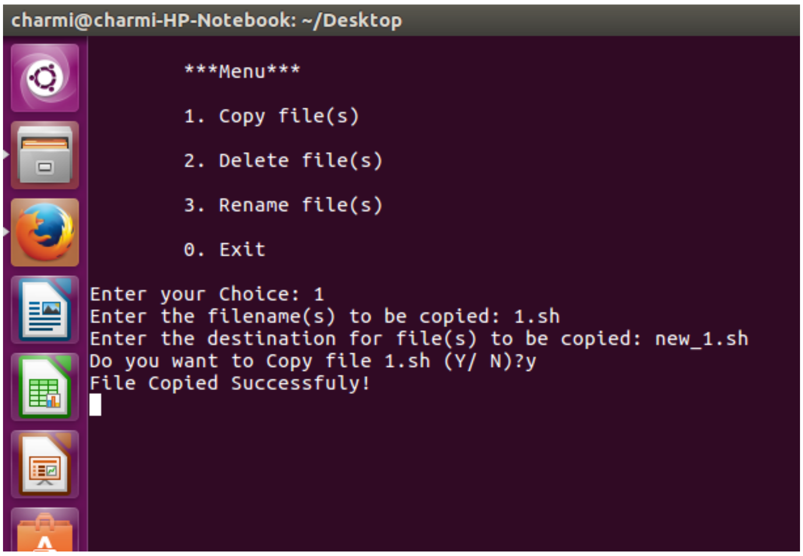
September 01, 2017

Q.1: Write a script to obtain the effect DELETE/CONFIRM command. Generalize it to be used for COPY/CONFIRM and RENAME/CONFIRM.

Code:

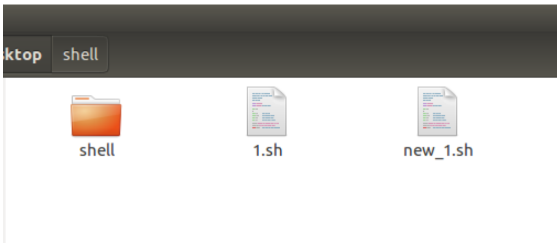
```
# To copy files with confirmation
fcopy()
{
	echo -n 'Enter the filename(s) to be copied: '
	read filename
	echo -n 'Enter the destination for file(s) to be copied: '
	read dest

	echo -n  'Do you want to Copy file' $filename '(Y/ N)?'
	read c

	if [[ ("$c" = "Y") || ("$c" = "y") ]];
	then
		cp -i $filename $dest
		echo 'File Copied Successfuly!'
		sleep 2
	else
		echo 'File NOT Copied.'
		sleep 2
	fi
}

# To delete files with confirmation
fdel()
{
	echo -n  'Enter the filename(s) to be deleted: '
	read filename
	rm -i $filename
	echo 'File Deleted successfuly!'
	sleep 2
}

# To rename files with confirmation
fren()
{
	echo -n  "Enter the filename(s) to be renamed: "
	read filename
	echo -n  'Enter new Name: '
	read dest

	echo -n  'Do you want to Rename file' $filename '(Y/ N)?'
	read c

	if [[ ("$c" = "Y") || ("$c" = "y") ]];
	then
		mv -i $filename $dest
		echo 'File Renamed Successfuly!'
```

```
			sleep 2
		else
			echo 'File NOT Renamed.'
			sleep 2
		fi
}

while true
do
clear
cat << MENU

		***Menu***

		1. Copy file(s)

		2. Delete file(s)

		3. Rename file(s)

		0. Exit

MENU

echo -n  'Enter your Choice: '
read choice

case $choice in

		1)	fcopy
	continue;;

		2)   fdel
			continue;;

		3)   fren
	continue;;

		0)   echo 'Thank You!'
			exit;;

		*) echo 'Please enter proper Choice.'

esac
done
```

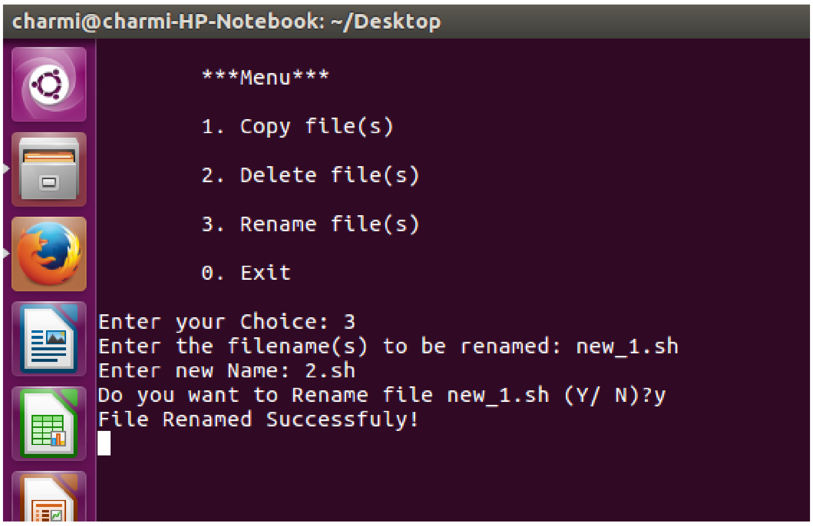


```
        ***Menu***

        1. Copy file(s)

        2. Delete file(s)

        3. Rename file(s)

        0. Exit

Enter your Choice: 1
Enter the filename(s) to be copied: 1.sh
Enter the destination for file(s) to be copied: new_1.sh
Do you want to Copy file 1.sh (Y/ N)?y
File Copied Successfuly!
```

Coping file 1.sh

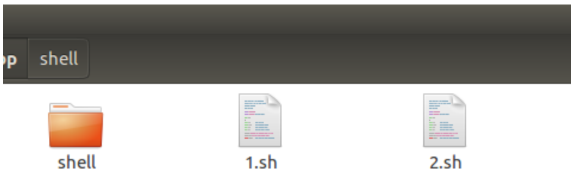


Copied file new_1.sh

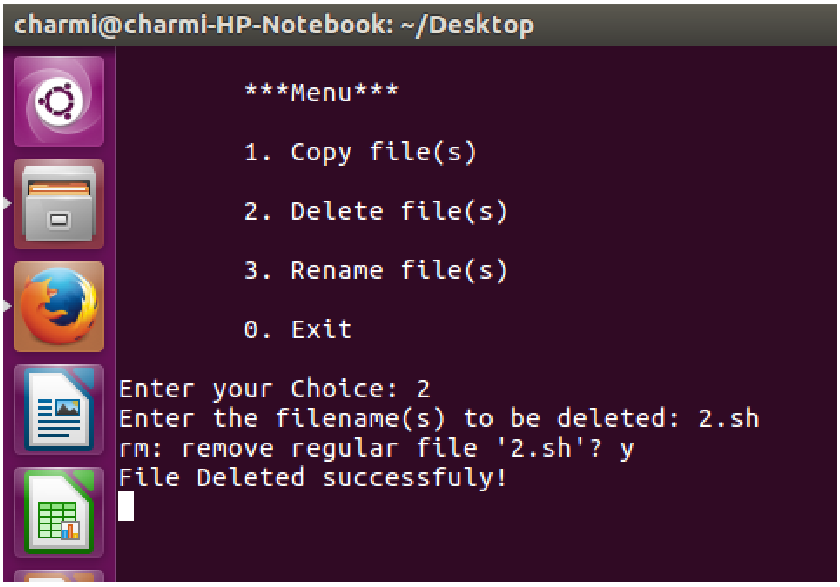


```
        ***Menu***

        1. Copy file(s)

        2. Delete file(s)

        3. Rename file(s)

        0. Exit

Enter your Choice: 3
Enter the filename(s) to be renamed: new_1.sh
Enter new Name: 2.sh
Do you want to Rename file new_1.sh (Y/ N)?y
File Renamed Successfuly!
```

Rename file new_1.sh to 2.sh

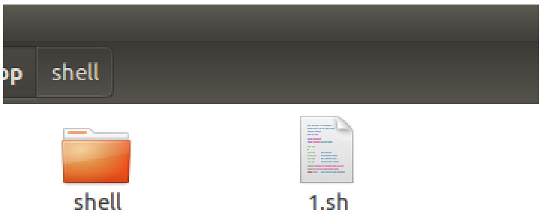


Renamed file



```
        ***Menu***

        1. Copy file(s)

        2. Delete file(s)

        3. Rename file(s)

        0. Exit

Enter your Choice: 2
Enter the filename(s) to be deleted: 2.sh
rm: remove regular file '2.sh'? y
File Deleted successfuly!
```

Deleting file 2.sh

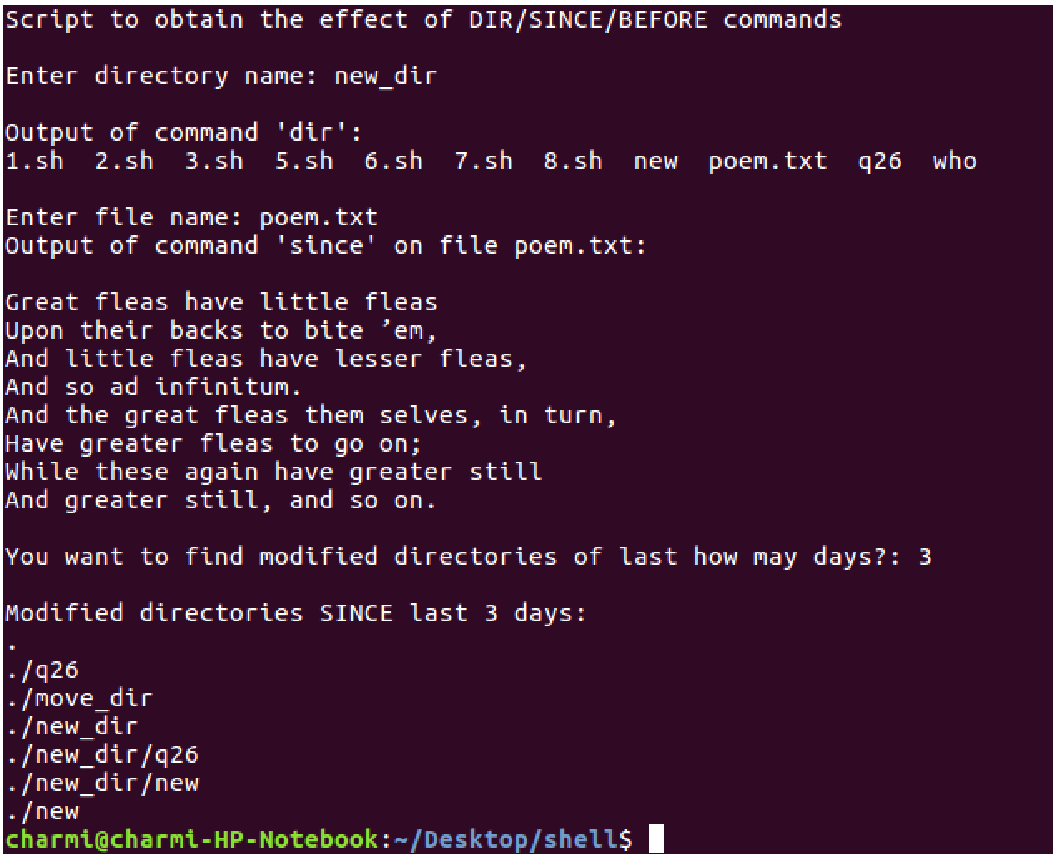


File 2.sh is deleted

Q.2: Write a script to obtain the effect of DIR/SINCE/BEFORE command.

Code:

```
clear
echo Script to obtain the effect of DIR/SINCE/BEFORE commands
echo

echo -n "Enter directory name: "
read d
echo
echo "Output of command 'dir': "
dir $d

echo
echo -n "Enter file name: "
read f
echo "Output of command 'since' on file $f: "
echo
since $f

echo
echo -n "You want to find modified directories of last how may days?: "
read days
echo
echo "Modified directories SINCE last $days days: "
find -type d -mtime -$days
```

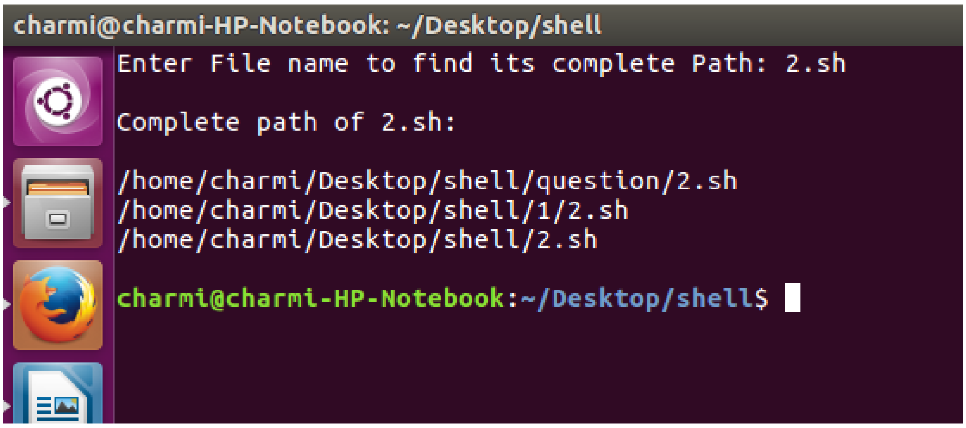


```
Script to obtain the effect of DIR/SINCE/BEFORE commands

Enter directory name: new_dir

Output of command 'dir':
1.sh  2.sh  3.sh  5.sh  6.sh  7.sh  8.sh  new  poem.txt  q26  who

Enter file name: poem.txt
Output of command 'since' on file poem.txt:

Great fleas have little fleas
Upon their backs to bite ’em,
And little fleas have lesser fleas,
And so ad infinitum.
And the great fleas them selves, in turn,
Have greater fleas to go on;
While these again have greater still
And greater still, and so on.

You want to find modified directories of last how may days?: 3

Modified directories SINCE last 3 days:
.
./q26
./move_dir
./new_dir
./new_dir/q26
./new_dir/new
./new
charmi@charmi-HP-Notebook:~/Desktop/shell$ 
```

Q.3: Input a file name from a user and find out the complete path for a give file name.

Code:

```
clear

echo -n 'Enter File name to find its complete Path: '
read filename
if [ -f $filename ]
then
echo "Complete path of "$filename":"
find $PWD -type f | grep "$filename"

else
echo -n 'File '$filename' NOT exist in: '
pwd
fi
```



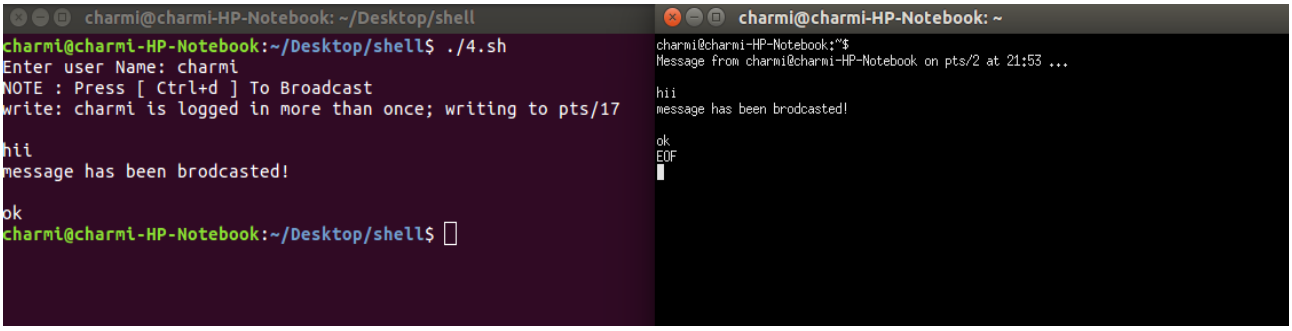
```
Enter File name to find its complete Path: 2.sh

Complete path of 2.sh:

/home/charmi/Desktop/shell/question/2.sh
/home/charmi/Desktop/shell/1/2.sh
/home/charmi/Desktop/shell/2.sh

charmi@charmi-HP-Notebook:~/Desktop/shell$
```

Complete path of file 2.sh

Q.4: Write a script to broadcast a message to a specified user or a group of users logged on any terminal.

Code:

```
echo -n "Enter user Name: "
read usrName
echo "NOTE : Press [ Ctrl+d ] To Broadcast"
write $usrName
```



charmi@charmi-HP-Notebook: ~/Desktop/shell

```
charmi@charmi-HP-Notebook:~/Desktop/shell$ ./4.sh
Enter user Name: charmi
NOTE : Press [ Ctrl+d ] To Broadcast
write: charmi is logged in more than once; writing to pts/17

hii
message has been brodcasted!

ok
charmi@charmi-HP-Notebook:~/Desktop/shell$
```

charmi@charmi-HP-Notebook: ~

```
charmi@charmi-HP-Notebook:~$
Message from charmi@charmi-HP-Notebook on pts/2 at 21:53 ...

hii
message has been brodcasted!

ok
EOF
```

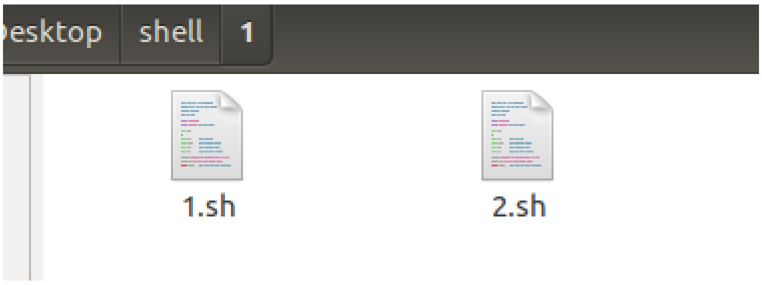
Brodcasting different messages from one user to another

Q.5: Write a script to copy the files from two directories onto a new directory in such a way that only the latest file is copied, in case there are common files in both the directories.

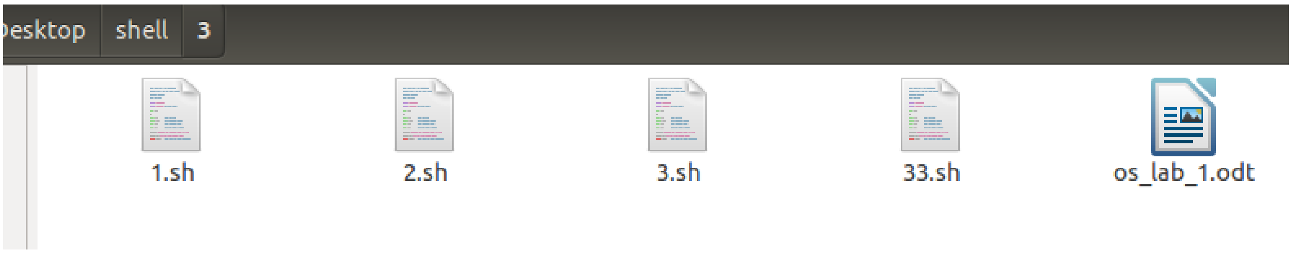
Code:

```
clear

echo -n 'Enter 1st Directory: '
read dir1

if [ ! -d $dir1 ]; then
    echo "Directory 1 not exist"
    exit 1
fi

echo -n 'Enter 2nd Directory: '
read dir2

if [ ! -d $dir2 ]; then
    echo "Directory 1 not exist"
    exit 1
fi

# copy all file names in to a text file
ls $dir1 > dir1.txt
ls $dir2 > dir2.txt

echo -n 'Enter the destination Directory for file(s) to be copied: '
read TargetDir

mkdir $TargetDir

for FileFromDir1 in `cat dir1.txt`
do
    flag=0
    for FileFromDir2 in `cat dir2.txt`
    do
        # compare file name is same or not
        if [ "$FileFromDir1" = "$FileFromDir2" ]
        then
            # check access time of both files and copy the latest file to target directory
            if [ "$FileFromDir1" -nt "$FileFromDir2" ];
            then
                cp $dir1/$FileFromDir1 $TargetDir
            else
                cp $dir2/$FileFromDir1 $TargetDir
        fi
        # file name from directory1 is match to file name of directory2 [ flage is true ]
        flag=1
        fi
    done
```

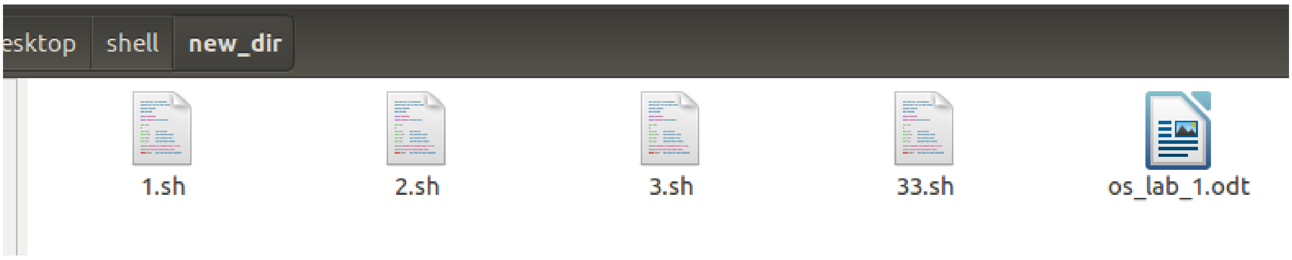
```
	#if file name from directory1 is does NOT match to file name of directory2 [ copy to TargetDir ]
	if [ $flag -eq 0 ]
	then
		cp $dir1/$FileFromDir1 $TargetDir
	fi
done

# copy rest of file from directory2 to TargetDir
for FileFromDir2 in `cat dir2.txt`
do
	flag=0
	for FileFromDir1 in `cat dir1.txt`
	do
		if [ "$FileFromDir2" = "$FileFromDir1" ]
		then
			flag=1
		fi
	done
	if [ $flag -eq 0 ]
	then
		cp $dir2/$FileFromDir2 $TargetDir
	fi
done

echo 'File(s) Copied Successfully!'
```



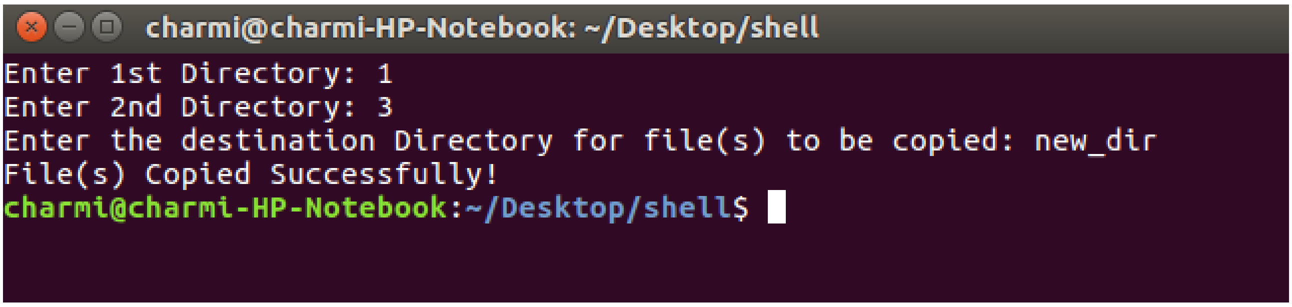
contains of directory 1



contains of directory 3



newly created directory



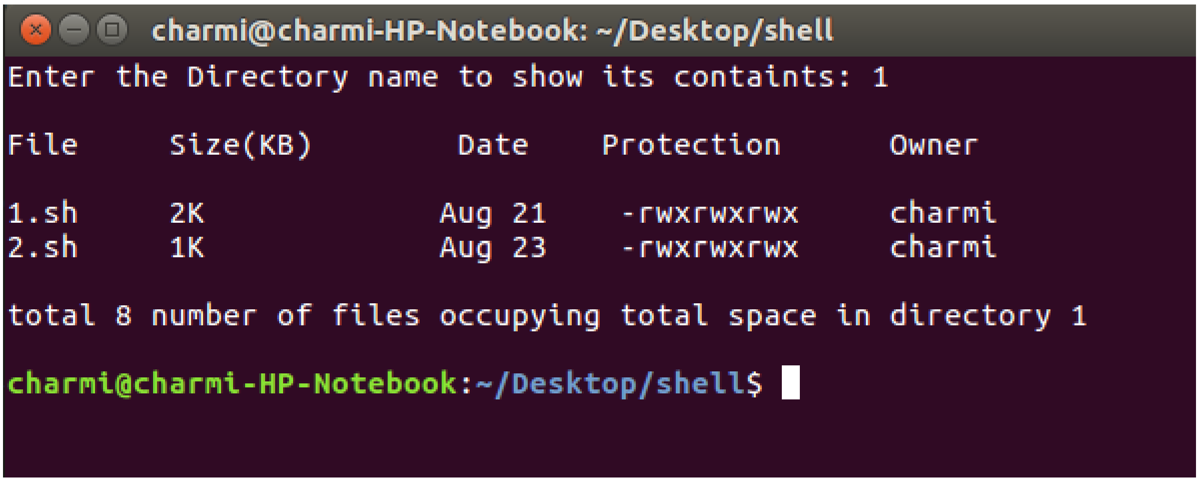
```
charmi@charmi-HP-Notebook: ~/Desktop/shell
Enter 1st Directory: 1
Enter 2nd Directory: 3
Enter the destination Directory for file(s) to be copied: new_dir
File(s) Copied Successfully!
charmi@charmi-HP-Notebook:~/Desktop/shell$
```

output of the script

Q.6: Write a script to display the files in the specified directory in the following format:
File Size in KB Date Protection Owner
At the end display total number of files occupying total space.

Code:

```
clear
echo -n 'Enter the Directory name to show its containts: '
read dir

echo
echo -e 'File \t Size(KB) \t Date \t Protection \t Owner'
echo

ls -l --block-size=K $dir | tail -n +2 | awk '{print $9, "   ", $5, "            ", $6,$7, "  ", $1 ,"   ", $3}'
echo

echo -n `ls -l $dir | head -1`
echo ' number of files occupying total space in directory '$dir
echo
```



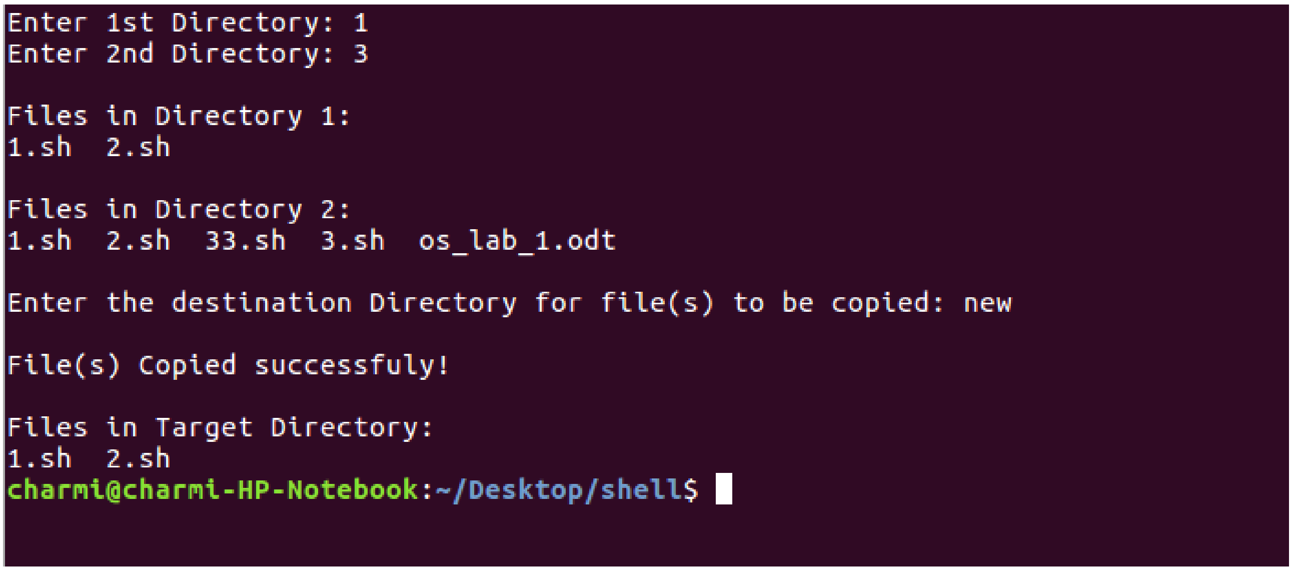
```
charmi@charmi-HP-Notebook: ~/Desktop/shell
Enter the Directory name to show its containts: 1

File      Size(KB)         Date     Protection      Owner

1.sh      2K              Aug 21     -rwxrwxrwx      charmi
2.sh      1K              Aug 23     -rwxrwxrwx      charmi

total 8 number of files occupying total space in directory 1

charmi@charmi-HP-Notebook:~/Desktop/shell$
```

Q.7: Write a script to compare identically named files in two directories and if they are same, copy one of them in a third directory.

Code:

```
clear
echo -n 'Enter 1st Directory: '
read dir1

if [ ! -d $dir1 ]; then
        echo "Directory 1 not exist"
        exit 1
fi

echo -n 'Enter 2nd Directory: '
read dir2

if [ ! -d $dir2 ]; then
        echo "Directory 1 not exist"
        exit 1
fi

# copy all file names in to a text file
ls $dir1 > dir1.txt
ls $dir2 > dir2.txt

echo
echo 'Files in Directory 1: '
ls $dir1
echo
echo 'Files in Directory 2: '
ls $dir2

echo
echo -n 'Enter the destination Directory for file(s) to be copied: '
read TargetDir

mkdir $TargetDir

for FileFromDir1 in `cat dir1.txt`
do
        flag=0
        for FileFromDir2 in `cat dir2.txt`
        do
                # compare file name is same or not
                if [ "$FileFromDir1" = "$FileFromDir2" ]
                then
                        cp -i $dir1/$FileFromDir1 $TargetDir
                fi
        done
done
```

```
echo
echo 'File(s) Copied successfuly!'
echo
echo 'Files in Target Directory: '
ls $TargetDir
```



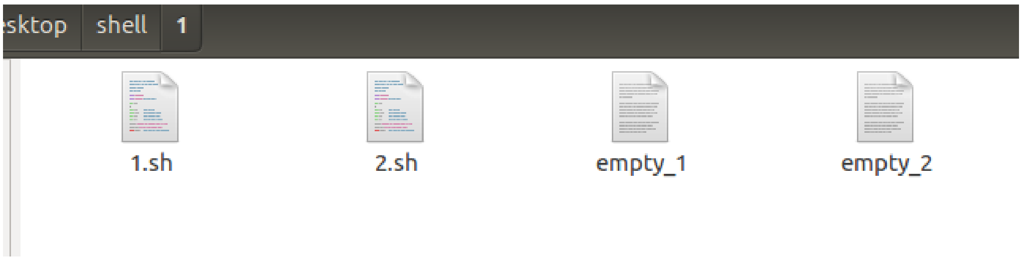
```
Enter 1st Directory: 1
Enter 2nd Directory: 3

Files in Directory 1:
1.sh  2.sh

Files in Directory 2:
1.sh  2.sh  33.sh  3.sh  os_lab_1.odt

Enter the destination Directory for file(s) to be copied: new

File(s) Copied successfuly!

Files in Target Directory:
1.sh  2.sh
charmi@charmi-HP-Notebook:~/Desktop/shell$ 
```

output of the script

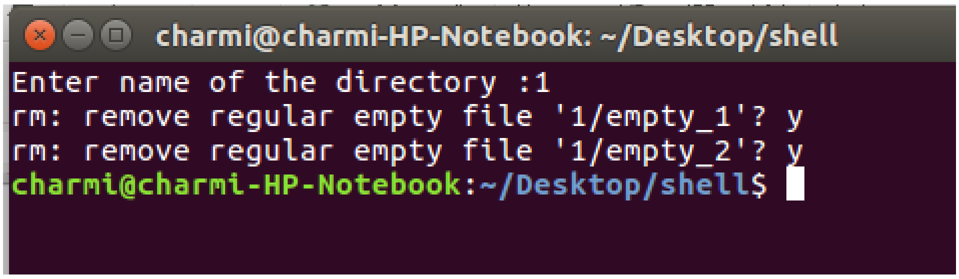
Q.8: Write a script to delete zero sized files from a given directory (and all its sub directories).

Code:

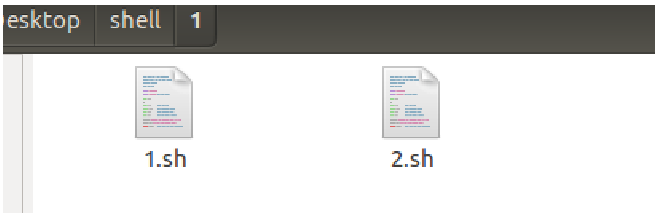
```
clear
echo -n "Enter name of the directory :"
read directory
if [ ! -d "$directory" ]
then
   echo "Directory does not exist"
else
        for i in `find $directory -type f -size 0`
        do
                rm -i $i
        done
fi
```



empty_1 and empty_2 are 0 sized files



```
Enter name of the directory :1
rm: remove regular empty file '1/empty_1'? y
rm: remove regular empty file '1/empty_2'? y
charmi@charmi-HP-Notebook:~/Desktop/shell$
```

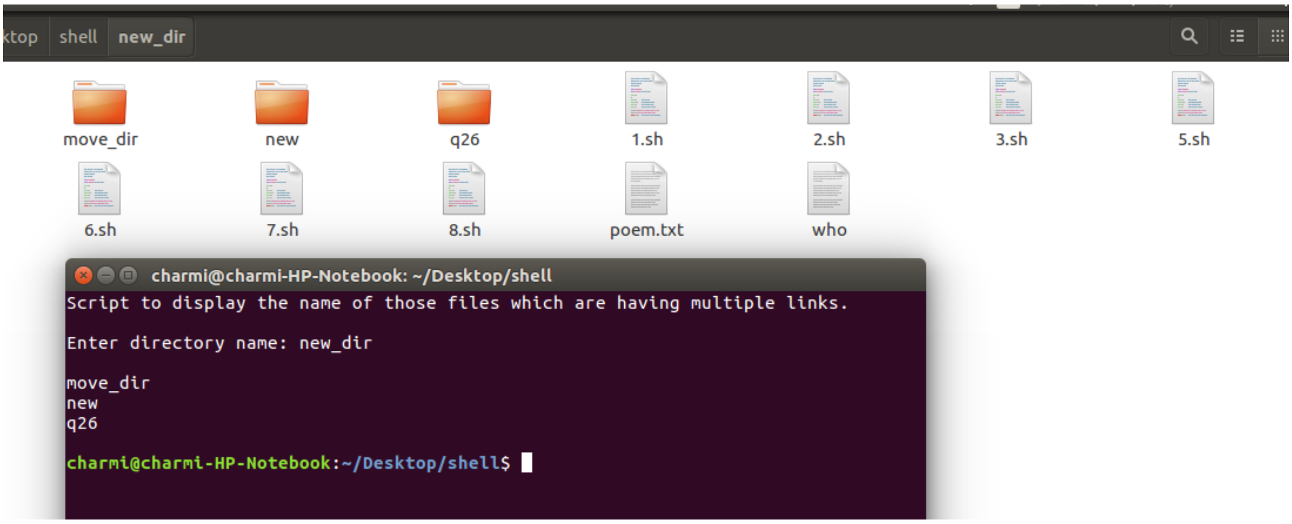


empty files has been removed

Q.9: Write a script to display the name of those files (in the given directory), which are having multiple links.

Code:

```
clear
echo Script to display the name of those files which are having multiple links.
echo
echo -n "Enter directory name: "
read d

ls -l $d | awk '{if($2>1) {print $9}}'
echo
```

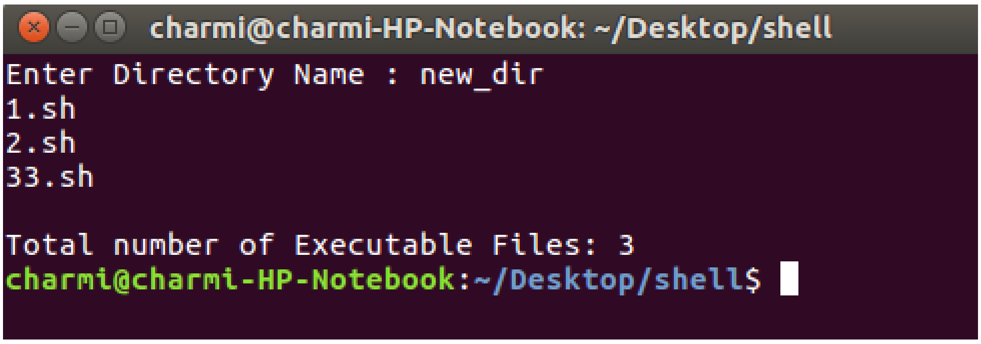


```
Script to display the name of those files which are having multiple links.

Enter directory name: new_dir

move_dir
new
q26

charmi@charmi-HP-Notebook:~/Desktop/shell$
```

Q.10: Write a script to display the name of all executable files in the given directory.

Code:

```
clear
echo -n "Enter name of the directory :"
read directory

if [ ! -d $directory ]
then
   echo "Directory not exist"
else
         count=0
         echo "Files with executable rights are"
fi

for i in $(find $directory -type f -perm +111)
do
      echo $i
      count=$(echo count + 1 | bc -l)
done

if [ $count -eq 0 ]
then
         echo "No files found with executable rights"
fi
```



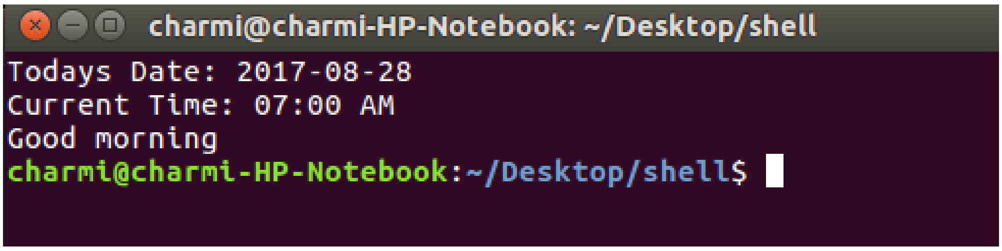
```
Enter Directory Name : new_dir
1.sh
2.sh
33.sh

Total number of Executable Files: 3
charmi@charmi-HP-Notebook:~/Desktop/shell$ 
```

output of the Script

Q.11: Write a script to display the date, time and a welcome message (like Good Morning etc.) The time should be displayed with “a.m.” Or “p.m.” and not in terms of 24 hours notation.

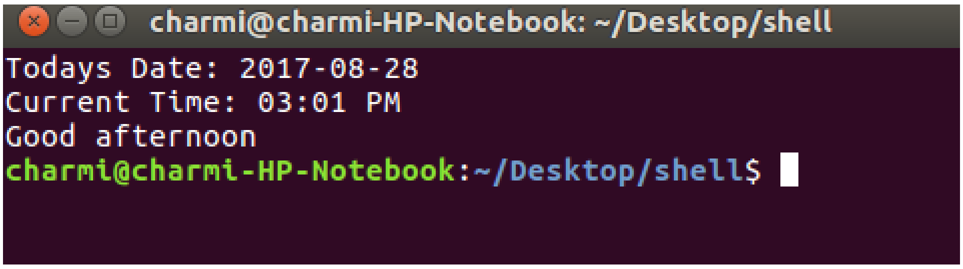
Code:

```
clear
echo -n 'Todays Date: '
date +%Y-%m-%d

echo -n 'Current Time: '
date  +"%I:%M %p"

p=$(date +"%p")
h=$(date +"%I")

if [ "$p" = "AM" ]; then
  echo Good morning
elif [ $h -lt 6 -a $p = PM ]; then
  echo Good afternoon
else
  echo Good evening
fi
```



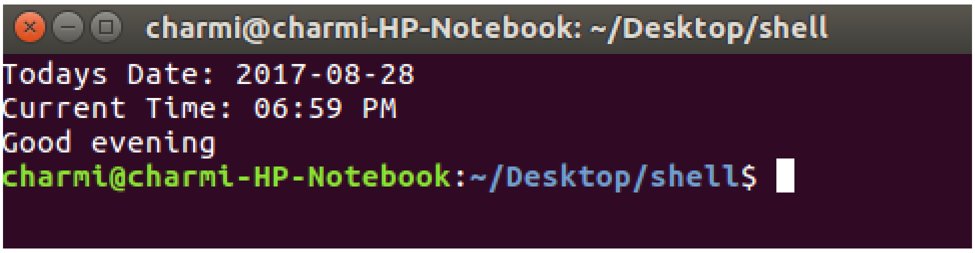
```
Todays Date: 2017-08-28
Current Time: 07:00 AM
Good morning
charmi@charmi-HP-Notebook:~/Desktop/shell$
```

Good morning message at 7:00 AM



```
Todays Date: 2017-08-28
Current Time: 03:01 PM
Good afternoon
charmi@charmi-HP-Notebook:~/Desktop/shell$
```

Good afternoon message at 3:01 PM



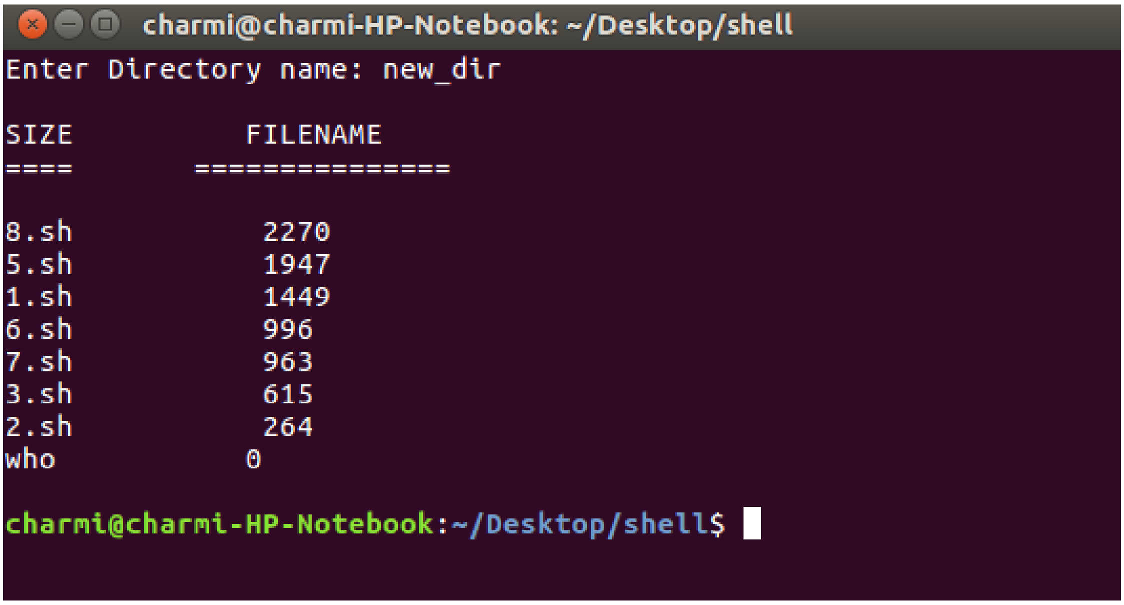
```
Todays Date: 2017-08-28
Current Time: 06:59 PM
Good evening
charmi@charmi-HP-Notebook:~/Desktop/shell$
```

Good evening message at 6:59 PM

Q.12: Write a script to display the directory in the descending order of the size of each file.

Code:

```
clear
echo -n "Enter Directory name: "
read dir

echo
echo "SIZE          FILENAME"
echo "====       ==============="

ls -lS  $dir | awk '{print $9, "         ", $5}'
echo
```



charmi@charmi-HP-Notebook: ~/Desktop/shell

```
Enter Directory name: new_dir

SIZE           FILENAME
====        ===============

8.sh           2270
5.sh           1947
1.sh           1449
6.sh           996
7.sh           963
3.sh           615
2.sh           264
who           0

charmi@charmi-HP-Notebook:~/Desktop/shell$ 
```

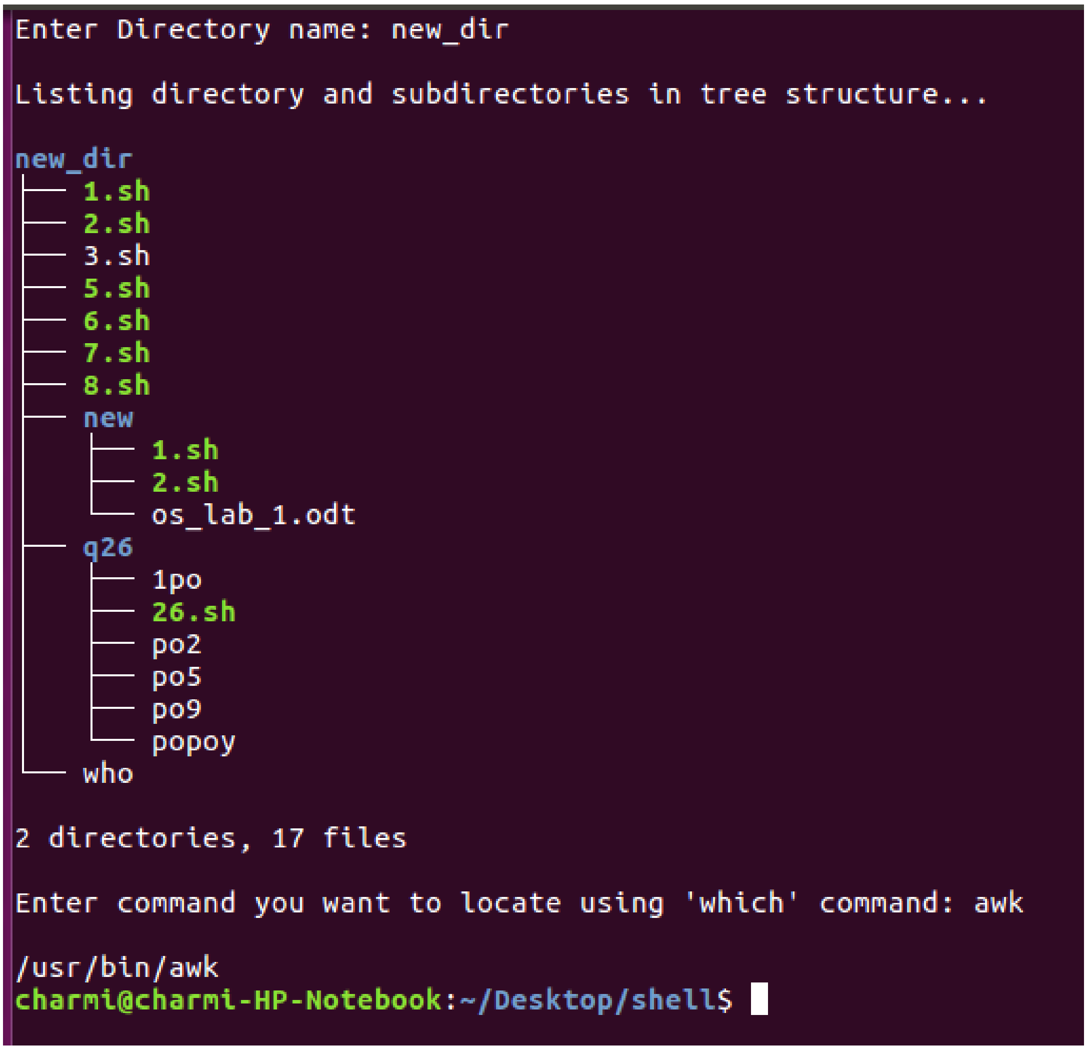
Q.13: Write a script to implement following commands

a. Tree

b. which

Code:

```
clear
echo -n "Enter Directory name: "
read dir

echo
echo "Listing directory and subdirectories in tree structure..."
echo
tree $dir

echo
echo -n "Enter command you want to locate using 'which' command: "
read c
echo
which $c
```



```
Enter Directory name: new_dir

Listing directory and subdirectories in tree structure...

new_dir
├── 1.sh
├── 2.sh
├── 3.sh
├── 5.sh
├── 6.sh
├── 7.sh
├── 8.sh
├── new
│   ├── 1.sh
│   ├── 2.sh
│   └── os_lab_1.odt
├── q26
│   ├── 1po
│   ├── 26.sh
│   ├── po2
│   ├── po5
│   ├── po9
│   └── popoy
└── who

2 directories, 17 files

Enter command you want to locate using 'which' command: awk

/usr/bin/awk
charmi@charmi-HP-Notebook:~/Desktop/shell$
```

Q.14: Write a script to make following file and directory management

- **a. Display current directory**
- **b. List directory**
- **c. Make directory**
- **d. Change directory**
- **e. Copy file**
- **f. Rename file**
- **g. Delete file**
- **h. Edit file**
- **i. Exit**

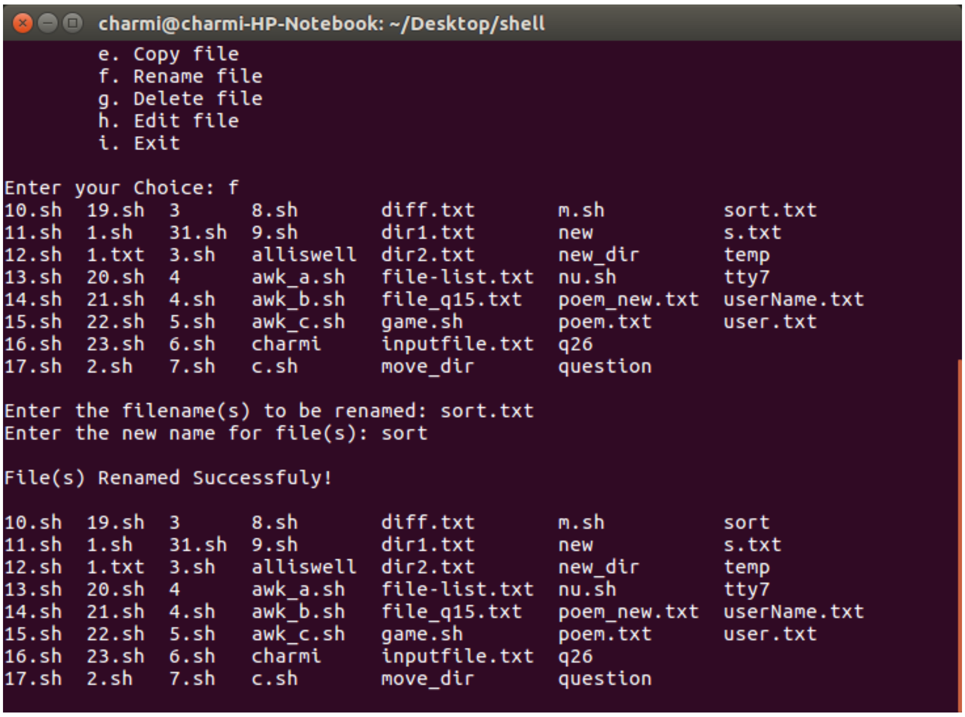
Code:

```
clear
while true
do
clear
cat << MENU

        ***Menu***

        a. Display current directory
        b. List directory
        c. Make directory
        d. Change directory
        e. Copy file
        f. Rename file
        g. Delete file
        h. Edit file
        i. Exit

MENU

echo -n  'Enter your Choice: '
read choice

case $choice in

        [aA])   echo -n "Current directory: "
                        pwd
                        sleep 5
     continue;;

        [bB])   echo -n "Enter directory name: "
                        read d
                        echo
                        echo "Listing directory: " $d
                        ls -l $d | tail -n +2
                        sleep 5
                continue;;
```

```
  [cC])   echo
                 ls
                 echo -n "Enter new directory name: "
                 read d
                 mkdir $d
                 echo "Directory "$d "created successfuly!"
                 echo
                 ls
                 sleep 5
continue;;

  [dD])   echo -n "Current path: "
                 pwd
                 echo -n "Enter directory name where you want to go: "
                 read d
                 cd $d
                 echo
                 echo "You are in Directory "$d
                 echo -n "Current path: "
                 pwd
                 sleep 5
continue;;

  [eE])   ls
                 echo
                 echo -n 'Enter the filename(s) to be copied: '
                 read filename
                 echo -n 'Enter the destination for file(s) to be copied: '
                 read dest

                 cp -i $filename $dest
                 echo
                 echo 'File(s) Copied Successfuly!'
                 echo
                 ls
                 sleep 5

          continue;;

  [fF])   ls
                 echo
                 echo -n  "Enter the filename(s) to be renamed: "
                 read filename
                 echo -n  'Enter the destination for file(s) to be moved: '
                 read dest

                 mv -i $filename $dest
                 echo
                 echo 'File(s) Renamed Successfuly!'
                 echo
                 ls
                 sleep 5
```

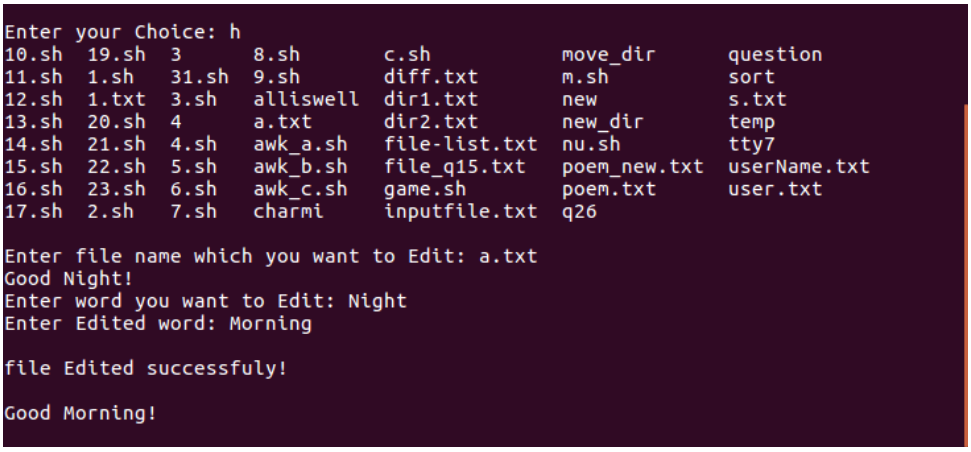
```
    continue;;

      [gG])   ls
                  echo
                  echo -n  'Enter the filename(s) to be deleted: '
                  read filename

                  rm -i $filename
                  echo
                  echo 'File(s) Deleted successfuly!'
                  echo
                  ls
                  sleep 5
    continue;;

      [hH])   ls
                  echo
                  echo -n "Enter file name which you want to Edit: "
                  read filename

                  cat $filename
                  sleep 5

                  echo -n "Enter word you want to Edit: "
                  read old

                  echo -n "Enter Edited word: "
                  read new

                  sed -i s/$old/$new/ $filename
                  echo
                  echo "file Edited successfuly!"
                  echo
                  cat $filename
                  sleep 5

            continue;;

      [iI])  echo 'Thank You!'
            exit;;

      *) echo 'Please enter proper Choice.'

esac
done
```

Some Snap Shots of the output of above script.



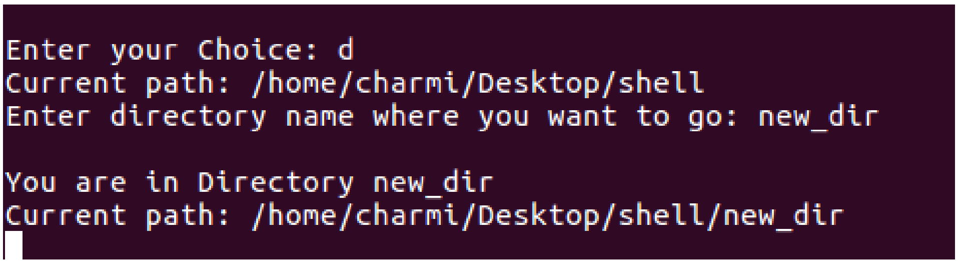
```
charmi@charmi-HP-Notebook: ~/Desktop/shell
        e. Copy file
        f. Rename file
        g. Delete file
        h. Edit file
        i. Exit

Enter your Choice: f
10.sh  19.sh  3      8.sh       diff.txt       m.sh          sort.txt
11.sh  1.sh   31.sh  9.sh       dir1.txt       new           s.txt
12.sh  1.txt  3.sh   alliswell  dir2.txt       new_dir       temp
13.sh  20.sh  4      awk_a.sh   file-list.txt  nu.sh         tty7
14.sh  21.sh  4.sh   awk_b.sh   file_q15.txt   poem_new.txt  userName.txt
15.sh  22.sh  5.sh   awk_c.sh   game.sh        poem.txt      user.txt
16.sh  23.sh  6.sh   charmi     inputfile.txt  q26
17.sh  2.sh   7.sh   c.sh       move_dir       question

Enter the filename(s) to be renamed: sort.txt
Enter the new name for file(s): sort

File(s) Renamed Successfuly!

10.sh  19.sh  3      8.sh       diff.txt       m.sh          sort
11.sh  1.sh   31.sh  9.sh       dir1.txt       new           s.txt
12.sh  1.txt  3.sh   alliswell  dir2.txt       new_dir       temp
13.sh  20.sh  4      awk_a.sh   file-list.txt  nu.sh         tty7
14.sh  21.sh  4.sh   awk_b.sh   file_q15.txt   poem_new.txt  userName.txt
15.sh  22.sh  5.sh   awk_c.sh   game.sh        poem.txt      user.txt
16.sh  23.sh  6.sh   charmi     inputfile.txt  q26
17.sh  2.sh   7.sh   c.sh       move_dir       question
```

Renaming file sort.txt to sort



```
Enter your Choice: h
10.sh  19.sh  3      8.sh       c.sh           move_dir      question
11.sh  1.sh   31.sh  9.sh       diff.txt       m.sh          sort
12.sh  1.txt  3.sh   alliswell  dir1.txt       new           s.txt
13.sh  20.sh  4      a.txt      dir2.txt       new_dir       temp
14.sh  21.sh  4.sh   awk_a.sh   file-list.txt  nu.sh         tty7
15.sh  22.sh  5.sh   awk_b.sh   file_q15.txt   poem_new.txt  userName.txt
16.sh  23.sh  6.sh   awk_c.sh   game.sh        poem.txt      user.txt
17.sh  2.sh   7.sh   charmi     inputfile.txt  q26

Enter file name which you want to Edit: a.txt
Good Night!
Enter word you want to Edit: Night
Enter Edited word: Morning

file Edited successfuly!

Good Morning!
```

Editing file a.txt which contains a string "Good Night!"



```
Enter your Choice: d
Current path: /home/charmi/Desktop/shell
Enter directory name where you want to go: new_dir

You are in Directory new_dir
Current path: /home/charmi/Desktop/shell/new_dir
```

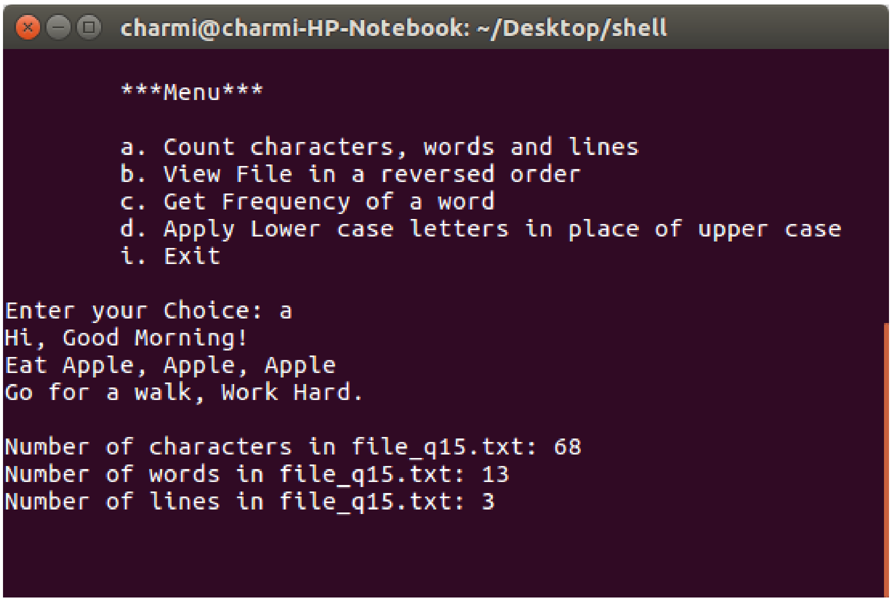
Output of option Change Directory

Q.15: Write a script which reads a text file and output the following:

a. Count of characters, words and lines
b. File in a reversed order
c. Frequency of particular word in the file
d. Lower case letters in place of upper case alphabets

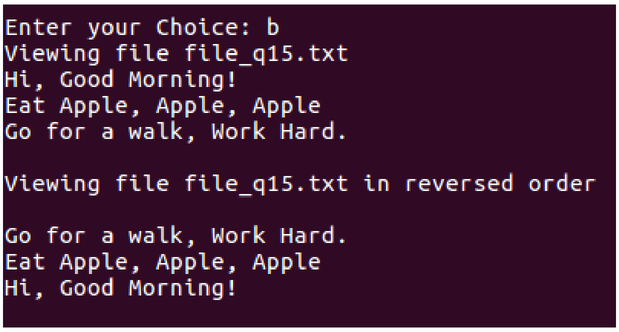
Code:

```
clear

echo -n "Enter file name: "
read file

while true
do
clear
cat << MENU

        ***Menu***

        a. Count characters, words and lines
        b. View File in a reversed order
        c. Get Frequency of a word
        d. Apply Lower case letters in place of upper case
        i. Exit

MENU

echo -n  'Enter your Choice: '
read choice

case $choice in

        [aA])   cat $file
                c=`cat $file | wc -m`
                w=`cat $file | wc -w`
                l=`cat $file | wc -l`

                echo
                echo Number of characters in $file: $c
                echo Number of words in $file: $w
                echo Number of lines in $file: $l
                sleep 5
     continue;;

        [bB])   echo Viewing file $file
                cat $file
                echo
                echo Viewing file $file in reversed order
                echo
                tac $file
                sleep 5
                continue;;

         [cC])   cat $file
                           echo
                           echo -n "Enter word to find Frequency: "
                           read word
                           fre=`grep -o "$word" $file | wc -l`
                           echo
                           echo Frequency of $word is $fre
                           sleep 5
      continue;;

         [dD])   echo Viewing file $file
                           cat $file
                           echo
                           echo Viewing file $file after changing Case
                           echo
                           # tr '[:upper:]' '[:lower:]' < $file
                           tr A-Z a-z < $file
                           echo
                           sleep 5
      continue;;


         [iI])  echo 'Thank You!'
                exit;;

         *) echo 'Please enter proper Choice.'
           sleep 5

esac
done
```



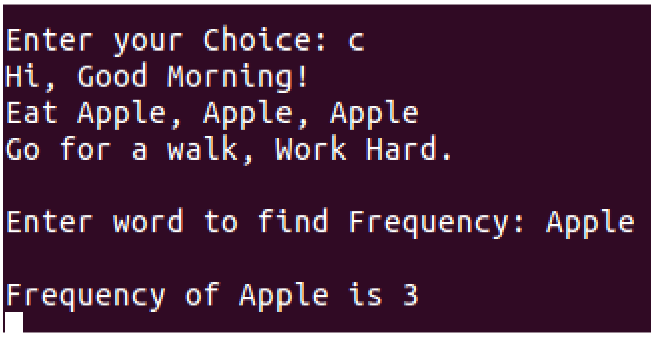
```
charmi@charmi-HP-Notebook: ~/Desktop/shell

        ***Menu***

        a. Count characters, words and lines
        b. View File in a reversed order
        c. Get Frequency of a word
        d. Apply Lower case letters in place of upper case
        i. Exit

Enter your Choice: a
Hi, Good Morning!
Eat Apple, Apple, Apple
Go for a walk, Work Hard.

Number of characters in file_q15.txt: 68
Number of words in file_q15.txt: 13
Number of lines in file_q15.txt: 3
```

output of choice a



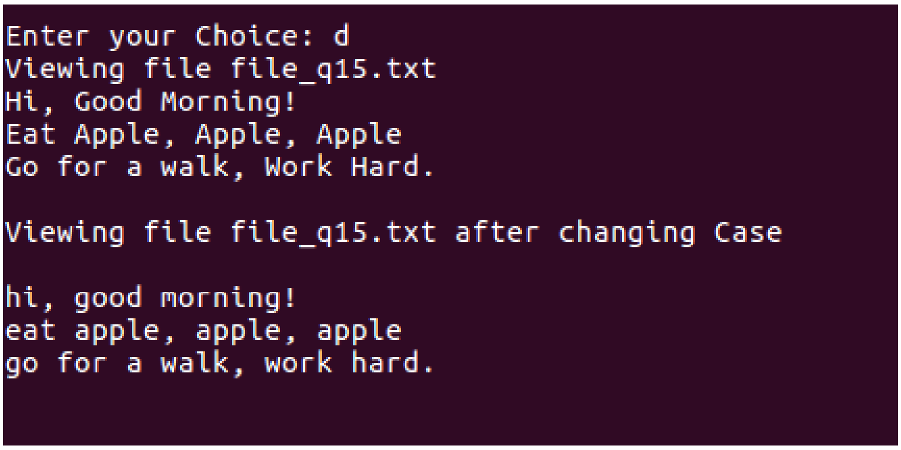
```
Enter your Choice: b
Viewing file file_q15.txt
Hi, Good Morning!
Eat Apple, Apple, Apple
Go for a walk, Work Hard.

Viewing file file_q15.txt in reversed order

Go for a walk, Work Hard.
Eat Apple, Apple, Apple
Hi, Good Morning!
```

output of choice b



```
Enter your Choice: c
Hi, Good Morning!
Eat Apple, Apple, Apple
Go for a walk, Work Hard.

Enter word to find Frequency: Apple

Frequency of Apple is 3
```

output of choice c



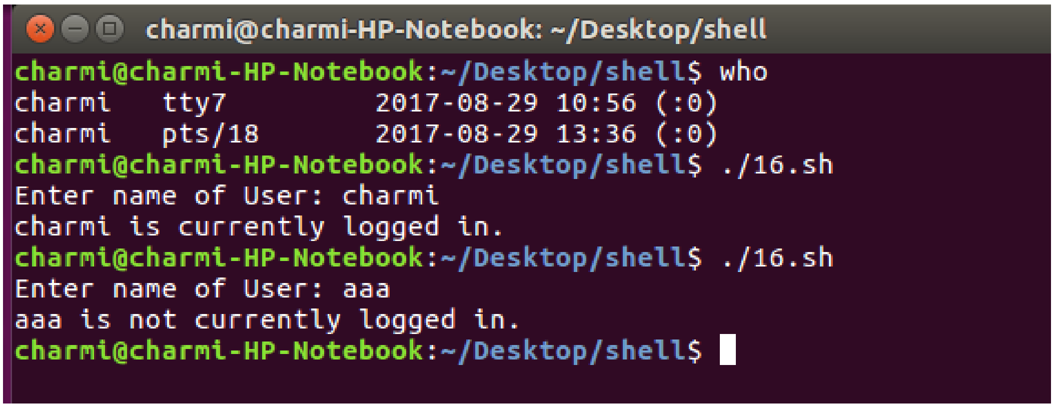
```
Enter your Choice: d
Viewing file file_q15.txt
Hi, Good Morning!
Eat Apple, Apple, Apple
Go for a walk, Work Hard.

Viewing file file_q15.txt after changing Case

hi, good morning!
eat apple, apple, apple
go for a walk, work hard.
```

output of choice d

Q.16: Write a shell script to ask for the name of a user, and check whether that user is currently online or not.

Code:

```
echo -n 'Enter name of User: '
read user
if [[ "$USER" = "$user" ]]; then
        echo $user is currently logged in.
else
        echo $user is not currently logged in.
fi
```



```
charmi@charmi-HP-Notebook:~/Desktop/shell$ who
charmi   tty7         2017-08-29 10:56 (:0)
charmi   pts/18       2017-08-29 13:36 (:0)
charmi@charmi-HP-Notebook:~/Desktop/shell$ ./16.sh
Enter name of User: charmi
charmi is currently logged in.
charmi@charmi-HP-Notebook:~/Desktop/shell$ ./16.sh
Enter name of User: aaa
aaa is not currently logged in.
charmi@charmi-HP-Notebook:~/Desktop/shell$
```

Q.17: Do operations on file poem

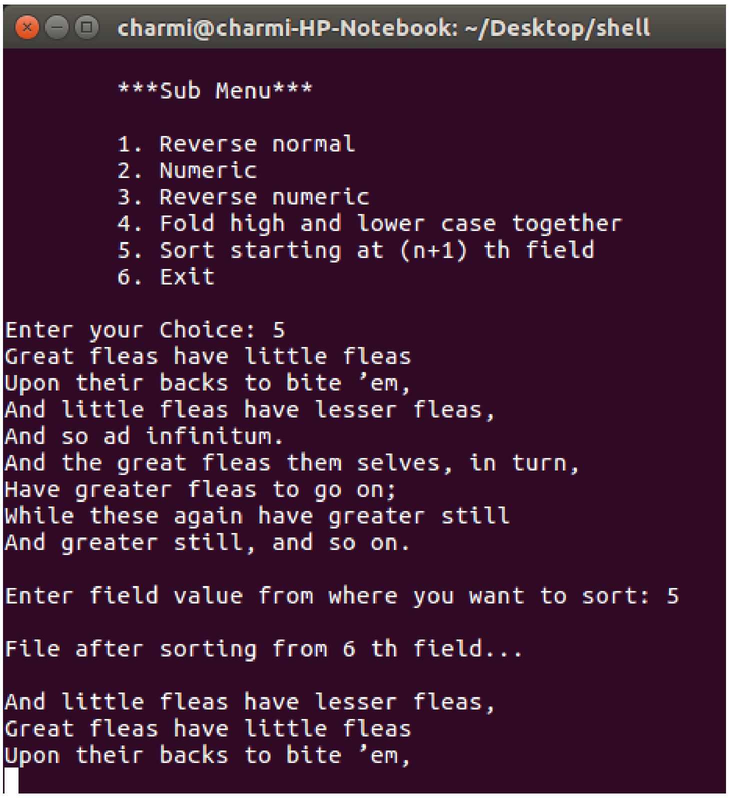
Code:

```
clear
while true
do
clear
cat << MENU

        ***Menu***

        1. Count the lines, words, and characters
        2. Pick up the lines containing word ‘fleas’
        3. Pick up the lines not containing word ‘fleas’
        4. Sort the file poem in line-by-line fashion
        5. Print last three lines of the file
        6. Print last lines starting from 3rd line
        7. Create two files poem and poem_new with different contents and compare them
        8. Exit

MENU

file='/home/charmi/Desktop/shell/poem.txt'

echo -n  'Enter your Choice: '
read choice

case $choice in

        1)      c=`cat $file | wc -m`
                w=`cat $file | wc -w`
                l=`cat $file | wc -l`

                echo
                echo "Number of characters: " $c
                echo "Number of words:      " $w
                echo "Number of lines:      " $l
                sleep 5
   continue;;

        2)      echo
                echo Viewing lines containing word fleas from file poem
                echo
                word='fleas'
                grep -r "$word" $file
                sleep 5
        continue;;

        3)  echo
                echo Viewing lines NOT containing word fleas from file poem
                echo
			word='fleas'
			grep -rv "$word" $file
			sleep 5
   continue;;

		4)
while :
do
clear
cat << MENU

		***Sub Menu***

		1. Reverse normal
		2. Numeric
		3. Reverse numeric
		4. Fold high and lower case together
		5. Sort starting at (n+1) th field
		6. Exit

MENU

echo -n  'Enter your Choice: '
read c

case $c in

		1)	echo Sorting file poem in Reverse order...
			echo

			sort -r $file
			sleep 5
		continue;;

		2)	echo Sorting file poem in Numeric order...
			echo

			sort -n $file
			sleep 5
		continue;;

		3)	echo Sorting file poem in Reverse Numeric order...
			echo

			sort -rn $file
			sleep 5
		continue;;

		4)	echo Fold high and lower case together...
			echo

			sort -f $file
```

```
            sleep 5
      continue;;

      5)    cat $file
            echo
            echo -n "Enter field value from where you want to sort: "
            read n
            let a=`(cat $file | wc -l)`-$n
            echo
            let nn=`expr $n+1`
            echo "File after sorting from $nn th field..."
            echo
            head -n $a $file | sort
            sleep 5
      continue;;

      6)  echo 'Exiting Sub Menu'
            sleep 3
      break;;

      *)    echo 'Please enter proper Choice.'
            sleep 5
esac
done
;;

  5)  echo Printing last 3 lines of file poem
            echo

            tail -3 $file
            slepp 5
  continue;;

  6)  echo Viewing last lines starting from 3rd line
            echo

            tail -n +3 $file

            sleep 5
  continue;;

  7)  echo
            echo Comparing files poem.txt and poem_new.txt...

            f='/home/charmi/Desktop/shell/poem_new.txt'

            cmp -s $file $f

            if [ $? -eq 1 ]; then
              echo Files are different
            else
              echo Files are not different
```

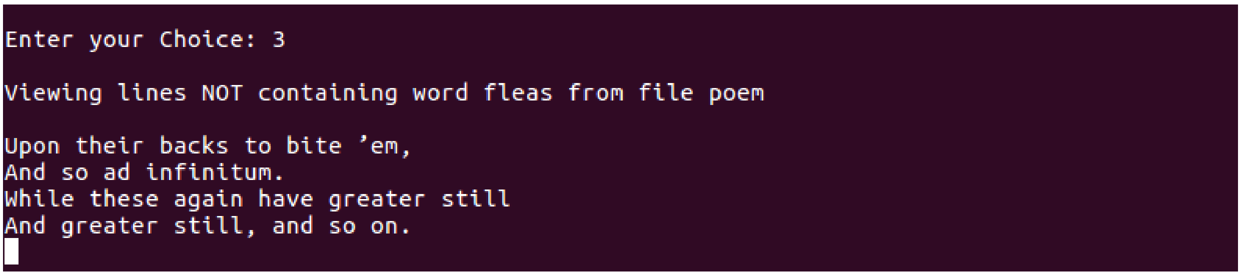
```
                fi

                sleep 5
        continue;;

            8)  echo 'Thank You!'
                exit;;

            *)      echo 'Please enter proper Choice.'
                sleep 5

      esac
            done
```

Snap Shots of some output of above script

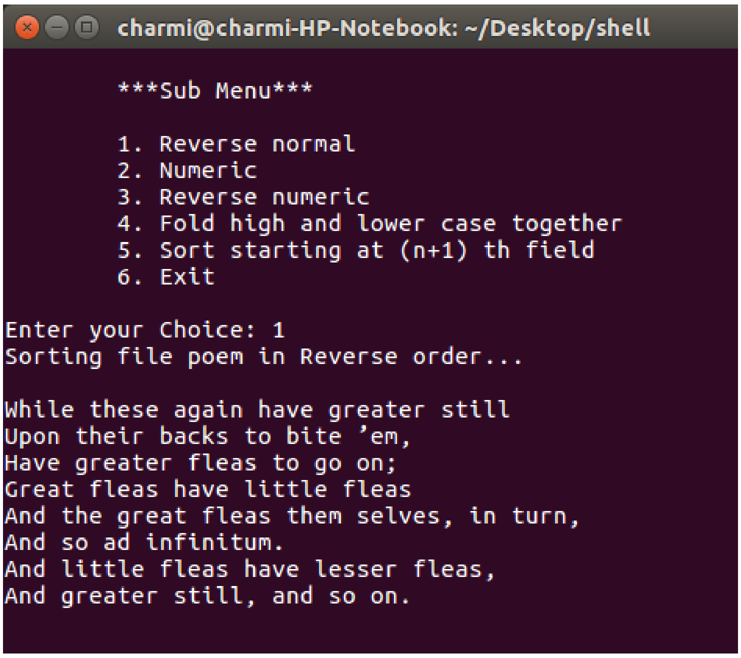


```
        ***Sub Menu***

        1. Reverse normal
        2. Numeric
        3. Reverse numeric
        4. Fold high and lower case together
        5. Sort starting at (n+1) th field
        6. Exit

Enter your Choice: 5
Great fleas have little fleas
Upon their backs to bite ’em,
And little fleas have lesser fleas,
And so ad infinitum.
And the great fleas them selves, in turn,
Have greater fleas to go on;
While these again have greater still
And greater still, and so on.

Enter field value from where you want to sort: 5

File after sorting from 6 th field...

And little fleas have lesser fleas,
Great fleas have little fleas
Upon their backs to bite ’em,
```

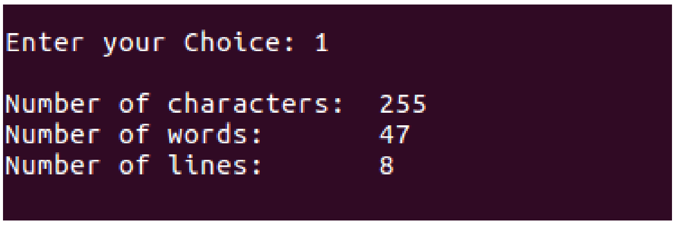
Output of sort field from n+1



```
Enter your Choice: 3

Viewing lines NOT containing word fleas from file poem

Upon their backs to bite ’em,
And so ad infinitum.
While these again have greater still
And greater still, and so on.
```



```
        ***Sub Menu***

        1. Reverse normal
        2. Numeric
        3. Reverse numeric
        4. Fold high and lower case together
        5. Sort starting at (n+1) th field
        6. Exit

Enter your Choice: 1
Sorting file poem in Reverse order...

While these again have greater still
Upon their backs to bite ’em,
Have greater fleas to go on;
Great fleas have little fleas
And the great fleas them selves, in turn,
And so ad infinitum.
And little fleas have lesser fleas,
And greater still, and so on.
```



```
Enter your Choice: 1

Number of characters:  255
Number of words:       47
Number of lines:       8
```

Q.18: Explain output for the following
$ls > temp
$wc temp > temp

$ ls > temp

- “>” will **redirects output** of a command “ls” to a file named “temp”.
- Firstly this will **make a file** named “temp” in current directory (if it do not exist else overwrites it).
- Then the output of command “ls” will be writen in “temp” file which is **list of all files** and sub-directories present in current directory.

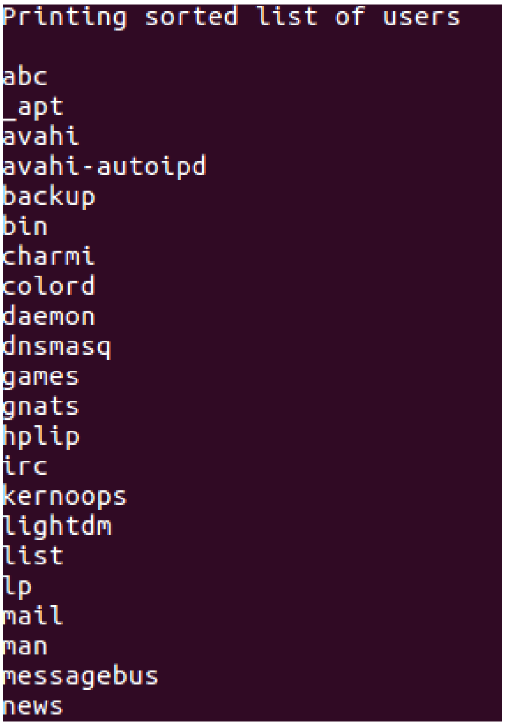
$ wc temp > temp

- Since the file “temp” is already present in current directory due to previous step, now due to “>” symbol, file named “temp” will be **overwritten** and will become **empty** temporarily.
- After that “wc” command will try to count number of characters, words and lines present in file “temp”. Since the file is empty output of command “wc” will be 0, 0, 0 respectivly.
- Now this output will be overwriten in file “temp” as:
    - no of characters   no of words   no of lines   filename (here **0 0 0 temp**)

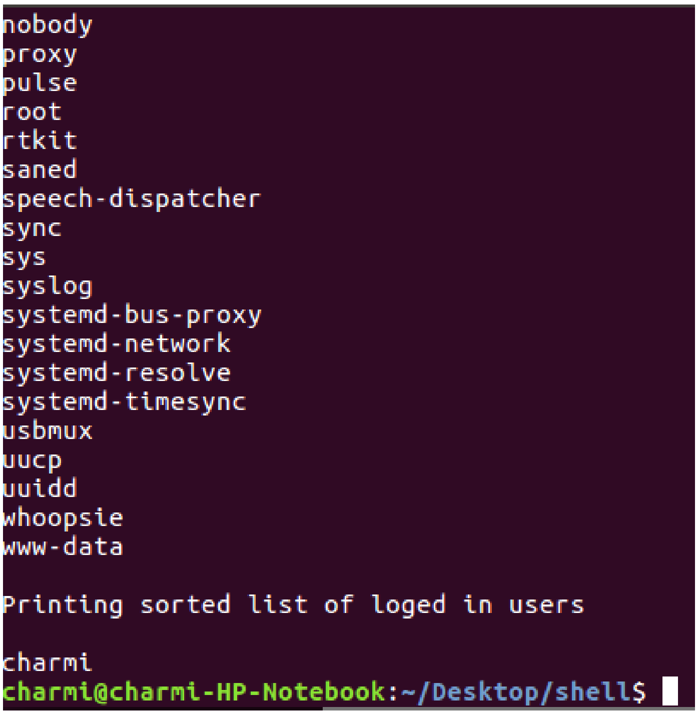
Q.19: Print sorted list of users

Code:

```
clear
echo 'Printing sorted list of users'
echo
cut -d ":" -f 1 /etc/passwd | sort
echo
echo 'Printing sorted list of loged in users'
echo
users | sort
```



```
Printing sorted list of users

abc
_apt
avahi
avahi-autoipd
backup
bin
charmi
colord
daemon
dnsmasq
games
gnats
hplip
irc
kernoops
lightdm
list
lp
mail
man
messagebus
news
```

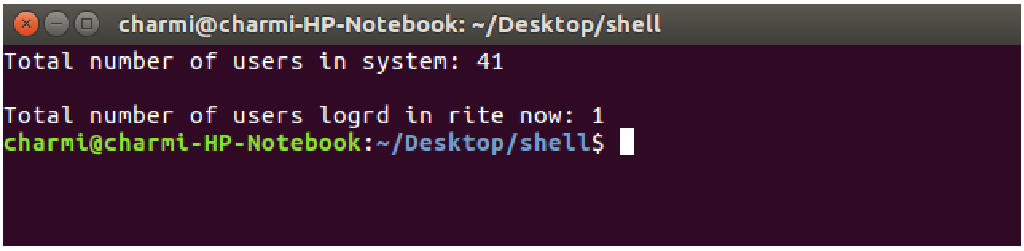


```
nobody
proxy
pulse
root
rtkit
saned
speech-dispatcher
sync
sys
syslog
systemd-bus-proxy
systemd-network
systemd-resolve
systemd-timesync
usbmux
uucp
uuidd
whoopsie
www-data

Printing sorted list of loged in users

charmi
charmi@charmi-HP-Notebook:~/Desktop/shell$
```

Q.20: Count the users

Code:

```
clear
echo -n 'Total number of users in system: '
cut -d ":" -f 1 /etc/passwd | wc -l
echo
echo -n 'Total number of users loged in rite now: '
who | wc -l
```

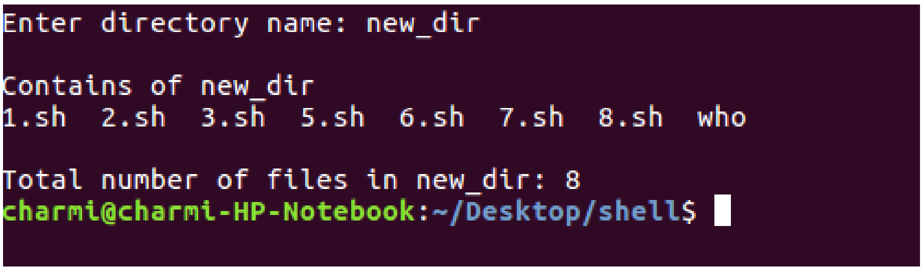


```
Total number of users in system: 41

Total number of users logrd in rite now: 1
charmi@charmi-HP-Notebook:~/Desktop/shell$
```

Q.21: Count the total files

Code:

```
clear
echo -n 'Enter directory name: '
read dir
echo
echo Contains of $dir
ls $dir
echo
echo -n 'Total number of files in '$dir': '
ls $dir | wc -l
```

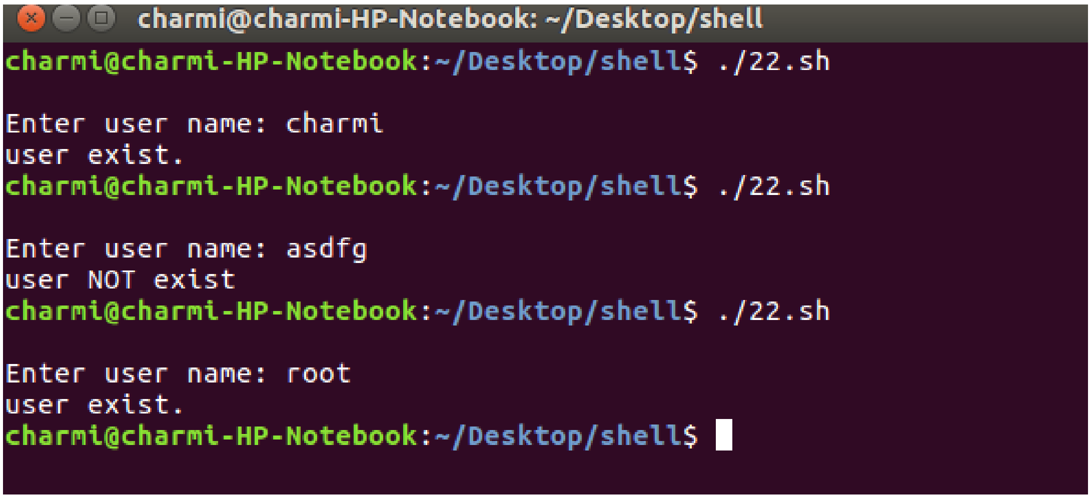


```
Enter directory name: new_dir

Contains of new_dir
1.sh  2.sh  3.sh  5.sh  6.sh  7.sh  8.sh  who

Total number of files in new_dir: 8
charmi@charmi-HP-Notebook:~/Desktop/shell$ 
```

Q.22: Look for a particular user

Code:

```
echo -n 'Enter user name: '
read un
grep -o '^[a-zA-Z]*' /etc/passwd > userName.txt

j=0

for i in `cat userName.txt`
do
	if [[ "$i" = "$un" ]];
	then
		echo user exist.
		j=1
		# break;
	fi
done

if [[ $j -eq 0 ]]; then
	echo user NOT exist
fi
```

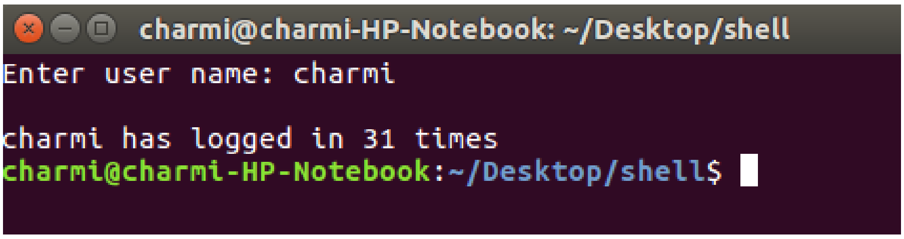


```
charmi@charmi-HP-Notebook:~/Desktop/shell$ ./22.sh

Enter user name: charmi
user exist.
charmi@charmi-HP-Notebook:~/Desktop/shell$ ./22.sh

Enter user name: asdfg
user NOT exist
charmi@charmi-HP-Notebook:~/Desktop/shell$ ./22.sh

Enter user name: root
user exist.
charmi@charmi-HP-Notebook:~/Desktop/shell$
```

Q.23: Count how many times you have logged in.

Code:

```
clear
echo -n 'Enter user name: '
read un
n=`last | grep $un | wc -l`
echo
echo $un has logged in $n times
```



```
Enter user name: charmi

charmi has logged in 31 times
charmi@charmi-HP-Notebook:~/Desktop/shell$
```

Q.24 & Q.25: Explain difference between
$ who | sort and who > sort

$ who | sort

- " | " a **pipe** is technique for passing information (output of any program process/ command) from one process to another.
- Here, output of command "who" will be passed to command "sort" as input.
- "who" command will give currently loged in user names along with their line, date-time and comment as output.
- This data will be given to command "sort" as input. As name suggest, this command will sort the inputed data in alphabatical order (here by user name) and will print it on terminal.

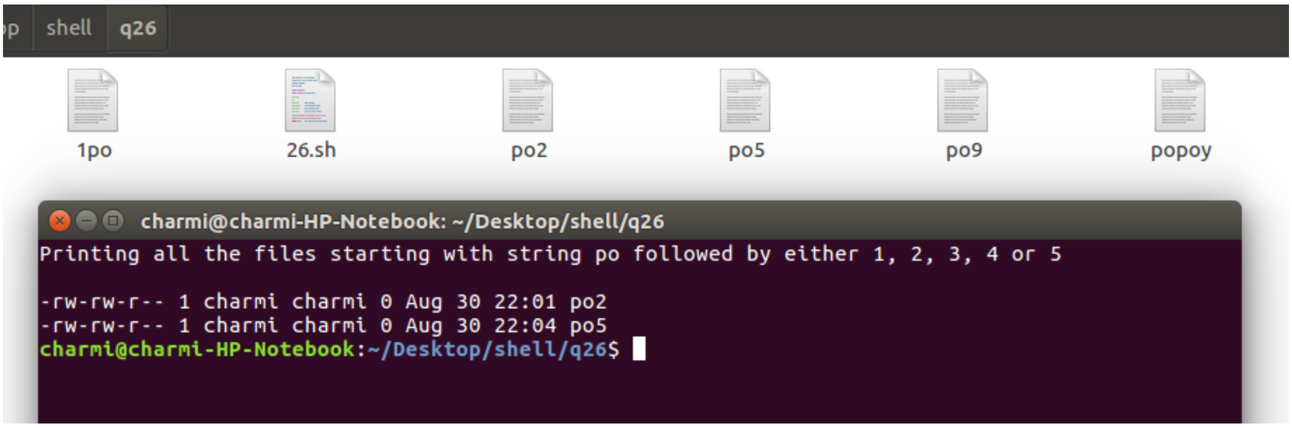
$ who > sort

- “>” will **redirects output** of a command “who” to a file named “sort”.
- Firstly this will **make a file** named “sort” in current directory (if it do not exist else overwrites it).
- Then the output of command “who” will be writen in “sort” file which is the currently loged in user names along with their line, date-time and comment.

Q.26: List detailed attributes of all files that have names beginning with “po” followed by either 1,2,3,4, or 5

Code:

```
clear
echo Printing all the files starting with string po followed by either 1, 2, 3, 4 or 5
echo
ls -la | grep "po[1-5]"
```

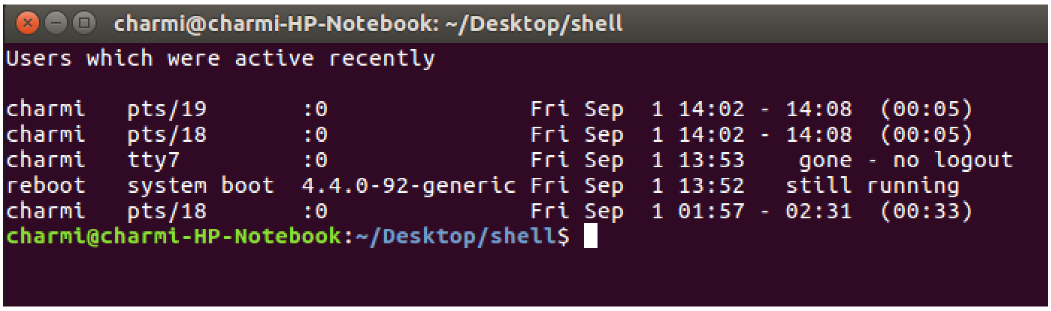


```
Printing all the files starting with string po followed by either 1, 2, 3, 4 or 5

-rw-rw-r-- 1 charmi charmi 0 Aug 30 22:01 po2
-rw-rw-r-- 1 charmi charmi 0 Aug 30 22:04 po5
charmi@charmi-HP-Notebook:~/Desktop/shell/q26$
```

Q.27: How can you tell if a user has been active at the terminal recently?

Code:

```
clear
echo Users which were active recently
echo
last | head -5
```

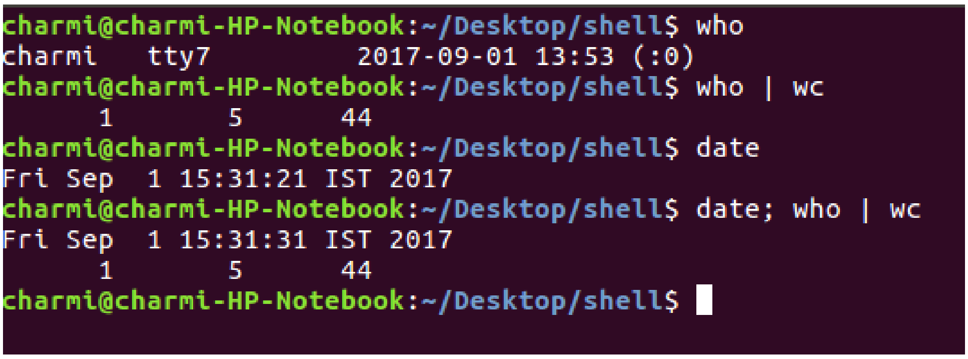


```
Users which were active recently

charmi   pts/19       :0               Fri Sep  1 14:02 - 14:08  (00:05)
charmi   pts/18       :0               Fri Sep  1 14:02 - 14:08  (00:05)
charmi   tty7         :0               Fri Sep  1 13:53   gone - no logout
reboot   system boot  4.4.0-92-generic Fri Sep  1 13:52   still running
charmi   pts/18       :0               Fri Sep  1 01:57 - 02:31  (00:33)
charmi@charmi-HP-Notebook:~/Desktop/shell$
```

Q.28: Find difference between
$ date ; who | wc and
$ (date;who) | wc

$ date ; who | wc

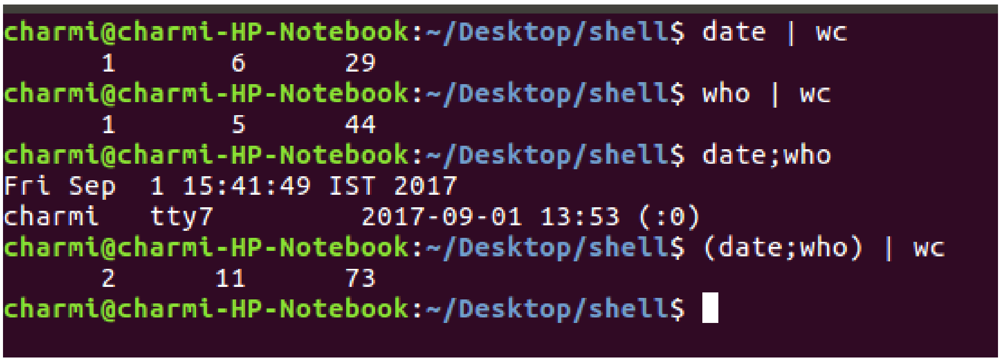
- "date" command will print system date and time on terminal.
- " ; " a semicolon is technique to separate two lines, here it is used to separete two command date and who | wc.
- " | " a **pipe** is technique for passing information (output of any program process/ command) from one process to another.
- Here, output of command "who" will be passed to command "wc" as input.
- "who" command will give currently loged in user names along with their line, date-time and comment as output.
- This data will be given to command "wc" as input. "wc" command will count number of lines, words and characters of this input (here it is 1 5 and 44 respectively) and will print it on terminal.



```
charmi@charmi-HP-Notebook:~/Desktop/shell$ who
charmi   tty7         2017-09-01 13:53 (:0)
charmi@charmi-HP-Notebook:~/Desktop/shell$ who | wc
      1       5      44
charmi@charmi-HP-Notebook:~/Desktop/shell$ date
Fri Sep  1 15:31:21 IST 2017
charmi@charmi-HP-Notebook:~/Desktop/shell$ date; who | wc
Fri Sep  1 15:31:31 IST 2017
      1       5      44
charmi@charmi-HP-Notebook:~/Desktop/shell$ 
```

$ (date;who) | wc

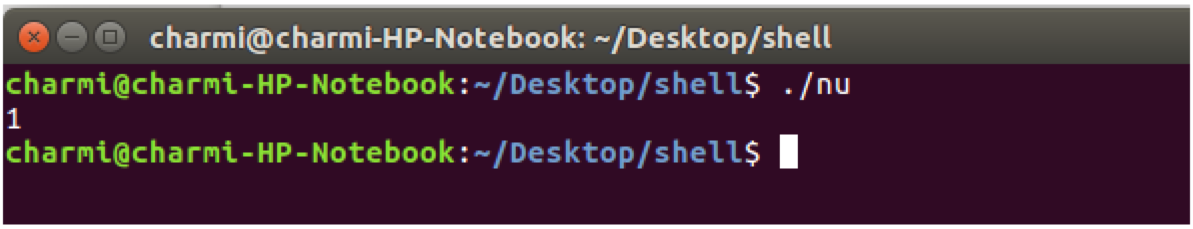
- " ( ) " a round bracket has higher presison that " | " pipe command.
- Here firstly date;who command will run, which is of 2 lines output shown below.
- After that "wc" command will count it's number of lines, words and characters and print on terminal.
- Besically, this numbers are sum of the output of commands "date | wc" and "who | wc".



```
charmi@charmi-HP-Notebook:~/Desktop/shell$ date | wc
      1       6      29
charmi@charmi-HP-Notebook:~/Desktop/shell$ who | wc
      1       5      44
charmi@charmi-HP-Notebook:~/Desktop/shell$ date;who
Fri Sep  1 15:41:49 IST 2017
charmi   tty7         2017-09-01 13:53 (:0)
charmi@charmi-HP-Notebook:~/Desktop/shell$ (date;who) | wc
      2      11      73
charmi@charmi-HP-Notebook:~/Desktop/shell$ 
```

Q.29: Create a file named ‘nu’ that contains
‘who | wc –l’ and run it on shell.

This will give us the number of users currently logged in the system.



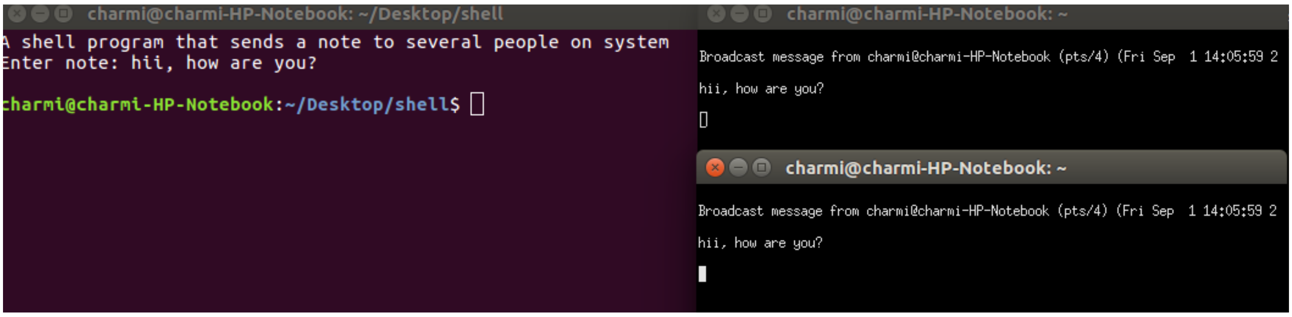
charmi@charmi-HP-Notebook: ~/Desktop/shell

```
charmi@charmi-HP-Notebook:~/Desktop/shell$ ./nu
1
charmi@charmi-HP-Notebook:~/Desktop/shell$ 
```

Q.30: Write a shell program that sends a note to several people on your system.

Code:

```
clear
echo A shell program that sends a note to several people on system

echo -n "Enter note: "
read note
echo
wall $note
```



charmi@charmi-HP-Notebook: ~/Desktop/shell

```
A shell program that sends a note to several people on system
Enter note: hii, how are you?

charmi@charmi-HP-Notebook:~/Desktop/shell$ 
```

charmi@charmi-HP-Notebook: ~

```
Broadcast message from charmi@charmi-HP-Notebook (pts/4) (Fri Sep  1 14:05:59 2

hii, how are you?
```

charmi@charmi-HP-Notebook: ~

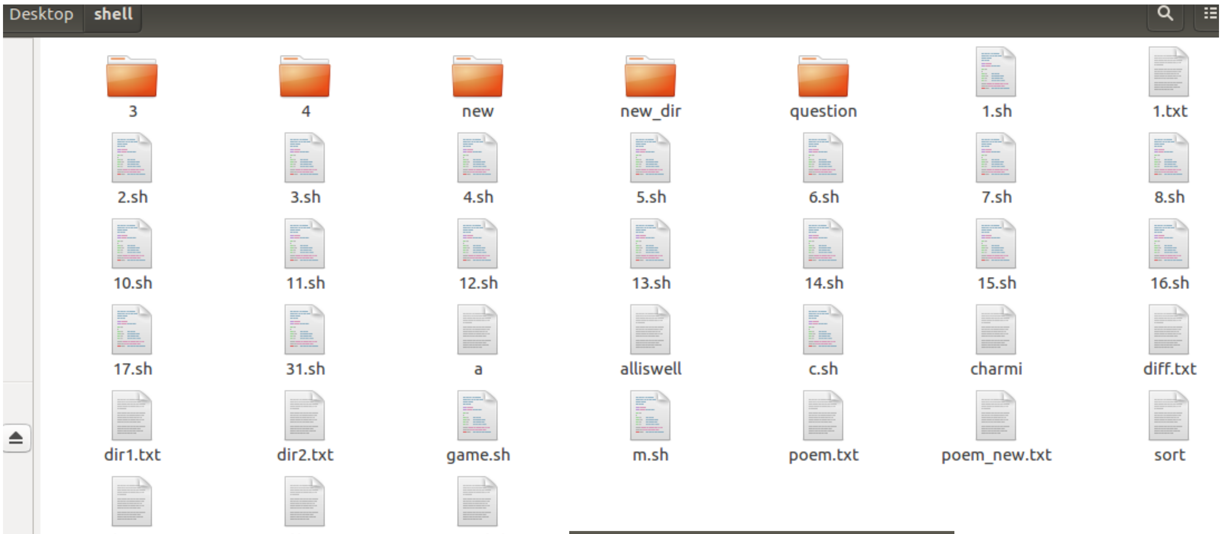
```
Broadcast message from charmi@charmi-HP-Notebook (pts/4) (Fri Sep  1 14:05:59 2

hii, how are you?
```

Q.31: Use a for loop to move a list of files in the current directory to another directory move all your files to another directory.

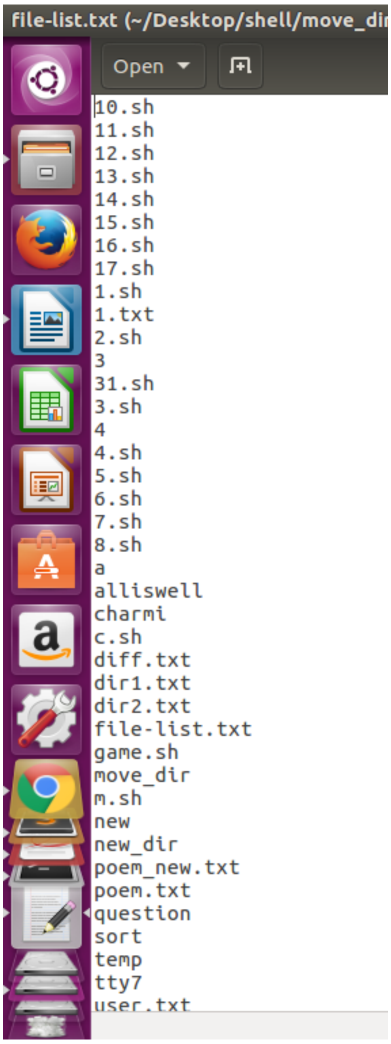
Code:

```
clear
echo program to move file-list and files from current directory to another directory using for loop
echo

echo -n 'Enter the destination Directory for file(s) to be moved: '
read TargetDir

mkdir $TargetDir
ls `pwd` > file-list.txt
```

```
# moving a list of files in the current directory to another directory
for i in `ls pwd`
do
        mv $i $TargetDir/file-list.txt
done
# moving all files in another directory
for FileFromDir1 in `cat file-list.txt`
do
        mv $FileFromDir1 $TargetDir
done
```

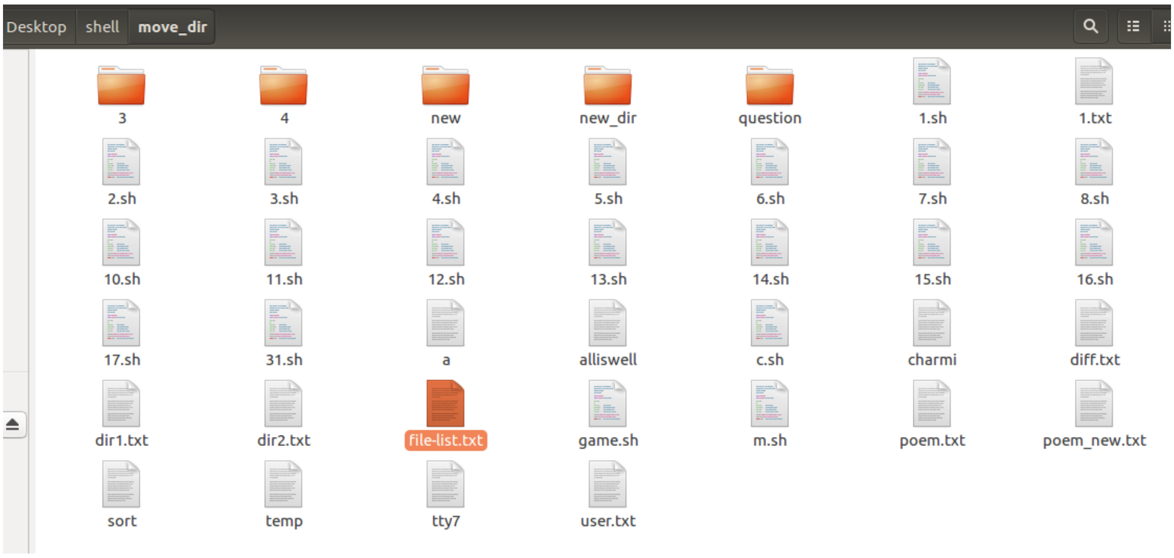


Contains of current directory



```
10.sh
11.sh
12.sh
13.sh
14.sh
15.sh
16.sh
17.sh
1.sh
1.txt
2.sh
3
31.sh
3.sh
4
4.sh
5.sh
6.sh
7.sh
8.sh
a
alliswell
charmi
c.sh
diff.txt
dir1.txt
dir2.txt
file-list.txt
game.sh
move_dir
m.sh
new
new_dir
poem_new.txt
poem.txt
question
sort
temp
tty7
user.txt
```

file-list of current directory moved to another directory



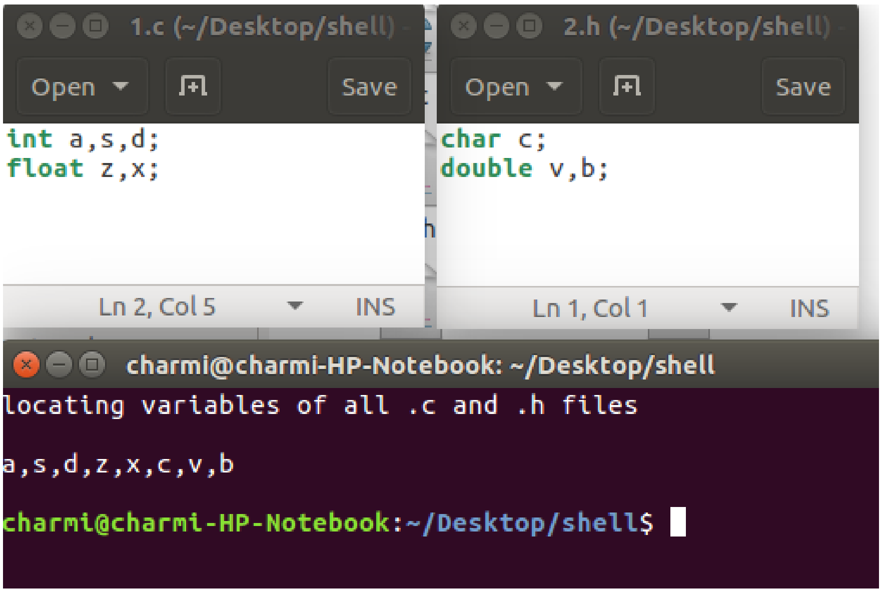
Contains of another directory

Q.32: Locate variable in C source files (i.e. with .c or .h extensions)

Code:

```
clear
echo locating variables of all .c and .h files
echo
grep -e int -e float -e double -e char *.{c,h} |
awk '{first = $1; $1 = ""; print substr($2,1,length($2)-1)}' |
tr '\n' ',' |
sed 's/.$//'
echo
```

We can also do this in gdb using commands: info locals, info variables, bt full



```
int a,s,d;
float z,x;
```

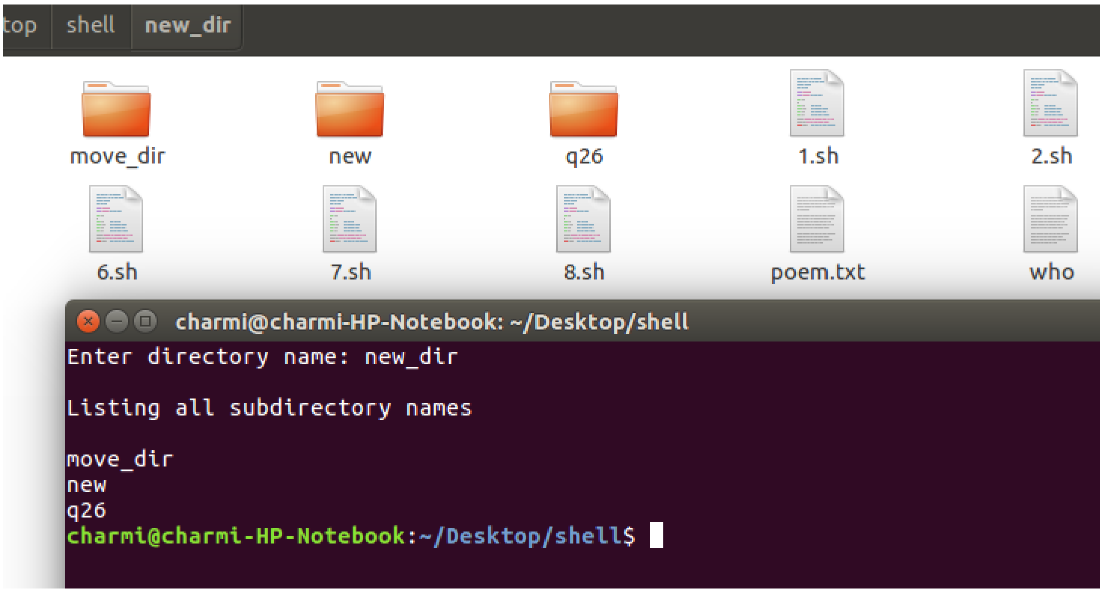
```
char c;
double v,b;
```

```
locating variables of all .c and .h files

a,s,d,z,x,c,v,b

charmi@charmi-HP-Notebook:~/Desktop/shell$
```

Q.33: List all subdirectory names

Code:

```
clear
echo -n "Enter directory name: "
read d
echo
echo Listing all subdirectory names
echo

ls -l $d | tail -n +2 | awk '{print $1, $9}' > prg_33.txt
while read line;
do
   case $line in
   d*)
      echo $line | cut -d ' ' -f 2
   esac
done < prg_33.txt
```



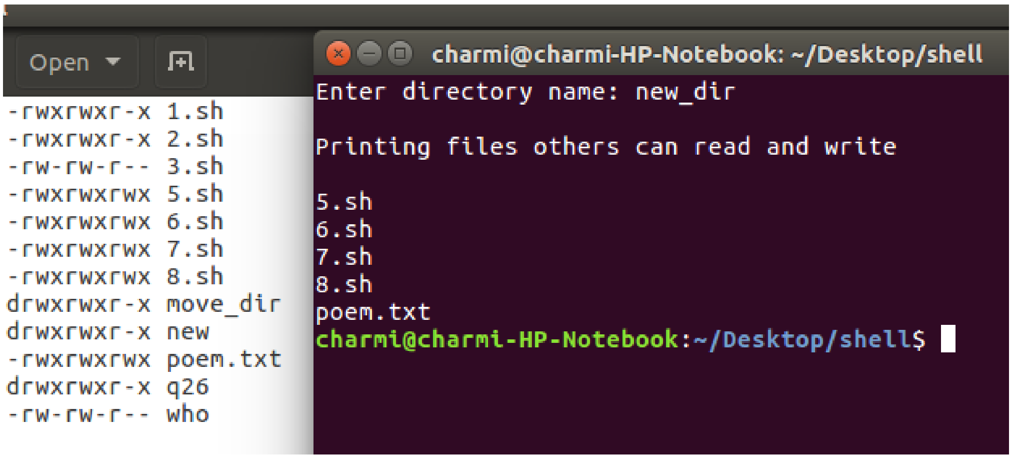
```
Enter directory name: new_dir

Listing all subdirectory names

move_dir
new
q26
charmi@charmi-HP-Notebook:~/Desktop/shell$
```

Q.34: List files others can read and write

Code:

```
clear
echo -n "Enter directory name: "
read d
echo
echo Printing files others can read and write
echo

ls -l $d | tail -n +2 | awk '{print $1 , $9}' > prg_34.txt
while read line;
```

```
do
    other=`echo $line | cut -c8-10`

   case $other in
   rw*)
      echo $line | cut -d ' ' -f 2
   esac

done < prg_34.txt
```



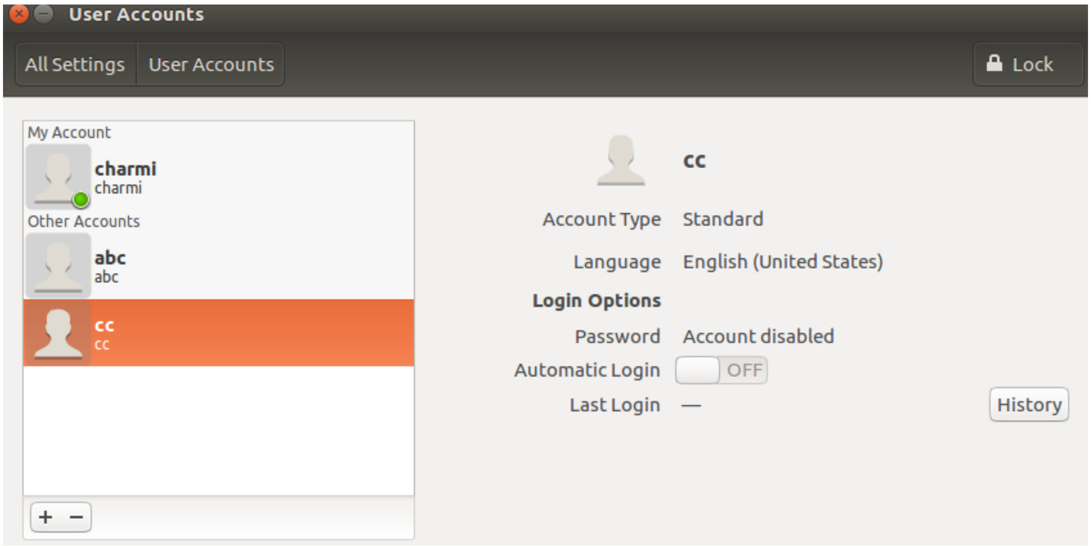
```
-rwxrwxr-x 1.sh
-rwxrwxr-x 2.sh
-rw-rw-r-- 3.sh
-rwxrwxrwx 5.sh
-rwxrwxrwx 6.sh
-rwxrwxrwx 7.sh
-rwxrwxrwx 8.sh
drwxrwxr-x move_dir
drwxrwxr-x new
-rwxrwxrwx poem.txt
drwxrwxr-x q26
-rw-rw-r-- who
```

```
Enter directory name: new_dir

Printing files others can read and write

5.sh
6.sh
7.sh
8.sh
poem.txt
charmi@charmi-HP-Notebook:~/Desktop/shell$
```

Q.35: List users without passwords

Code:

```
clear

awk -F'[/:]' '{if ($3 >= 1000 && $3 != 65534) print $1}' /etc/passwd > pass.txt
echo Printing all users of the system
echo
cat pass.txt

echo
echo Printing usernames without password
echo

set `awk -F'[/:]' '{if ($3 >= 1000 && $3 != 65534) print $1}' /etc/passwd | sort`
while [ -n "$1" ]
do
        sudo awk -F'[/:]' '{if ($2!="*" && !($2!="!")) print $1}' /etc/shadow | grep -w $1
        shift 1
done
echo
```



My Account
charmi
charmi

Other Accounts
abc
abc
cc
cc

cc

Account Type Standard

Language English (United States)

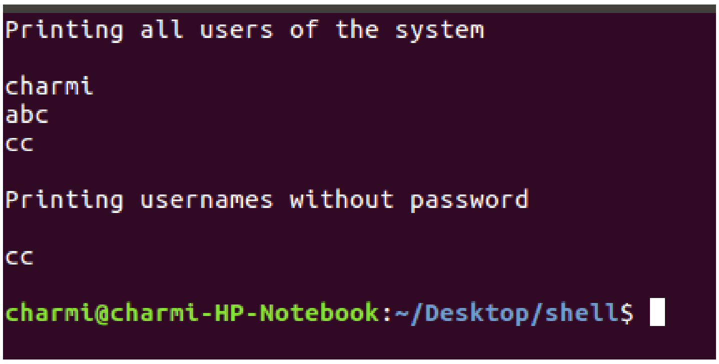
Login Options

Password Account disabled

Automatic Login OFF

Last Login —

user: cc does not have password



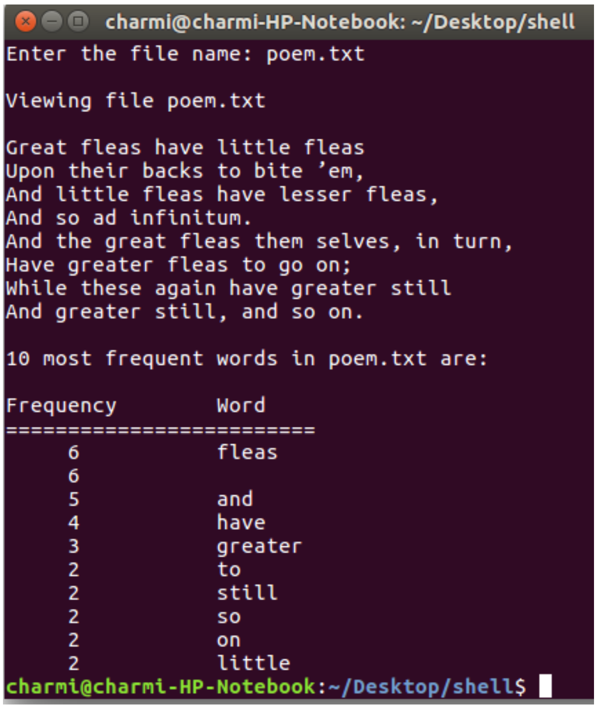
```
Printing all users of the system

charmi
abc
cc

Printing usernames without password

cc

charmi@charmi-HP-Notebook:~/Desktop/shell$
```

Q.36: Using filters (pipes etc.) print 10 most frequent words in its input

Code:

```
clear

#function to calculate the the frequency of words
wordfrequency() {
  awk '
     BEGIN { FS="[^a-zA-Z]+" } {
         for (i=1; i<=NF; i++) {
             word = tolower($i)
             words[word]++
         }
     }
     END {
         for (w in words)
              printf("   %3d \t\t %s\n", words[w], w)
     } ' | sort -rn
}
```

```
echo -n 'Enter the file name: '
read filename
echo
echo Viewing file $filename
echo
cat $filename
echo
echo '10 most frequent words in '$filename' are: '
echo
echo -e "Frequency \t Word"
echo "========================"
cat $filename | wordfrequency | head -10
```



```
Enter the file name: poem.txt

Viewing file poem.txt

Great fleas have little fleas
Upon their backs to bite ’em,
And little fleas have lesser fleas,
And so ad infinitum.
And the great fleas them selves, in turn,
Have greater fleas to go on;
While these again have greater still
And greater still, and so on.

10 most frequent words in poem.txt are:

Frequency        Word
=========================
      6          fleas
      6
      5          and
      4          have
      3          greater
      2          to
      2          still
      2          so
      2          on
      2          little
charmi@charmi-HP-Notebook:~/Desktop/shell$
```

Q.37: List all files in a directory that are

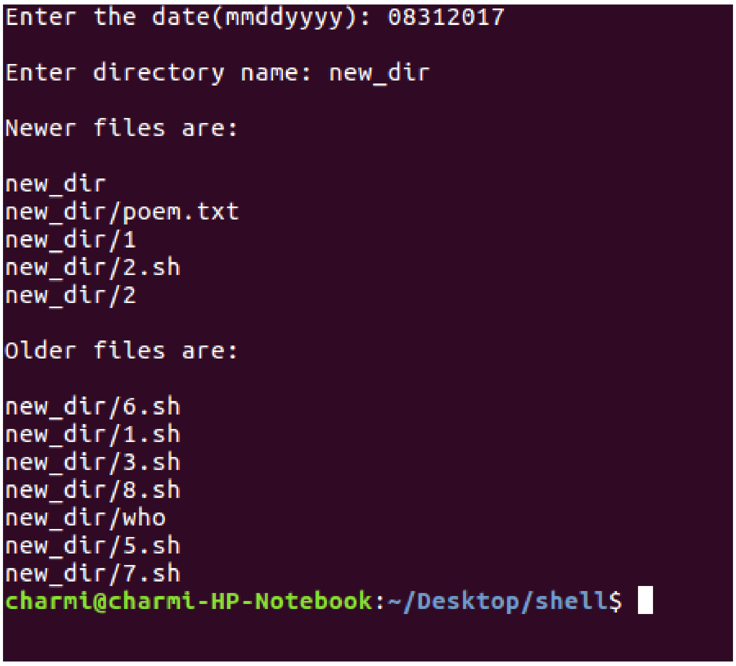
a. Newer than a specified date

b. Older than a specified date

Code:

```
clear
echo -n "Enter the date(mmddyyyy): "
read userDate
echo

echo -n "Enter directory name: "
read d

dir="/home/charmi/Desktop/shell/date"
echo

echo "Newer files are: "
```

```
echo
touch -t $userDate $dir
find $d -newer $dir
echo
echo "Older files are: "
echo
touch -t $userDate $dir
find $d -not -newer $dir
```



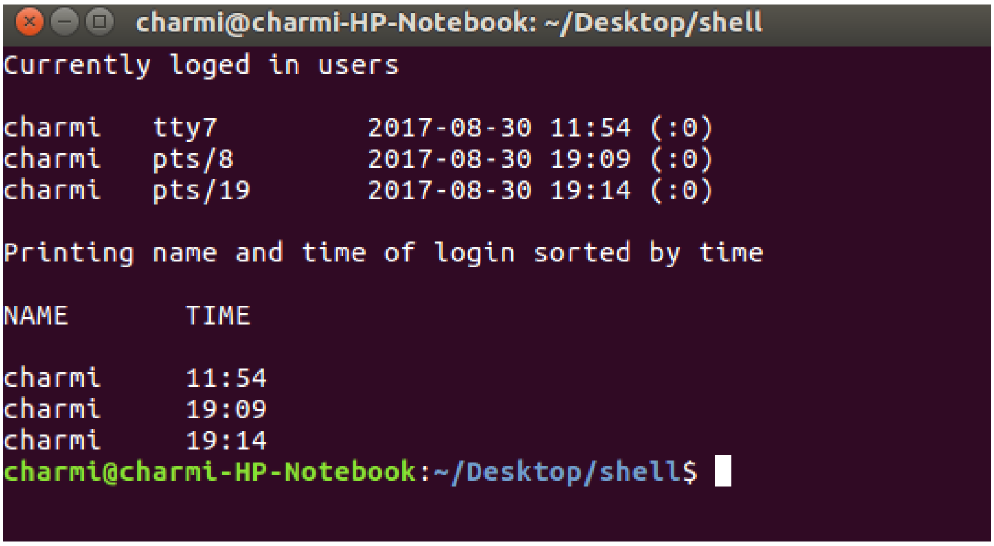
```
Enter the date(mmddyyyy): 08312017

Enter directory name: new_dir

Newer files are:

new_dir
new_dir/poem.txt
new_dir/1
new_dir/2.sh
new_dir/2

Older files are:

new_dir/6.sh
new_dir/1.sh
new_dir/3.sh
new_dir/8.sh
new_dir/who
new_dir/5.sh
new_dir/7.sh
charmi@charmi-HP-Notebook:~/Desktop/shell$
```

awk Commands:

a. Print name and time of login sorted by time

Code:

```
clear
echo Currently loged in users
echo
who
echo
echo Printing name and time of login sorted by time
echo
echo -e "NAME \t   TIME"
echo
who | awk '{print $1 , "   ", $4}' | sort -nk4
```

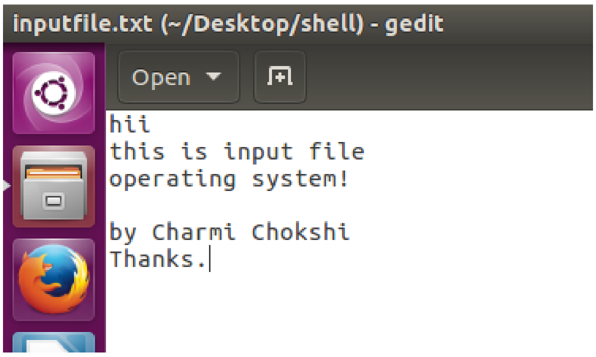


```
Currently loged in users

charmi   tty7         2017-08-30 11:54 (:0)
charmi   pts/8        2017-08-30 19:09 (:0)
charmi   pts/19       2017-08-30 19:14 (:0)

Printing name and time of login sorted by time

NAME       TIME

charmi     11:54
charmi     19:09
charmi     19:14
charmi@charmi-HP-Notebook:~/Desktop/shell$
```

b. Add line numbers to an input stream

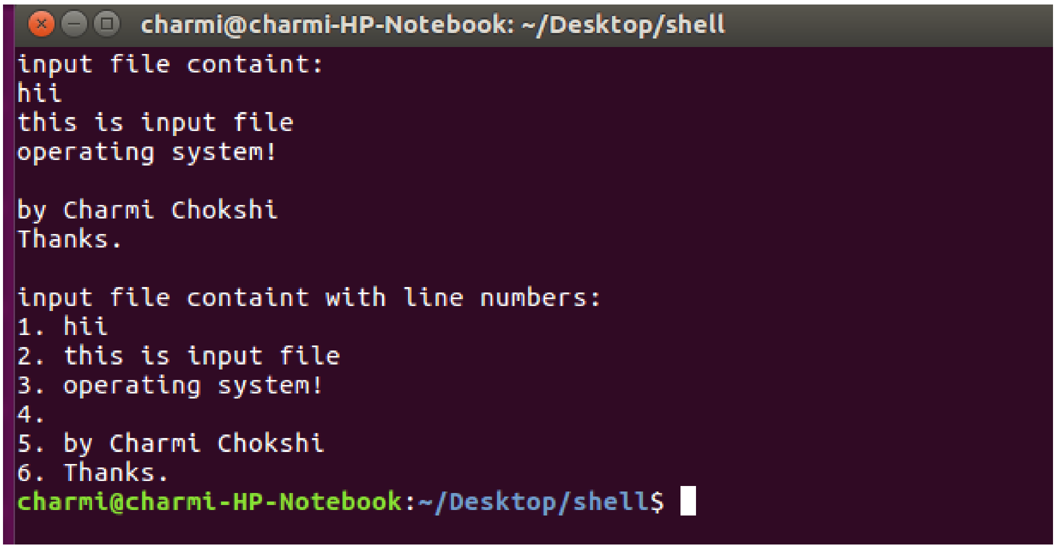
Code:

```
clear
echo input file containt:
cat inputfile.txt
echo
echo input file containt with line numbers:
awk '{print NR  ". " $s}' inputfile.txt
```



```
hii
this is input file
operating system!

by Charmi Chokshi
Thanks.
```

input stream



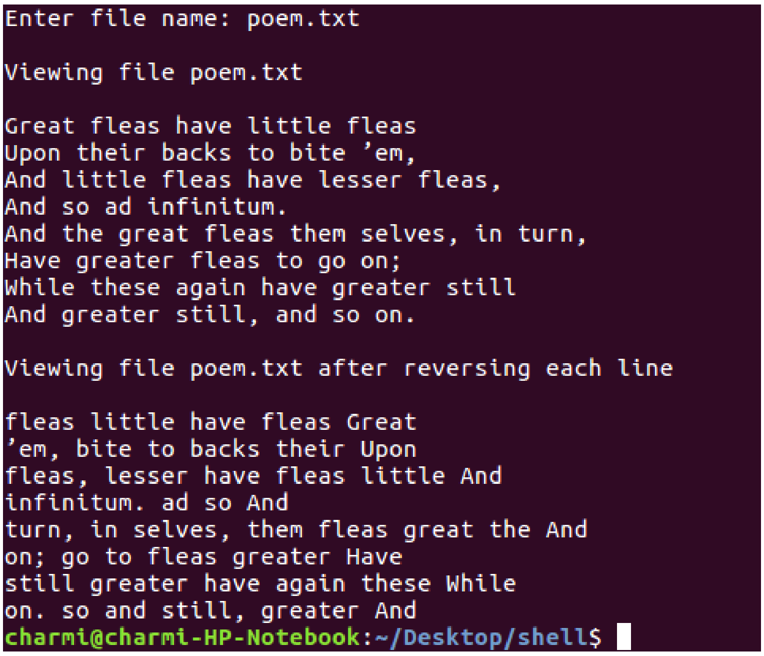
```
input file containt:
hii
this is input file
operating system!

by Charmi Chokshi
Thanks.

input file containt with line numbers:
1. hii
2. this is input file
3. operating system!
4.
5. by Charmi Chokshi
6. Thanks.
charmi@charmi-HP-Notebook:~/Desktop/shell$
```

output with line numbers using awk command

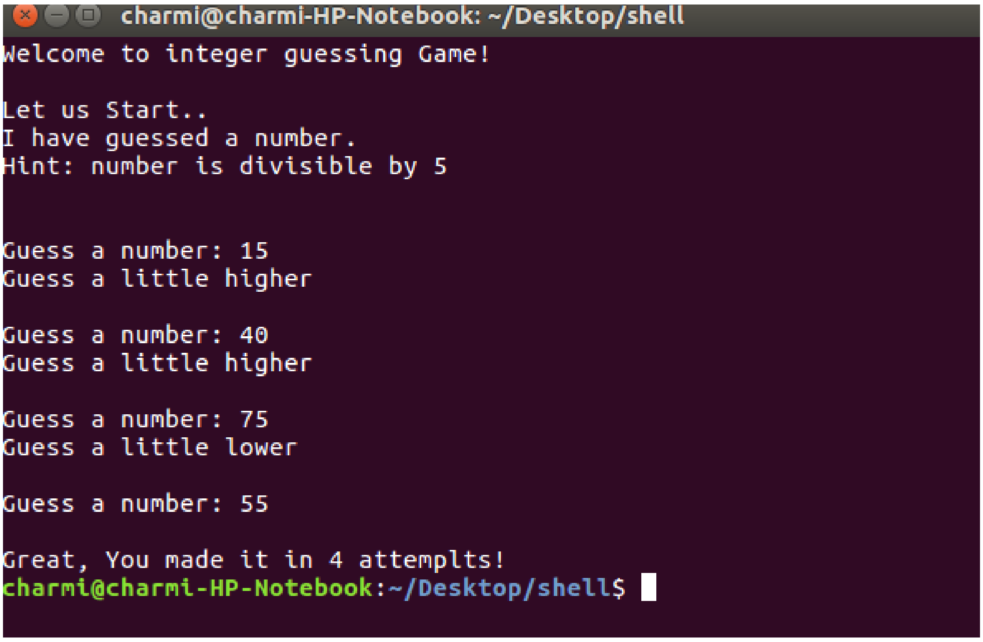
c. Collect each line of input in a separate array element then prints them out in reverse order.

Code:

```
clear
echo -n "Enter file name: "
read fn

echo
echo Viewing file $fn
echo
cat $fn
echo
echo Viewing file $fn after reversing each line
echo
awk '{for(i=NF; i>=1; i--) printf "%s ", $i; print ""}' $fn
```



```
Enter file name: poem.txt

Viewing file poem.txt

Great fleas have little fleas
Upon their backs to bite ’em,
And little fleas have lesser fleas,
And so ad infinitum.
And the great fleas them selves, in turn,
Have greater fleas to go on;
While these again have greater still
And greater still, and so on.

Viewing file poem.txt after reversing each line

fleas little have fleas Great
’em, bite to backs their Upon
fleas, lesser have fleas little And
infinitum. ad so And
turn, in selves, them fleas great the And
on; go to fleas greater Have
still greater have again these While
on. so and still, greater And
charmi@charmi-HP-Notebook:~/Desktop/shell$
```



```
Welcome to integer guessing Game!

Let us Start..
I have guessed a number.
Hint: number is divisible by 5


Guess a number: 15
Guess a little higher

Guess a number: 40
Guess a little higher

Guess a number: 75
Guess a little lower

Guess a number: 55

Great, You made it in 4 attemplts!
charmi@charmi-HP-Notebook:~/Desktop/shell$
```