

EXPT NO : 6

REPORT

DATE :29/03/2019

AIM : To write the shell scripts of the given questions and verify the output

=====

1. Write a shell script that displays a special listing showing the permissions, size filename and last modification time of filename supplied as arguments. Provide suitable headers using the printf command.

Algorithm:

- 1) Start
- 2) Read the arguments , say n.
- 3) If n=0, print “required at least one filename as argument” and go to step 9.
- 4) If arguments are not valid print “required valid filenames as arguments” and exit.
- 5) For i=n to 1 do
- 6) Print permission , size, last modified date , etc.
- 7) Set i=i-1
- 8) End for.
- 9) Stop

Program:

```
#!/bin/bash
if [[ $# -eq 0 ]]
then
    echo "No file specified as arguments"
    exit
fi
for file in $*
do
    if [[ !(-a $file) ]]
    then
        echo "Enter a valid filename"
```

```

        exit
    fi
done
for file in $* do
    NAME=`basename $file`
    PERMS=`ls -lah $file | awk -F " " '{print $1}'` SIZE=`ls -lah $file | awk -F " " '{print
$5}'`
    DATE=`ls -lah $file | awk -F " " '{print $6 " " $7 " " $8 }'`
    echo "The name of file is: $NAME"
    echo "The last modified date is: $DATE"
    echo "The permissions are: $PERMS"
    echo "The size is: $SIZE"
done

```

Output:

```

haseena@localhost:~$ nano s1.sh
haseena@localhost:~$ nano sample.txt
haseena@localhost:~$ bash s1.sh sample.txt
The name of file is: sample.txt
The last modifieddate is: Mar 29
The permissions are: -rw-r--r--
The size is: 25
haseena@localhost:~$ |

```

2. Write a script that compares two directories dir1 and dir2(supplied as arguments) and copies to dir1 from dir2 every file that is not present in dir1.

Algorithm:

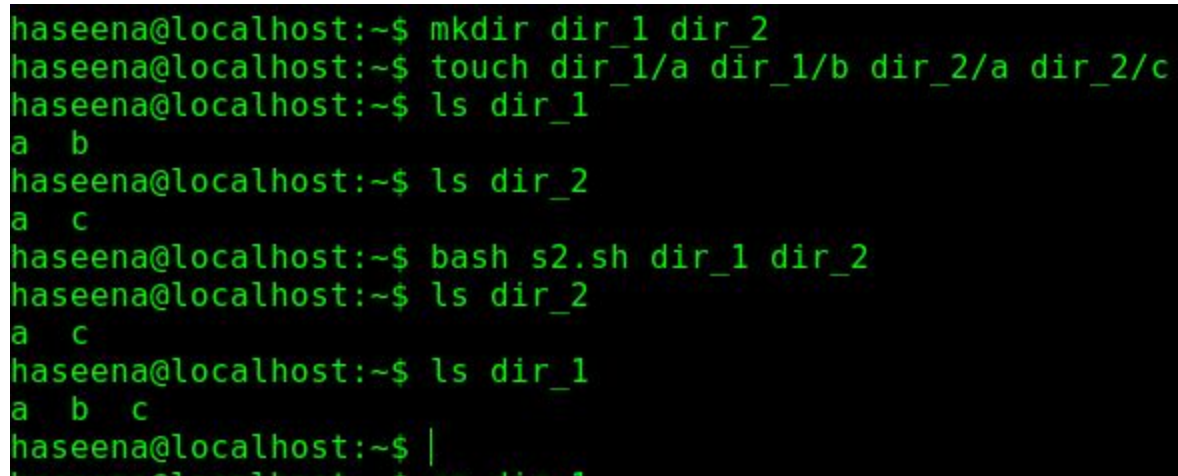
- 1) Start
- 2) Read the arguments , say n.
- 3) If n!=2, print "required two directories as argument" and go to step 9.
- 4) If arguments are not valid print "required valid directories as arguments" and exit.
- 5) Compare dir1 and dir2
- 6) If same filenames present in both do nothing
- 7) Else copy files of dir1 to dir2

- 8) End if
- 9) Stop

Program:

```
#!/bin/bash
if [[ $# -ne 2 ]]
then
    echo "Enter two directories as argument"
    exit
fi
if [[ !(-d $1 && -d $2) ]]
then
    echo "Enter valid directories"
    exit
fi
NAME2=`basename $2`
diff $1 $2 | grep -w "$NAME2" | awk -F ":" '{print $2}' >>
b.temp
while read line
do
    cp "$2/$line" $1
done < b.temp
rm b.temp
```

Output:



```
haseena@localhost:~$ mkdir dir_1 dir_2
haseena@localhost:~$ touch dir_1/a dir_1/b dir_2/a dir_2/c
haseena@localhost:~$ ls dir_1
a  b
haseena@localhost:~$ ls dir_2
a  c
haseena@localhost:~$ bash s2.sh dir_1 dir_2
haseena@localhost:~$ ls dir_2
a  c
haseena@localhost:~$ ls dir_1
a  b  c
haseena@localhost:~$ |
```

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

Algorithm:

- 1) Start
- 2) Read the arguments , say n.
- 3) If n!=2, print "required two directories as argument" and go to step 8.
- 4) Find permission of the files and compare
- 5) If permissions equal then print "same permissions"
- 6) Else print each files with permissions
- 7) End if
- 8) Stop

Program:

```
#!/bin/bash
if [[ $# -ne 2 ]]
then
    echo "Enter two files as argument"
    exit
fi
if [[ !(-a $1 && -a $2) ]]
then
    echo "Enter valid files"
    exit
fi
PERMS1=`ls -lah $1 | awk -F " " '{print $1}'`
PERMS2=`ls -lah $2 | awk -F " " '{print $1}'`
if [[ $PERMS1== $PERMS2]]
then
    echo "The common permission is: $PERMS1"
else
    echo "The permission for $1 is: $PERMS1"
    echo "The permission for $2 is: $PERMS2"
fi
```

Output:

```
haseena@localhost:~$ nano s3.sh
haseena@localhost:~$ touch exec file
haseena@localhost:~$ chmod ugo+rwX exec
haseena@localhost:~$ bash s3.sh exec file
The permission for exec is: -rwxrwxrwx
The permission for file is: -rw-r--r--
haseena@localhost:~$
```

4. Write a shell script which receives two file names as arguments. It should check whether the two file contents are same or not. If they are same then second file should be deleted.

Algorithm:

- 1) Start
- 2) Read the arguments , say n.
- 3) If n!=2, print "required two files as argument" and go to step 9.
- 4) If arguments are not valid print "required valid files as arguments" and exit.
- 5) Compare file1 and file2
- 6) If same contents present in both delete file2
- 7) Else print "files not repeated"
- 8) End if
- 9) Stop

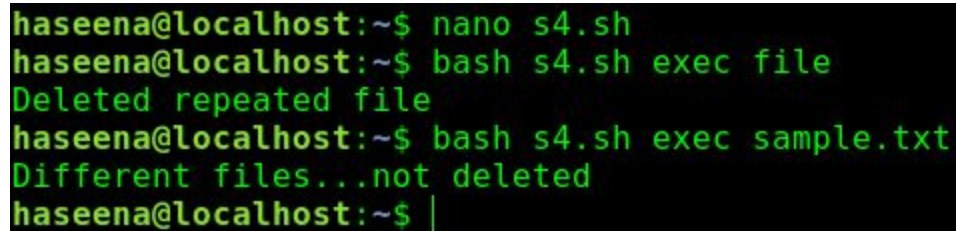
Program:

```
#!/bin/bash
if [[ $# -ne 2 ]]
then
    echo "Enter two files as argument"
    exit
fi
if [[ !(-f $2 && -f $2) ]]
then
    echo "Enter valid files"
    exit
```

fi

```
(cmp -s $1 $2) && (rm $2; echo "Deleted repeated file") ||  
(echo "Different files...not deleted")
```

Output:



```
haseena@localhost:~$ nano s4.sh  
haseena@localhost:~$ bash s4.sh exec file  
Deleted repeated file  
haseena@localhost:~$ bash s4.sh exec sample.txt  
Different files...not deleted  
haseena@localhost:~$ |
```

5. Write a shell script that, given a filename as the argument will count vowels, blank spaces, characters, number of line and symbols.

Algorithm:

- 1) Start
- 2) Read the arguments , say n.
- 3) If n=0, print “required at least one file as argument” and go to step 7.
- 4) If arguments are not valid print “required valid file as arguments” and exit.
- 5) Take count of the vowels, blank spaces, characters, number of line and symbols.
- 6) Print the count
- 7) Stop

Program:

```
#!/bin/bash  
if [[ $# -ne 1 ]]  
then  
    echo "Enter file as argument"  
    exit  
fi  
if [[ !(-a $1) ]]  
then  
    echo "Enter valid file"  
    exit  
fi
```

```

SYM=0
BS=`grep -o ' ' $1 | wc -l`
CHA=`wc -c $1 | awk '{print $1}'`
V=0
L=`wc -l $1 | awk '{print $1}'`
while read -n1 c
do
    if [[ $c == *['!'"@#$%^&*()_+]* ]]
    then
        ((SYM++))
    elif [[ $c == *[aAeEiloOuU]* ]]
    then
        ((V++))
    fi
done < "$1"
echo "The number of lines are: $L"
echo "The number of vowels are: $V"
echo "The number of characters are: $CHA"
echo "The number of spaces are: $BS"
echo "The number of symbols are: $SYM"

```

Output:

```

haseena@localhost:~$ nano s5.sh
haseena@localhost:~$ nano sample.txt
haseena@localhost:~$ cat sample.txt
Hello World $$$## TEST 123
LINE 2 %@@@
haseena@localhost:~$ bash s5.sh sample.txt
The number of lines are: 2
The number of vowels are: 6
The number of characters are: 39
The number of spaces are: 6
The number of symbols are: 9
haseena@localhost:~$ |

```

6. Write a shell script that will take an input file and remove identical lines.

Algorithm:

- 1) Start
- 2) Read the arguments , say n.
- 3) If n=0, print “required a file as argument” and go to step 9.
- 4) If arguments are not valid print “required valid file as arguments” and exit.
- 5) Read contents of file and check if duplicates present
- 6) If present delete duplicates
- 7) Display the file after deletion
- 8) Stop

Program:

```
#!/bin/bash
if [[ $# -ne 1 ]]
then
    echo "Enter file as argument"
    exit
fi
if [[ !(-a $1) ]]
then
    echo "Enter valid file"
    exit
fi
awk '!a[$0]++' $1 >> b.temp
rm $1
mv b.temp $1
```

Output:


```

haseena@localhost:~$ nano s6.sh
haseena@localhost:~$ nano sample.txt
haseena@localhost:~$ cat sample.txt
test
will remove
same lines
test
like
same lines
test
haseena@localhost:~$ bash s6.sh sample.txt
haseena@localhost:~$ cat sample.txt
test
will remove
same lines
like
haseena@localhost:~$

```

7. Write a shell script that displays a list of all the files in the current directory to which the user has read, write and execute permissions.

Algorithm:

- 1) Start
- 2) Read the arguments , say n.
- 3) If n=0, print "required a filenames as argument" and go to step 7.
- 4) If arguments are not valid print "required valid files as arguments" and exit.
- 5) Read permissions of file and compare
- 6) If permission contain 'rwx' then print filename
- 7) Stop

Program:

```

#!/bin/bash
ls -lAh $PWD | awk -F " " '{print $1 " " $9}' > b.temp
while read l
do
    PERM=`echo $l | awk '{print $1}'`
    USRPERM=${PERM:7:3}
    if [[ $USRPERM == "rwx" ]]
    then

```

```

        echo "$I"
    fi
done < b.temp
rm b.temp

```

Output:

```

haseena@localhost:~$ nano s7.sh
haseena@localhost:~$ ls -l
total 1516
-rw-r--r-- 1 haseena haseena      0 Feb 22 22:55 absolute_path.c
-rwxr-xr-x 1 haseena haseena  8640 Mar 17 14:19 a.out
-rw-r--r-- 1 haseena haseena      0 Feb 22 22:35 Count.txt
drwxr-xr-x 4 haseena haseena  4096 Mar 21 21:28 Desktop
drwxr-xr-x 3 haseena haseena  4096 Feb 22 22:19 dir_1
drwxr-xr-x 2 haseena haseena  4096 Mar 29 00:05 dir_2
drwxr-xr-x 6 haseena haseena  4096 Mar 26 20:43 Documents
drwxr-xr-x 4 haseena haseena  4096 Mar 24 12:19 DotslashMain
drwxr-xr-x 7 haseena haseena  4096 Mar 27 23:55 Downloads
-rwxrwxrwx 1 haseena haseena      0 Mar 29 00:20 exec
-rw-r--r-- 1 haseena haseena   312 Feb 22 23:23 FileList.txt
-rw-r--r-- 1 haseena haseena 1393487 Mar 13 22:57 mozilla.pdf
drwxr-xr-x 2 haseena haseena  4096 Feb 17 18:37 Music
drwxr-xr-x 2 haseena haseena  4096 Feb 22 22:18 newdir
drwxr-xr-x 2 haseena haseena 12288 Mar 29 00:36 Pictures
-rw-r--r-- 1 haseena haseena   188 Feb 22 23:49 SortCountry.txt
-rw-r--r-- 1 haseena haseena   359 Feb 22 22:53 Story.txt
-rw-r--r-- 1 haseena haseena      0 Feb 22 23:27 symlink
drwxr-xr-x 2 haseena haseena  4096 Feb 17 18:37 Templates
-rw-r--r-- 1 haseena haseena    58 Mar 17 14:19 test.c
drwxr-xr-x 2 haseena haseena  4096 Feb 21 18:23 testdir
-rw-r--r-- 1 haseena haseena   359 Feb 22 23:31 Togglestory.txt
drwxr-xr-x 2 haseena haseena  4096 Feb 17 18:37 Videos
haseena@localhost:~$ bash s7.sh
-rwxrwxrwx exec
haseena@localhost:~$

```

- 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 the processing is to be continued with the residue. The input is to be through a text file created by the user.

Algorithm:

- Start

- 2) Read the arguments , say n.
- 3) If n!=1, print “required filename as argument” and go to step 8.
- 4) If arguments are not valid print “required valid filename as arguments” and exit.
- 5) Read contents of files and by ignoring spaces count in each line 40 characters
- 6) Separate it with a '/' and display balance characters in next line
- 7) Print it on terminal
- 8) stop

Program:

```
#!/bin/bash
if [[ $# -ne 1 ]]
then
    echo "Enter file as argument"
    exit
fi
if [[ !(-a $1) ]]
then
    echo "Enter valid file"
    exit
fi
n=`wc -l $1 | cut -d " " -f 1`
i=1
while [ $i -le $n ]
do
    line=`sed -n "$i p" $1`
    cc=`echo $line | wc -c | cut -d " " -f 1`
    while [ $cc -ge 40 ]
    do
        ext=`echo $line | cut -c 41-`
        line=`echo $line | cut -c 1-40`
        echo "$line \\"
        line=$ext
        cc=`echo $ext | wc -c | cut -d " " -f 1`
    done
    echo "$line"
    i=`expr $i + 1`
done
```

Output:

```
haseena@localhost:~$ nano s8.sh
haseena@localhost:~$ nano file
haseena@localhost:~$ cat file
ohellohellohellohellohellohellohellohellohellohellohellohell
ohellohellohellohellohellohellohellohellohellohellohello
lohellohellohellohellohellohellohellohellohellohellohell
llohellohellohellohellohellohellohellohellohellohellohello
hellohellohellohellohellohellohellohellohellohellohellohello
hellohellohellohellohellohellohellohellohellohellohello
haseena@localhost:~$ bash s8.sh file
ohellohellohellohellohellohellohellohell \
ohellohellohellohellohellohellohellohell \
ohellohellohellohellohellohellohellohell \
ohellohellohello
lohellohellohellohellohellohellohell \
lohellohellohellohell
llohellohellohellohellohellohellohe \
llohellohellohellohellohellohello
hellohellohellohellohellohellohello \
hellohellohellohellohellohellohello \
hellohellohellohellohellohellohello \
hellohellohellohello
haseena@localhost:~$ |
```

9. Write a shell script to delete all lines containing a specific word in one or more file supplied as argument to it.

Algorithm:

- 1) Start
- 2) Read the arguments , say n.
- 3) If n=0 , print "required atleast one file as argument" and go to step 9.
- 4) Read the word to search .
- 5) Read contents of the file and search word containing links
- 6) If words present then delete that line
- 7) Stop

Program:

```
#!/bin/bash
if [ $# -eq 0 ]
then
    echo "Enter atleast one file"
exit
```

```
fi
echo "Enter word to be searched"
read word
for file in $*
do
    sed "/$word/d" $file > b.temp
    mv b.temp $file
done
```

Output:

```
haseena@localhost:~$ nano s9.sh
haseena@localhost:~$ nano q1
haseena@localhost:~$ nano q2
haseena@localhost:~$ cat q1
test
will be
deleted
test
from
test
the files
haseena@localhost:~$ cat q2
this
test file
will not contain
test
test file
haseena@localhost:~$ bash s9.sh q1 q2
Enter word to be searched
test
haseena@localhost:~$ cat q1
will be
deleted
from
the files
haseena@localhost:~$ cat q2
this
will not contain
haseena@localhost:~$ |
```

CONCLUSION

Verified the outputs for the above questions and improved the hold over bash shell scripting.