INDEX

PROGRAM 1.1

1a.Write a shell script that takes a valid directory name as an argument recursively descend all the sub-directors, find the maximum length of any file in that hierarchy, and write the maximum value to the standard output.

AIM

Script to take a valid directory name as an argument recursively descend all the sub-directors, find the maximum length of any file in that hierarchy, and write the maximum value to the standard output.

SOURCE CODE

```
#!/bin/sh
for i in $*
do
if [ -d $i ]
then
echo " large file size is "
else
echo " not a directory "
fi
done
echo `ls -Rl $1 | grep "^-" | tr -s ' ' | cut -d ' ' -f 5,9 | sort -n |
tail -1`
```

OUTPUT

```
nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt1a.sh
717 file_properties.sh
nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt1a.sh sam
 large file size is
223 output.sh
nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt1a.sh nishanth
 not a directory
```

PROGRAM 1.2

1b.Write a shell script that accepts a path name and creates all the components in that path name as directories. For example, if the script is named as mpc, then the command mpc a/b/c/d should create subdirectories a, a/b, a/b/c, a/b/c/d.

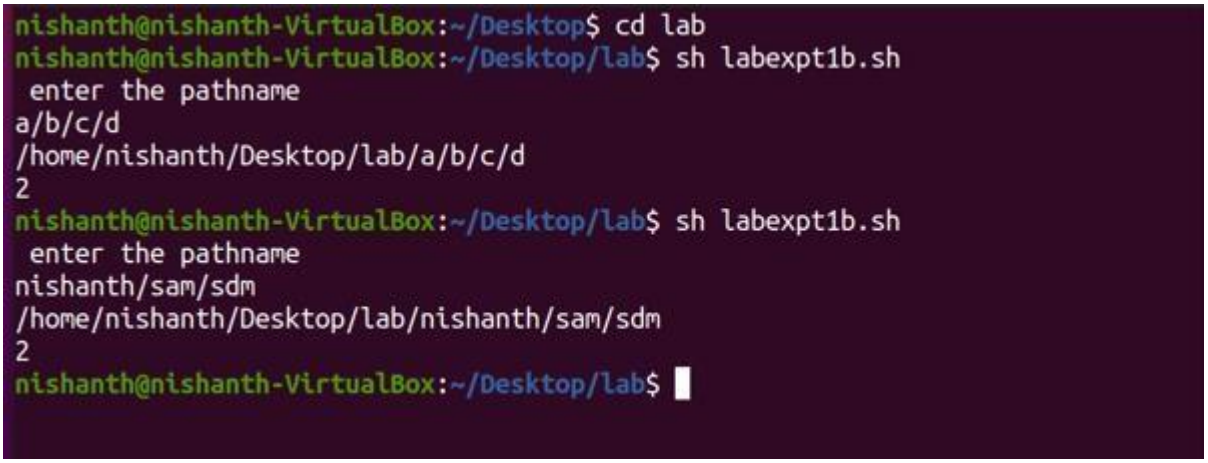
AIM

Shell script to accepts a path name and creates all the components in that path name as directories. For example, if the script is named as mpc, then the command mpc a/b/c/d should create subdirectories a, a/b, a/b/c, a/b/c/d.

SOURCE CODE

```
#!/bin/bash
echo " enter the pathname"
read p
i=1
j=1
len=`
echo $p | wc -c`
while [ $i -le $len ]
do
x=`echo $p | cut -d ' ' -f $j`
namelength=`echo $x | wc -c`
mkdir -p $x
cd $x
pwd
j=`expr $j + 1`
i=`expr $i + $namelength`
done
```

OUTPUT


nishanth@nishanth-VirtualBox:~/Desktop$ cd lab
nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt1b.sh
enter the pathname
a/b/c/d
/home/nishanth/Desktop/lab/a/b/c/d
2
nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt1b.sh
enter the pathname
nishanth/sam/sdm
/home/nishanth/Desktop/lab/nishanth/sam/sdm
2
nishanth@nishanth-VirtualBox:~/Desktop/lab$

PROGRAM 2.1

2a.Write a shell script that accepts two filenames as arguments, checks if the permissions for these files are identical and if the permissions are identical, output common permissions otherwise output each filename followed by its permissions.

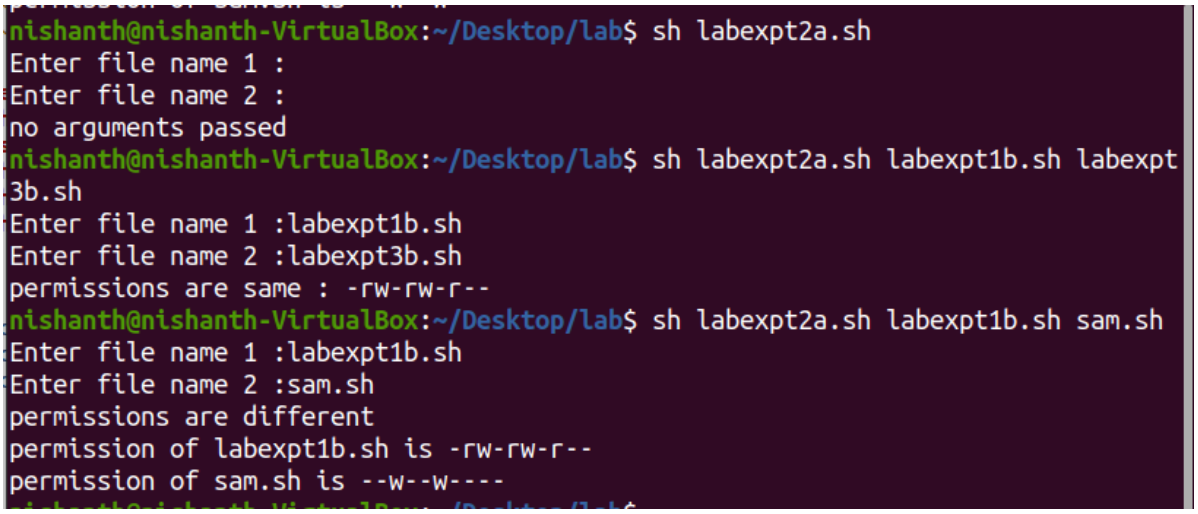
AIM

Shell script to accepts two filenames as arguments, checks if the permissions for these files are identical and if the permissions are identical, output common permissions otherwise output each filename followed by its permissions.

SOURCE CODE

```
#!/bin/sh
echo "Enter file name 1 :$1"
echo "Enter file name 2 :$2"
if [ $# -eq 0 ]
then
echo "no arguments passed"
elif [ ! -e $1 -o ! -e $2 ]
then
echo "file does not exist"
else
x=`ls -l $1 | cut -c 1-10`
y=`ls -l $2 | cut -c 1-10`
if [ $x = $y ]
then
echo "permissions are same : $x"
else
echo "permissions are different"
echo "permission of $1 is $x"
echo "permission of $2 is $y"
fi
fi
```

OUTPUT


nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt2a.sh
Enter file name 1 :
Enter file name 2 :
no arguments passed
nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt2a.sh labexpt1b.sh labexpt
3b.sh
Enter file name 1 :labexpt1b.sh
Enter file name 2 :labexpt3b.sh
permissions are same : -rw-rw-r--
nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt2a.sh labexpt1b.sh sam.sh
Enter file name 1 :labexpt1b.sh
Enter file name 2 :sam.sh
permissions are different
permission of labexpt1b.sh is -rw-rw-r--
permission of sam.sh is --w--w----

PROGRAM 2.2

2b.Write a shell script which accepts valid log-in names as arguments and prints their corresponding home directories, if no arguments are specified, print a suitable error message.

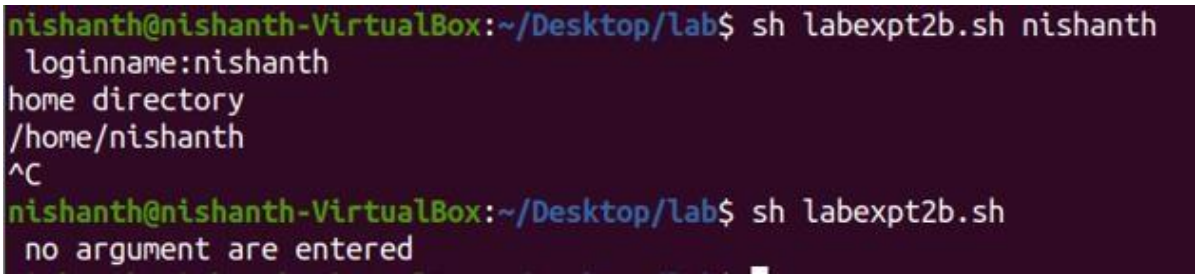
AIM

Shell script to accepts valid log-in names as arguments and prints their corresponding home directories, if no arguments are specified, print a suitable error message.

SOURCE CODE

```
#!/bin/sh
echo "enter arguments : $1";
if [ $# -eq 0 ]
then
echo "No command line argument passed"
exit
fi
while [ $1 ]
do
cat /etc/passwd | cut -d ":" -f1 | grep "^$1" > temp
a=`cat temp`
if [ "$a" != "$1" ]
then
echo "ERROR:$1 is an invalid login name"
else
echo "Home Directory for $1 is"
echo `cat /etc/passwd | grep "^$1" | cut -d ":" -f6`
fi
shift
done
```

OUTPUT


nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt2b.sh nishanth
loginname:nishanth
home directory
/home/nishanth
^C
nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt2b.sh
no argument are entered

PROGRAM 3.1

3a.Create a script file called file properties that reads a filename entered and outputs its properties.

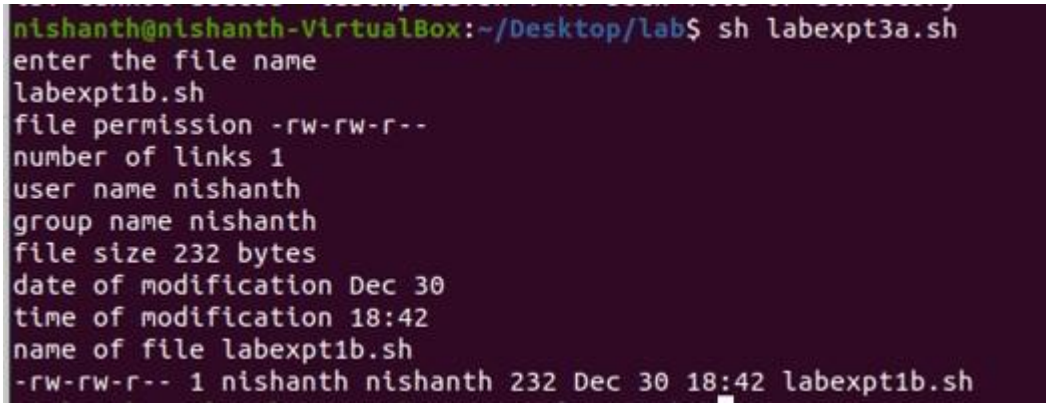
AIM

Shell script to reads a filename entered and outputs its properties.

SOURCE CODE

```
#!/bin/bash
echo "enter the file name"
read file
if [ -f $file ]
then
set -- `ls -l $file`
echo "file permission $1"
echo "number of links $2"
echo "user name $3"
echo "group name $4"
echo "file size $5 bytes"
echo "date of modification $6 $7"
echo "time of modification $8"
echo "name of file $9"
else
echo "file doesnot exit"
fi
ls -l $file
```

OUTPUT


nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt3a.sh
enter the file name
labexpt1b.sh
file permission -rw-rw-r--
number of links 1
user name nishanth
group name nishanth
file size 232 bytes
date of modification Dec 30
time of modification 18:42
name of file labexpt1b.sh
-rw-rw-r-- 1 nishanth nishanth 232 Dec 30 18:42 labexpt1b.sh

PROGRAM 3.2

3b.Write a shell script to implement terminal locking (Similar to the lock command). It should prompt for the user for a password. After accepting the password entered by the user, it must prompt again for the matching password as confirmation and if match occurs, it must lock the keyword until a matching password is entered again by the user.

AIM

Shell script to implement terminal locking (Similar to the lock command). It should prompt for the user for a password. After accepting the password entered by the user, it must prompt again for the matching password as confirmation and if match occurs, it must lock the keyword until a matching password is entered again by the user.

SOURCE CODE

```
#!/bin/bash
stty -echo
clear
echo "Enter the password"
read pass1
echo "Re-Enter the password"
read pass2
val=1
while [ $val -eq 1 ]
do
if [ $pass1 = $pass2 ]
then
val=0
echo "Terminal locked"
echo "To unlock enter the password"
pass1=""
until [ "$pass1" = "$pass2" ]
do
read pass1
```

```
echo "Terminal locked
done
echo "Terminal unlocked"
stty echo
else
echo "Password mismatch please re-type it"
stty -echo
read pass2

fi
done
```

OUTPUT

PROGRAM 4.1

4a.Write a shell script that accept one or more file names as argument and convert all of them to uppercase, provided they exists in current directory.

AIM

Shell script to accept one or more file names as argument and convert all of them to uppercase, provided they exists in current directory.

SOURCE CODE

```
#!/bin/bash
echo "Enter the file name"
read file
if [ -z $file ]
then
echo "no arguments passed"
elif [  ! -f $file ]
then
echo "file does not exist"
else
tr '[a-z]' '[A-Z]' < $file
fi
```

OUTPUT

```
nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt4a.sh
Enter the file name

no arguments passed
nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt4a.sh
Enter the file name
output.sh
#!/BIN/BASH
ECHO "ENTER THE FILE NAME"
READ FILE
IF [ -Z $FILE ]
THEN
ECHO "NO ARGUMENTS PASSED"
ELIF [  ! -F $FILE ]
THEN
ECHO "FILE DOES NOT EXIST"
ELSE
F=`LS -L $FILE | CUT -D  " " -F8`
ECHO "FILE CREATION TIME :" $F
FI
```

PROGRAM 4.2

4b.Write a shell script that displays all the links to a file specified as the first argument to the script. The second argument, which is optional, can be used to specify in which the search is to begin. If this second argument is not present, the search is to begin in the current working directory. In either case, the starting directory as well as its subdirectories at all levels must be searched. The script need not include error checking.

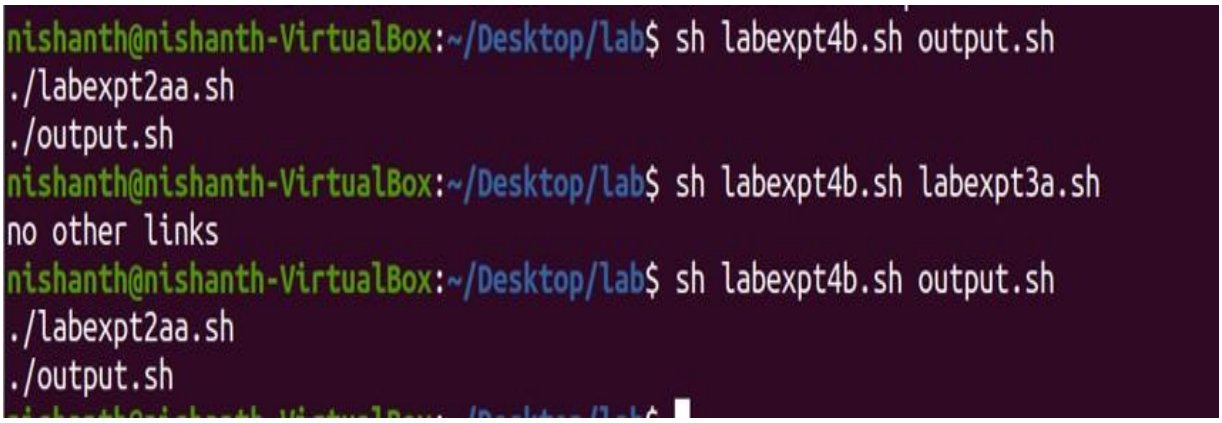
AIM

Shell script to displays all the links to a file specified as the first argument to the script. The second argument, which is optional, can be used to specify in which the search is to begin. If this second argument is not present, the search is to begin in the current working directory. In either case, the starting directory as well as its subdirectories at all levels must be searched. The script need not include error checking.

SOURCE CODE

```
#!/bin/bash
file=$1
set -- `ls -l $file`
lcnt=$2
if [ $lcnt -eq 1 ]
then
 echo "no other links"
 exit
else
 set -- `ls -i $file`
 inode=$1
 find "." -xdev -inum $inode -print
fi
```

OUTPUT


nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt4b.sh output.sh
./labexpt2aa.sh
./output.sh
nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt4b.sh labexpt3a.sh
no other links
nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt4b.sh output.sh
./labexpt2aa.sh
./output.sh

PROGRAM 5.1

5a.Write a shell script that accepts filename as argument and display its creation time if file exist and if does not send output error message.

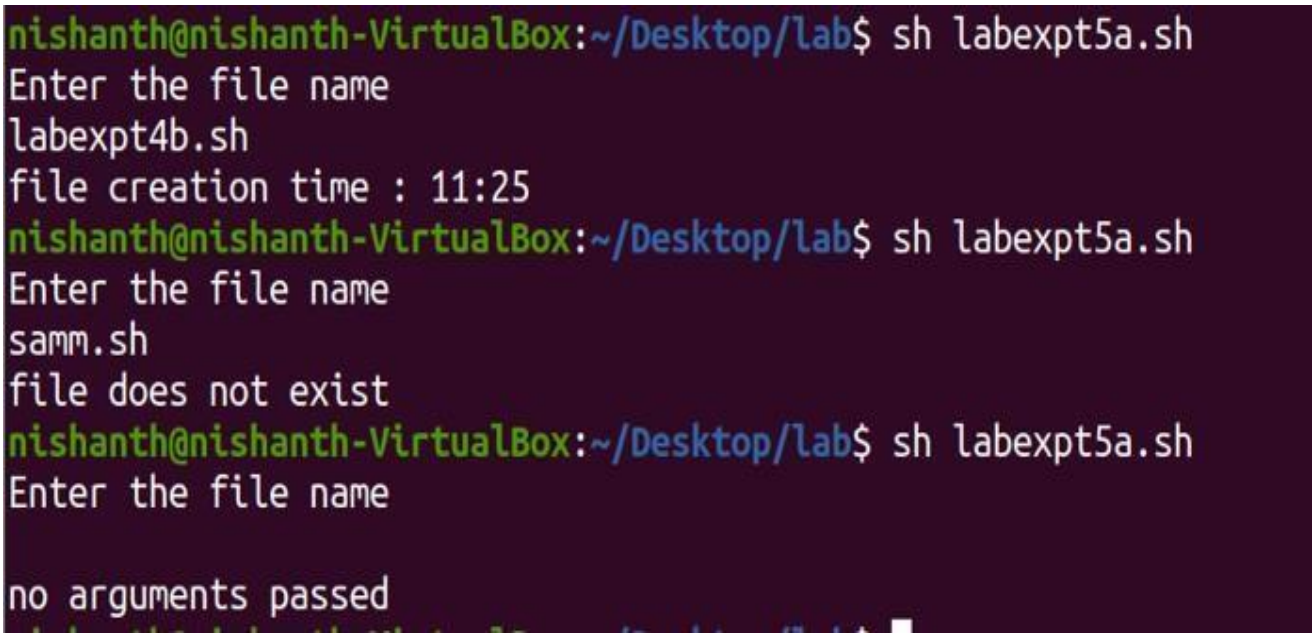
AIM

Shell script to accepts filename as argument and display its creation time if file exist and if does not send output error message.

SOURCE CODE

```
#!/bin/bash
echo "Enter the file name"
read file
if [ -z $file ]
then
echo "no arguments passed"
elif [  ! -f $file ]
then
echo "file does not exist"
else
f=`ls -l $file | cut -d " " -f8`
echo "file creation time :" $f
fi
```

OUTPUT


nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt5a.sh
Enter the file name
labexpt4b.sh
file creation time : 11:25
nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt5a.sh
Enter the file name
samm.sh
file does not exist
nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt5a.sh
Enter the file name
no arguments passed

PROGRAM 5.2

5b.Write a shell script to display the calendar for the current month with current date replaced by * or ** depending whether the date is one digit or two digit.

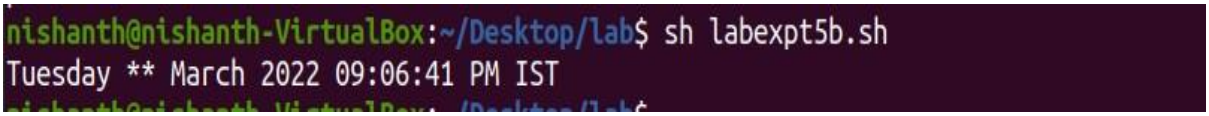
AIM

Shell script to display the calendar for the current month with current date replaced by * or ** depending whether the date is one digit or two digit.

SOURCE CODE

```
#!/bin/bash
set `date`
y=$2
if [ $y -le 9 ]
then
date |sed "s/$2/*/"
else
date |sed "s/$2/**/"
fi
```

OUTPUT



PROGRAM 6.1

6a.Write a shell script to find a file/s that matches a pattern given as command line argument in the home directory, display the contents of the file and copy the file into the directory ~/mydir.

AIM

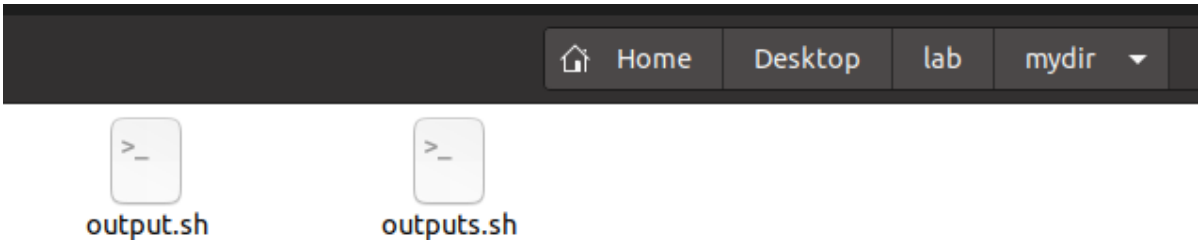
Shell script to find a file/s that matches a pattern given as command line argument in the home directory, display the contents of the file and copy the file into the directory ~/mydir.

SOURCE CODE

```
#!/bin/sh
echo "Enter the file name"
read file
if [ -z $file ]
then
echo "no arguments passed"
elif [  ! -f $file ]
then
echo "file does not exist"
else
#ls $file
cat $file
cp -f $file /home/nishanth/Desktop/lab/mydir
fi
```

OUTPUT

```
nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt6a.sh
Enter the file name

no arguments passed
nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt6a.sh
Enter the file name
lam.sh
file does not exist
nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt6a.sh
Enter the file name
outputs.sh
outputs.sh
#!/bin/bash
echo "Enter the file name"
read file
echo $file
fi
```



PROGRAM 6.2

6b.Write a shell script to list all the files in a directory whose filename is at least 10 characters. (Use expr command to check the length).

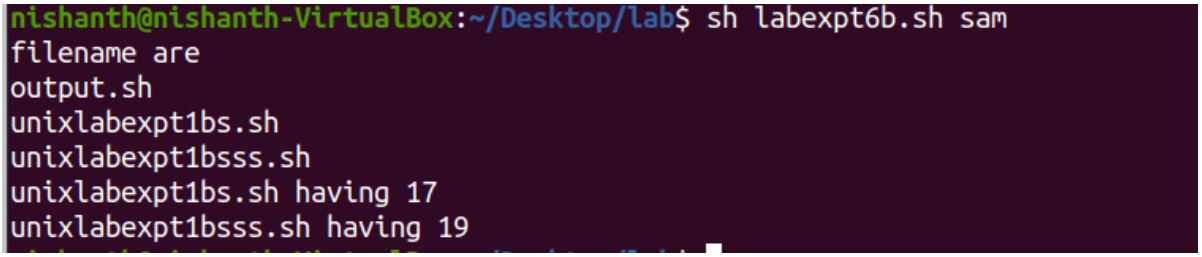
AIM

Shell script to list all the files in a directory whose filename is at least 10 characters. (Use expr command to check the length).

SOURCE CODE

```
#!/bin/bash
if [ $# -eq 0 ]
then
echo "no argument"
else
c=`ls  $1`
echo "filename are\n$c"
for i in $c
do
len=`expr length $i`
if [ $len -ge 10 ]
then
echo "$i having $len"
fi
done
fi
```

OUTPUT


nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt6b.sh sam
filename are
output.sh
unixlabexpt1bs.sh
unixlabexpt1bsss.sh
unixlabexpt1bs.sh having 17
unixlabexpt1bsss.sh having 19

PROGRAM 7.1

7a.Write a shell script that gets executed and displays the message either "Good Morning" or "Good Afternoon" or "Good Evening" depending upon time at which the user logs in.

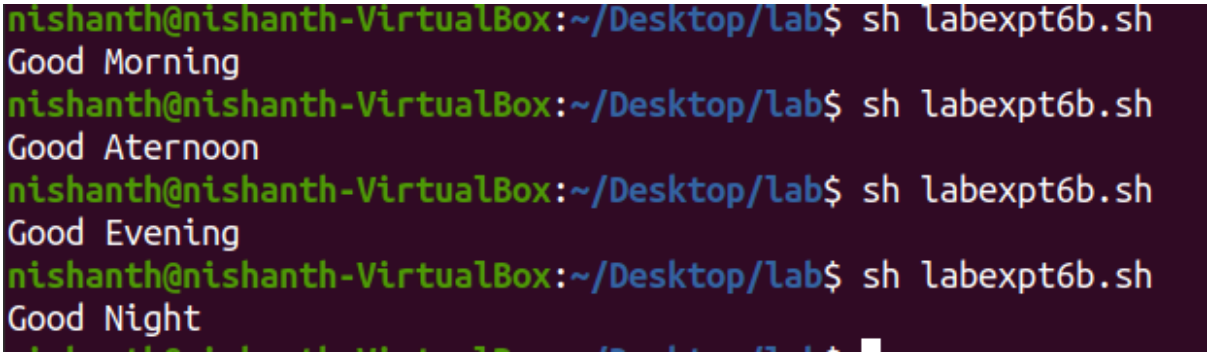
AIM

Shell script that gets executed and displays the message either "Good Morning" or "Good Afternoon" or "Good Evening" depending upon time at which the user logs in.

SOURCE CODE

```
#!/bin/bash
t=`date +%H`    #%H: Prints the hour.
if [ $t -gt 0 -a $t -le 12 ]
then
echo "Good  Morning  "
elif [ $t -gt 12 -a $t -le 14 ]
then
echo "Good Aternoon "
elif [ $t -gt 14 -a $t -le 18 ]
then
echo "Good Evening "
else
echo "Good Night"
fi
```

OUTPUT


nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt6b.sh
Good Morning
nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt6b.sh
Good Aternoon
nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt6b.sh
Good Evening
nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt6b.sh
Good Night

PROGRAM 7.2

7b.Write a shell script that accepts a list of filenames as its argument, count and report occurrence of each word that is present in the first argument file on other argument files

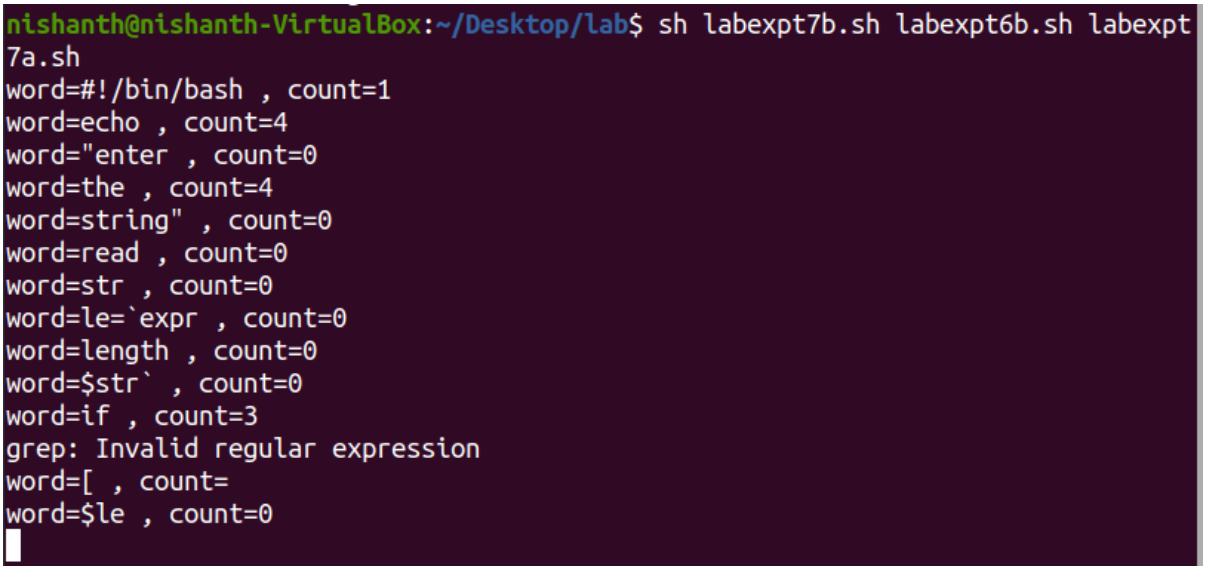
AIM

Shell script to accepts a list of filenames as its argument, count and report occurrence of each word that is present in the first argument file on other argument files

SOURCE CODE

```
#!/bin/sh
if [ $# -ne 2 ]
then
echo "Error :invalid no of arguments "
exit
fi
str=`cat $1 | tr '\n' ' '`
for a in $str
do
echo "word=$a , count=`grep -c "$a" $2` "
done
```

OUTPUT


nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt7b.sh labexpt6b.sh labexpt
7a.sh
word=#!/bin/bash , count=1
word=echo , count=4
word="enter , count=0
word=the , count=4
word=string" , count=0
word=read , count=0
word=str , count=0
word=le=`expr , count=0
word=length , count=0
word=$str` , count=0
word=if , count=3
grep: Invalid regular expression
word=[, count=
word=$le , count=0

PROGRAM 8.1

8a.Write a shell script that determine the period for which as specified user is working on a system and display appropriate message.

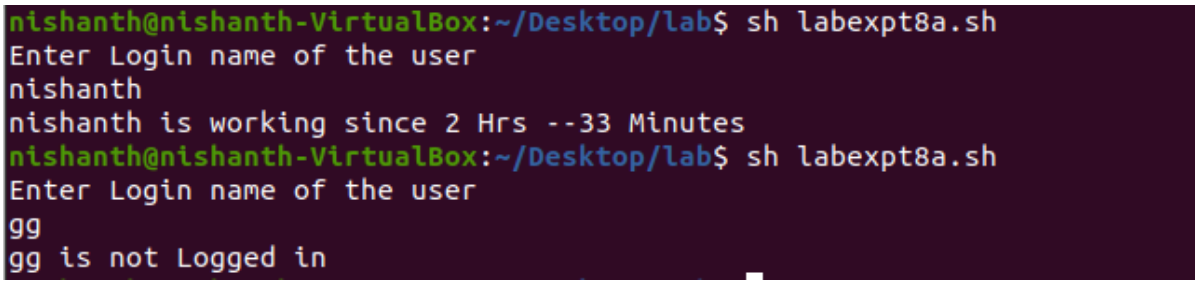
AIM

Shell script to determine the period for which as specified user is working on a system and display appropriate message

SOURCE CODE

```
#!/bin/bash
echo "Enter Login name of the user"
read name
userinfo=`who | grep -w "$name"` # |grep"pts"`
if [ $? -ne 0 ]
then
echo "$name is not Logged in"
 exit
fi
loginhours=`echo "$userinfo" | tr -s " " | cut -c 24-25`
loginminuts=`echo "$userinfo" | tr -s " " | cut -c 27-28`
minnow=`date +%M`
hournow=`date +%H`
th=`expr $hournow - $loginhours`
tm=`expr $minnow - $loginminuts`
echo "$name is working since $th Hrs -$tm Minutes"
```

OUTPUT



PROGRAM 8.2

8b.Write a shell script that reports the logging on of as specified user within one minute after he/she login. The script automatically terminates if specified user does not login during specified in period of time.

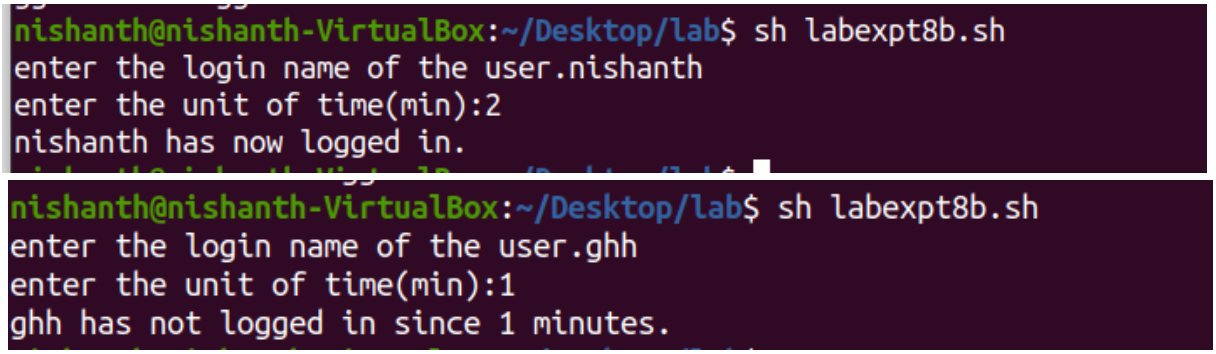
AIM

Shell script to reports the logging on of as specified user within one minute after he/she login. The script automatically terminates if specified user does not login during specified in period of time.

SOURCE CODE

```
echo -n "Enter the login name of the user: "
read lname

period=0

echo -n "Enter the unit of time (min): "
read min

until who | grep -w "$lname" > /dev/null
do
  sleep 10
  # (sleep command is used to create a dummy job. A dummy job helps in delaying the execution.)
  period=`expr $period + 1`

  if [ $period -ge $min ]
  then
    echo "$lname has not logged in within the last $min minutes."
    exit
  fi
done

echo "$lname has now logged in.
```

OUTPUT


nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt8b.sh
enter the login name of the user.nishanth
enter the unit of time(min):2
nishanth has now logged in.
nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt8b.sh
enter the login name of the user.ghh
enter the unit of time(min):1
ghh has not logged in since 1 minutes.

PROGRAM 9.1

9a.Write a shell script that accepts the filename, starting and ending line number as an argument and display all the lines between the given line number.

AIM

Shell script to accepts the filename, starting and ending line number as an argument and display all the lines between the given line number.

SOURCE CODE

```
#!/bin/sh
echo "Enter file name"
read file
echo "Enter starting line"
read a
echo "Enter ending line"
read b
if [ -z $file ] || [ -z $a ] || [ -z $b ]
then
echo "no arguments"
elif [ ! -f $file ]
then
echo "File does not exist"
elif [ $a -eq 0 ]|| [ $b -eq 0 ] || [ $a -gt $b ]
then
echo "Invalid input"
else
sed -n "$a,$b p" $file
fi
```

OUTPUT

```
nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt9a.sh
Enter file name
labexpt8a.sh
Enter starting line
1
Enter ending line
8
echo "Enter Login name of the user"
read name
userinfo=`who | grep -w "$name"` # |grep"pts"`
if [ $? -ne 0 ]
then
echo "$name is not Logged in"
 exit
fi
nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt9a.sh
Enter file name

Enter starting line

Enter ending line

no arguments
nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt9a.sh
Enter file name
sams.sh
Enter starting line
1
Enter ending line
1
File does not exist
```

PROGRAM 9.2

9b.Write a shell script that folds long lines into 40 columns. Thus any line that exceeds 40 characters must be broken after 40th, a "/" is to be appended as the indication of folding and processing is to be continued with the residue. The input is to be supplied through a text file created by the user.

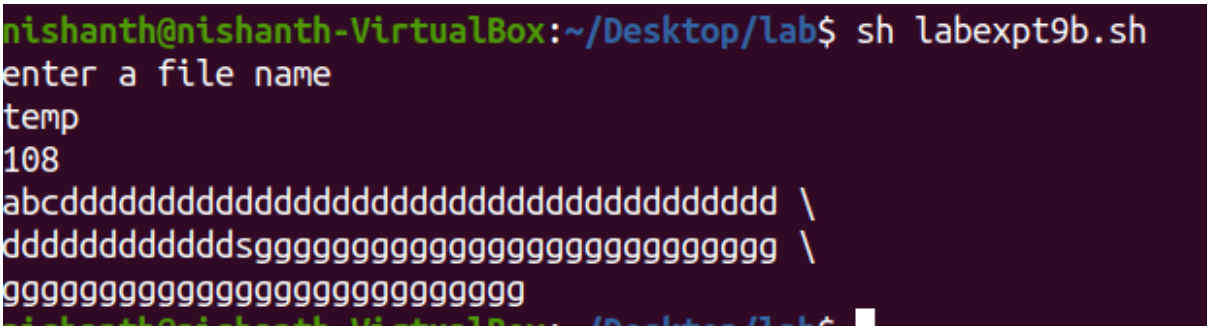
AIM

Shell script to folds long lines into 40 columns. Thus any line that exceeds 40 characters must be broken after 40th, a "/" is to be appended The input is to be supplied through a text file created by the user.

SOURCE CODE

```
echo "Enter the filename: \c"
read fn
for ln in `cat $fn`
do
lgth=`echo $ln | wc -c`
lgth=`expr $lgth - 1`
s=1
e=40
if [ $lgth -gt 40 ]
then
while [ $lgth -gt 40 ]
do
echo "`echo $ln | cut -c $s-$e` \\"
lgth=`expr $lgth - 40`
done
else
echo $ln
fi
done
```

OUTPUT


nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt9b.sh
enter a file name
temp
108
abcddddddddddddddddddddddddddddddddddd \
ddddddddddddsgggggggggggggggggggggggggg \
gggggggggggggggggggggggggg

PROGRAM 10.1

10a.Write an awk script that accepts date argument in the form of dd-mm-yy and display it in the form month, day and year. The script should check the validity of the argument and in the case of error, display a suitable message.

AIM

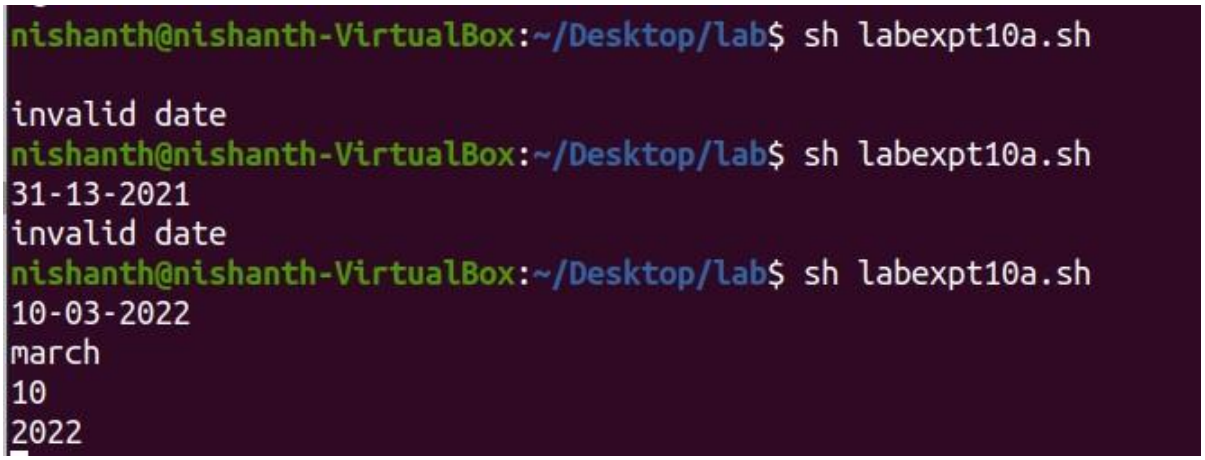
Awk script to accepts date argument in the form of dd-mm-yy and display it in the form month, day and year. The script should check the validity of the argument and in the case of error, display a suitable message.

SOURCE CODE

```
awk '{ split ($0, arr, "-")
if ((arr[2] < 1) || (arr[2] > 12) || (arr[1] < 1) || (arr[1] > 31))
 {
 print "invalid date"}
 else
 {
 if (arr[2] == 1)
 print "jan"
 if (arr[2] == 2)
 print "feb"
 if (arr[2] == 3)
 print "march"
 if (arr[2] == 4)
 print "apirl"
if (arr[2] == 5)
 print "may"
 if (arr[2] == 6)
 print "jun"
 if (arr[2] == 7)
 print "july"
 if (arr[2] == 8)
 print "aug"
 if (arr[2] == 9)
 print "sept"
 if (arr[2] == 10)
 print "oct"
 if (arr[2] == 11)
```

```
 print "nov"
 if (arr[2] == 12)
 print "dec"
print arr[1]
print arr[3]
exit 0 } }’
'
```

OUTPUT



PROGRAM 10.2

10b.Write an awk script to delete duplicated line from a text file. The order of the original lines must remain unchanged.

AIM

Awk script to delete duplicated line from a text file. The order of the original lines must remain unchanged.

SOURCE CODE

```
#!/bin/sh
echo "Enter file name"
read file
if [ -z $file ]
then
echo “no arguments”
elif [ ! -f $file ]
then
echo “files does not exist”
else
awk '!visited[$0]++' $file
fi
```

OUTPUT

```
nishanth@nishanth-VirtualBox:~/Desktop/lab$ cat output.sh
echo "nishanth"
xyz
echo "nishanth"
abcd
egf
echo "sam"
sss
echo "sam"
hij
lkm
nishanth@nishanth-VirtualBox:~/Desktop/lab$ sh labexpt10b.sh
Enter file name
output.sh
echo "nishanth"
xyz
abcd
egf
echo "sam"
sss
hij
lkm
```

PROGRAM 11.1

11a.Write an awk script to find out total number of books sold in each discipline as well as total book sold using associate array down table as given below. Electrical-34 Mechanical-67 Electrical-80 Computer Science-43 Civil-98 Mechanical-65 Computer Science-64. DA=50% of basic

AIM

Awk script to find out total number of books sold in each discipline as well as total book sold using associate array down table as given below. Electrical-34 Mechanical-67 Electrical-80 Computer Science-43 Civil-98 Mechanical-65 Computer Science-64.DA=50% of basic

SOURCE CODE

```
BEGIN {print "TOTAL NUMBER OF BOOOKS SOLD IN EACH CATAGORY"}

{ books [ $1 ] +=$2 }

END { for(item in books)

{ printf("\t%-17s %1s %-5d\n", item, "=",books[item])

total+=books[item]

}

printf("%-17s %1s %-5s\n", "total books sold","=", total)

}
```

OUTPUT

```
nishanth@nishanth-VirtualBox:~/Desktop/lab$ awk -f labexpt11a.sh data.txt
TOTAL NUMBER OF BOOOKS SOLD IN EACH CATAGORY
        Civil             = 198
        Mechanical        = 132
        Electrical        = 114
        ComputerScience   = 64
        ComputeScSience   = 43
total books sold  = 551
```

PROGRAM 11.2

11b.Write an awk script to compute gross salary of an employee accordingly to rule given below. If basic salary < 10000 then HRA=15% of basic & DA=45% of basic. If basic salary is >=1000 then HRA=20% of basic & DA=50% of basic.

AIM

Awk script to compute gross salary of an employee accordingly to rule given below. If basic salary < 10000 then HRA=15% of basic & DA=45% of basic. If basic salary is >=1000 then HRA=20% of basic & DA=50% of basic.

SOURCE CODE

```
BEGIN { FS=":"
 print "\n\t\tsalary statement of employees for the month of feb"
 print "sl.no","\t","name","\t\t","designation","\t","BASIC",
 "\t","DA","\t","HRA","\t","GROSS"
 print}
 { slno++; basic_tot+=$5;
  if($5>=10000)
{da=0.45*$5;da_tot+=da;
hra=0.15*$5;hra_tot+=hra;}
else { da=0.50*$5;da_tot+=da;
hra=0.20*$5;hra_tot+=hra;}
sal_tot+=$5 + da +hra
printf "%2d\t%-15s %-12s %8d %8.2f %8.2f %8.2f\n",slno,$2,$3,$5,da,hra,$5+da+hra}
 END {print"\n\t basic salary paid is rs" basic_tot
 print"\n\t total da paid in rs" da_tot
```

```
print"\n\t total hra paid is" hra_tot

print"\n\t total salary paid" sal_tot

}
```

OUTPUT