Shree Rahul Education Society's (Regd.)
SHREE L. R. TIWARI COLLEGE OF ENGINEERING
Kanakia Park, Near Commissioner's Bungalow, Mira Road (East), Thane 401107, Maharashtra
(Approved by AICTE, Govt. of Maharashtra & Affiliated to University of Mumbai)
Tel. No.: 022-65295732 / 022-65142376 | Email: slrtce@rahuleducation.com | Website: www.slrtce.in

DEPARTMENT OF INFORMATION TECHNOLOGY

TEITC502 Operating Systems

Fifth Semester, 2017-2018 (Odd Semester)

Name of Student :

Roll No. :

Division : B

Batch : B1

Assignment No. : 01 (Mini Project)

Date of Submission : 11/10/ 17

Particulars	Max. Marks	Marks Obtained
Timely Submission (TS)	3	
Originality of the Material(OM)	3	
Neatness (NT)	3	
Innovative Solution (IS)	3	
Total	12	

Grades – Meet Expectations (3 Marks), Moderate Expectations (2 Marks), Below Expectations (1 Mark)

Checked and Verified by

Name of Faculty : Prof. Saurabh Suman

Signature :

Date :

SHELL SCRIPTING

Third Year Information Technology

by

Harshvardhan Dubey

Sumit Awinash

Dayashankar Das

Jyotish Dubey

Under the supervision of

Prof. Saurabh Suman

October 2017

SHREE L. R. TIWARI COLLEGE OF ENGINEERING, MIRA ROAD

(Approved by AICTE, Govt. of Maharashtra & Affiliated to University of Mumbai)

DEPARTMENT OF INFORMATION TECHNOLOGY

2017-18

A Mini Project

on

SHELL SCRIPTING

Submitted in fulfillment of the requirements for the course in

Operating Systems

THIRD YEAR ENGINEERING

October 2017

SHREE L. R. TIWARI COLLEGE OF ENGINEERING, MIRA ROAD

(Approved by AICTE, Govt. of Maharashtra & Affiliated to University of Mumbai)

DEPARTMENT OF INFORMATION TECHNOLOGY

2017-18

SHELL SCRIPTING:

A **shell script** is a text file that contains a sequence of commands for a UNIX-based operating system. It's called a **shell script** because it combines into a "**script**" in a single file a sequence of commands that would otherwise have to be presented to the system from a keyboard one at a time.
A shell script acts as an intermediate between the user and the kernel. That is, it is used for the communication between the user and the kernel.
Every Unix shell offers two key facilities. In addition to an interactive command interpreter, it also offers a very useful programming environment. The shell programming environment accepts programs written in shell programming language. In later sections in this chapter we shall learn about the shell programming language and how to make use of it for enhancing productivity. A user obtains maximum gains when he learns not only to automate a sequence of commands but also to make choices within command sequences and invoke repeated executions.

SCRIPTS:

1. Create a shell script which creates a list of all broken symbolic links on the system and print it.

In computing, a symbolic link (also symlink or soft link) is the nickname for any file that contains a reference to another file or directory in the form of an absolute or relative path and that affects pathname resolution.

A symbolic link contains a text string that is automatically interpreted and followed by the operating system as a path to another file or directory. This other file or directory is called the "target". The symbolic link is a second file that exists independently of its target. If a symbolic link is deleted, its target remains unaffected. If a symbolic link points to a target, and sometime later that target is moved, renamed or deleted, the symbolic link is not automatically updated or deleted, but continues to exist and still points to the old target, now a non-existing location or file. Symbolic links pointing to moved or non-existing targets are sometimes called ***broken***, *orphaned*, *dead*, or *dangling*.

SCRIPT:

```
echo "Broken symbolic link program "
find ./ -type l -exec file {} \; |grep broken

echo "Broken symbolic link found"
```

OUTPUT:

```
root@sumit-VirtualBox:/home/sumit/awii# bash broken.sh
Broken symbolic link program
Broken symbolic link found
root@sumit-VirtualBox:/home/sumit/awii#
```

```
./proc/359/ns/net: broken symbolic link to net:[4026531957]
./proc/359/ns/uts: broken symbolic link to uts:[4026531838]
./proc/359/ns/ipc: broken symbolic link to ipc:[4026531839]
./proc/359/ns/pid: broken symbolic link to pid:[4026531836]
./proc/359/ns/user: broken symbolic link to user:[4026531837]
./proc/359/ns/mnt: broken symbolic link to mnt:[4026531840]
./proc/359/ns/cgroup: broken symbolic link to cgroup:[4026531835]
./proc/665/task/665/fd/3: broken symbolic link to socket:[11975]
./proc/665/task/665/ns/net: broken symbolic link to net:[4026531957]
./proc/665/task/665/ns/uts: broken symbolic link to uts:[4026531838]
./proc/665/task/665/ns/ipc: broken symbolic link to ipc:[4026531839]
./proc/665/task/665/ns/pid: broken symbolic link to pid:[4026531836]
./proc/665/task/665/ns/user: broken symbolic link to user:[4026531837]
./proc/665/task/665/ns/mnt: broken symbolic link to mnt:[4026531840]
./proc/665/task/665/ns/cgroup: broken symbolic link to cgroup:[4026531835]
./proc/665/task/679/fd/3: broken symbolic link to socket:[11975]
./proc/665/task/679/ns/net: broken symbolic link to net:[4026531957]
./proc/665/task/679/ns/uts: broken symbolic link to uts:[4026531838]
./proc/665/task/679/ns/ipc: broken symbolic link to ipc:[4026531839]
./proc/665/task/679/ns/pid: broken symbolic link to pid:[4026531836]
./proc/665/task/679/ns/user: broken symbolic link to user:[4026531837]
./proc/665/task/679/ns/mnt: broken symbolic link to mnt:[4026531840]
./proc/665/task/679/ns/cgroup: broken symbolic link to cgroup:[4026531835]
```

2. Create a script which compresses all files individually in a given directory and all it's subdirectories using gzip. This is handy for saving space on a system where files containing documentation for programs installed on the system can be compressed.

The "gzip" command is a common way of compressing files within Linux and therefore it is worth knowing how to compress files using this tool.

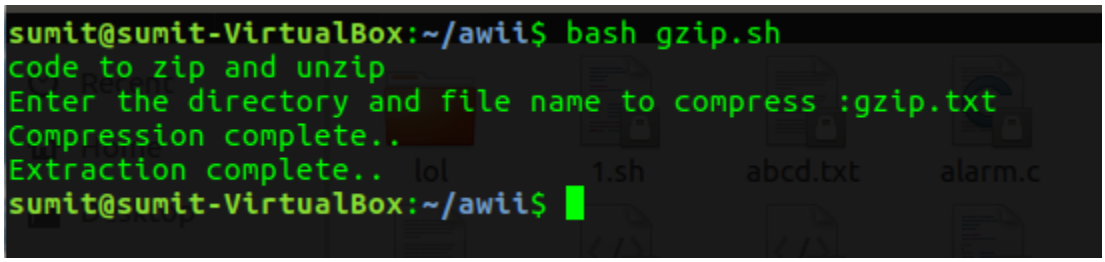
The compression method used by "gzip" is Lempel-Ziv (LZ77). All you need to know is that the files get smaller when you compress them with the "gzip" command.

By default when you compress a file or folder using the "gzip" command it will have the same file name as it did before but now it will have the extension ".gz".

SCRIPT:

```
echo "code to zip and unzip "
read -p "Enter the directory and file name to compress :" filename
gzip "$filename"
echo "Compression complete.."
gunzip "$filename".gz
echo "Extraction complete.."
```

OUTPUT:



3. Write a shell script that creates a list of all mp3 files on a file system containing complete path names of these. This file can be used as a play-list for media players on Linux. Enhance the script to invoke your favorite media player (mplayer, xmms etc.) with this play-list.

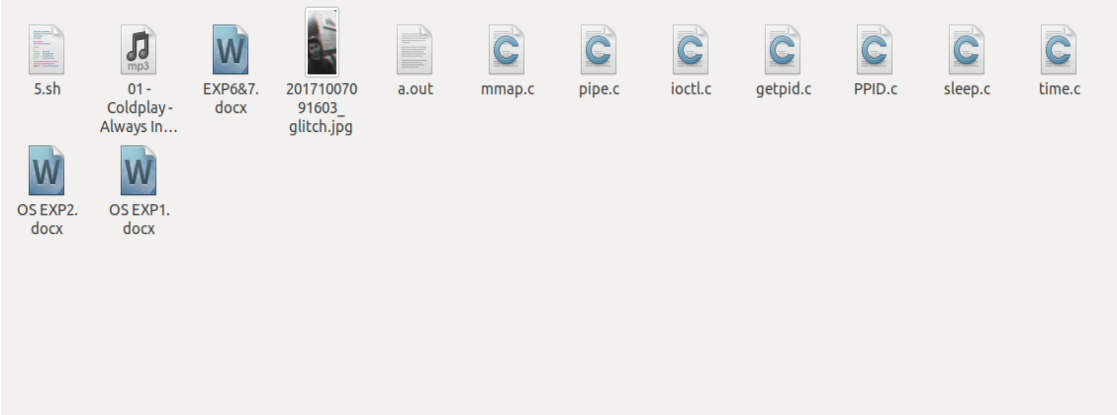
The find command in UNIX is a command line utility for walking a file hierarchy. It can be used to find files and directories and perform subsequent operations on them.

SCRIPT:

```
echo "playlist manager "
find / -iname "*.mp3" -print
echo "Playlist displayed successfully "
```

OUTPUT:

```
root@sumit-VirtualBox:/home/sumit/awii# bash playlist.sh
playlist manager
/home/sumit/Downloads/01 - Coldplay - Always In My Head.mp3
find: ‘/run/user/1000/gvfs’: Permission denied
Playlist displayed successfully
root@sumit-VirtualBox:/home/sumit/awii#
```



4. There is a wget program to automatically download complete websites for viewing the websites offline later. Create a shell script which uses wget to download a given website automatically for you in a given directory for later viewing. After the website has been downloaded have the script tar and gzip the website in a separate file.

wget is a free utility for non-interactive download of files from the web. It supports HTTP, HTTPS, and FTP protocols, as well as retrieval through HTTP proxies.

wget is non-interactive, meaning that it can work in the background, while the user is not logged on, which allows you to start a retrieval and disconnect from the system, letting wget finish the work. By contrast, most web browsers require constant user interaction, which make transferring a lot of data difficult.

wget can follow links in HTML and XHTML pages and create local versions of remote websites, fully recreating the directory structure of the original site, which is sometimes called "recursive downloading." While doing that, wget respects the Robot Exclusion Standard (robots.txt). wget can be instructed to convert the links in downloaded HTML files to the local files for offline viewing.

SCRIPT:

```
echo "WGET Program"
read -p "Enter the website URL :" url
wget "$url"
echo "Successfully created "
echo "Now, creating gzip archive... "
gzip index.html
read -p "Enter directory to put the archive in :" path
cp index.html.gz "$path"
echo "Successfully archived "
```

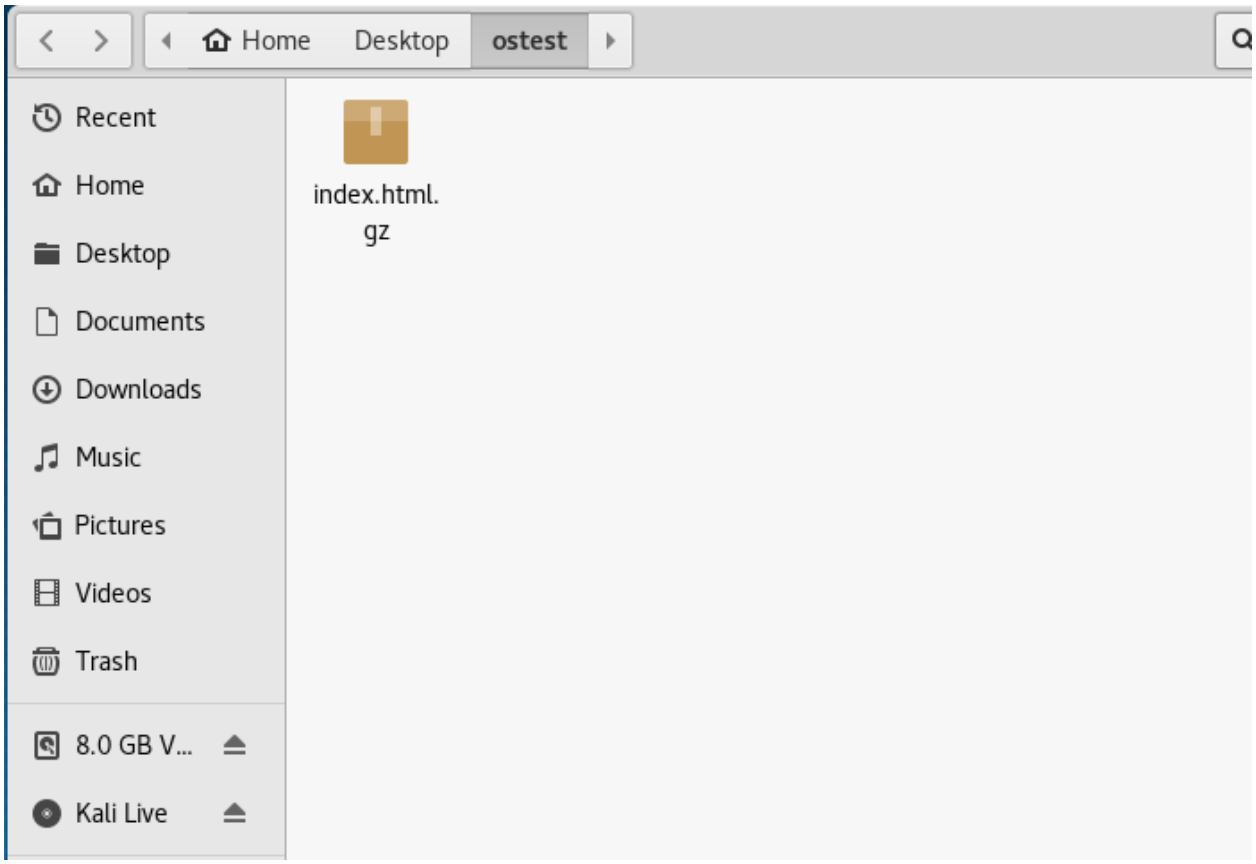
OUTPUT:

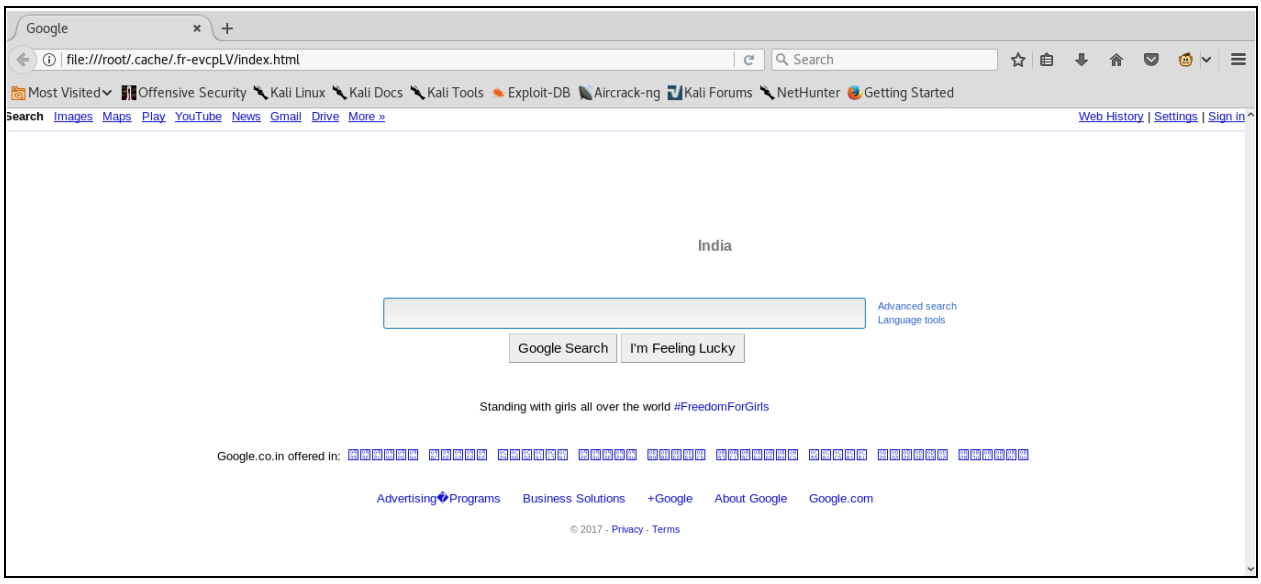
```
root@Batman:~/Desktop/os# bash 5.sh
WGET Program
Enter the website URL :www.google.com
--2017-10-11 22:55:55--  http://www.google.com/
Resolving www.google.com (www.google.com)... 172.217.27.196, 2404:6800:4009:800::2004
Connecting to www.google.com (www.google.com)|172.217.27.196|:80... connected.
HTTP request sent, awaiting response... 302 Found
Location: http://www.google.co.in/?gfe_rd=cr&dcr=0&ei=I1TeWbO4LrCcX7aTl8AH [following]
--2017-10-11 22:55:55--  http://www.google.co.in/?gfe_rd=cr&dcr=0&ei=I1TeWbO4LrCcX7aTl8AH
Resolving www.google.co.in (www.google.co.in)... 172.217.27.195, 2404:6800:4009:800::2003
Connecting to www.google.co.in (www.google.co.in)|172.217.27.195|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [text/html]
Saving to: 'index.html'

index.html              [ <=>                ]  13.88K  --.-KB/s    in 0.001s

2017-10-11 22:55:55 (17.2 MB/s) - 'index.html' saved [14208]

Successfully created
Now, creating gzip archive...
Enter directory to put the archive in :/root/Desktop/ostest
Successfully archived
```





5. Write a script that adds a directory to the system path variable.

PATH is an environment variable on Unix-like operating systems, DOS, OS/2, and Microsoft Windows, specifying a set of directories where executable programs are located. In general, each executing process or user session has its own PATH setting.

We can invoke programs directly from that directory that has been added to the variable whenever necessary without addressing the whole directory.

SCRIPT:

```
echo "Add directory to PATH variable "
read -p "Enter the directory :" path
export PATH=$PATH:"$path"
echo "---echo successful---"
echo "Your directory has been added to the path variable..
Now you can invoke programs directly from that directory without specifying the whole directory "
```

OUTPUT:

```
root@Batman:~/Desktop/os# bash 6.sh
Add directory to PATH variable
Enter the directory :/root/Desktop/os
---echo successful---
Your directory has been added to the path variable..
Now you can invoke programs directly from that directory without specifying the whole directory
root@Batman:~/Desktop/os#
```

6. Write a script that copies a list of files from one directory to the other. During copying the script should make the following changes to the files: capitalize the first letter of the filename and lowercase all other letters. Change the owner of the files to a given owner and change the permissions to a given set of permissions.

cp command here is used to copy files from the source directory to the destination directory and sed is used to edit the filename with only first letter as uppercase.

chown is used to change the file owner.

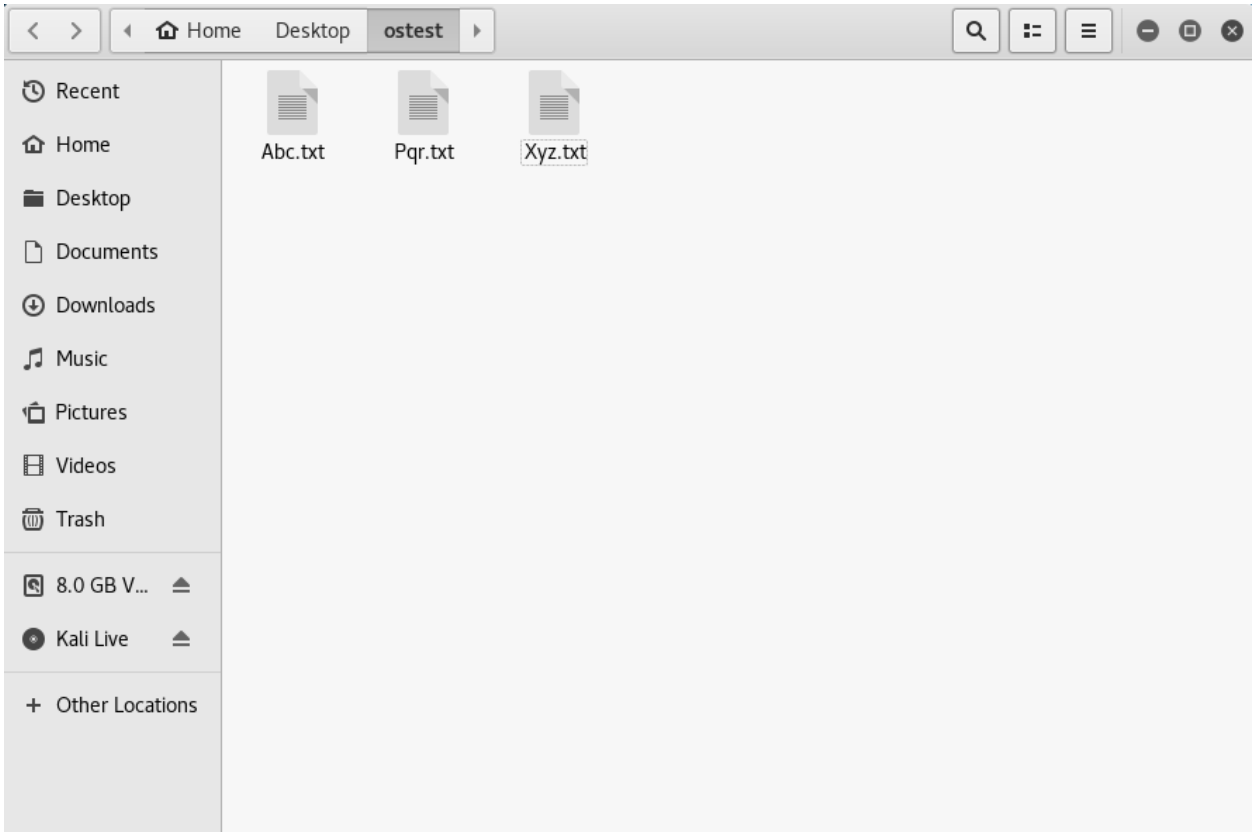
chmod is used to change the file permissions.

SCRIPT:

```
echo "Copy multiple files to another directory "
read -p "Enter source directory :" sourcedir
read -p "Enter the names of the files you want to copy to another directory :" file1 file2 file3
read -p "Enter the directory where you want to copy the files to :" destdir
sed 's/./\u&/' <<< "$file1"
sed 's/./\u&/' <<< "$file2"
sed 's/./\u&/' <<< "$file3"
cp "$sourcedir"/{"$file1","$file2","$file3"} "$destdir"
echo "Copy successful"
read -p "Enter the user you want to allocate this file to :" user
chown "$user" "$file1"
chown "$user" "$file2"
chown "$user" "$file3"
echo "Files allocated to $user "
read -p "Enter permission (eg. 777 etc) :" per
chmod -R "$per" "$destdir"
echo "Destination folder permissions updated to $per"
```

OUTPUT:

```
root@Batman:~/Desktop/os# bash 7.sh
Copy multiple files to another directory
Enter source directory :/root/Desktop/os
Enter the names of the files you want to copy to another directory :abc.txt pqr.txt xyz.txt
Enter the directory where you want to copy the files to :/root/Desktop/ostest
Abc.txt
Pqr.txt
Xyz.txt
Copy successful
Enter the user you want to allocate this file to :root
Files allocated to root
Enter permission (eg. 777 etc) :444
Destination folder permissions updated to 444
root@Batman:~/Desktop/os# ls -l /root/Desktop/ostest
total 0
-r--r--r-- 1 root root 0 Oct 11 23:03 Abc.txt
-r--r--r-- 1 root root 0 Oct 11 23:03 Pqr.txt
-r--r--r-- 1 root root 0 Oct 11 23:03 Xyz.txt
root@Batman:~/Desktop/os#
```



7. Write a script to convert DOS CR/LF files to UNIX files in which lines end in LF.

The main difference between UNIX and DOS is that DOS was originally designed for single-user systems, while UNIX was designed for systems with many users. While PC's have evolved into GUI interfaces such as Windows, UNIX systems have never evolved into GUI environments.
The commands in a DOS file end in CR/LF when they have to go to a new line. Whereas in UNIX, the files only LF to change the line.
Here we use the sed (Stream editor) command to change every occurrence of CR/LF to only LF.

SCRIPT:

```
echo "Program to convert the DOS files to UNIX files "
read -p "Enter the DOS filename :" dos
sed 's/$'"/`echo \\\r`/" "$dos" > unix.txt
echo "The DOS file with CR and LF has
been converted to UNIX file with only LF"
```

OUTPUT:

```
root@Batman:~/Desktop/os# bash 8.sh
Program to convert the DOS files to UNIX files
Enter the DOS filename :dos.txt
The DOS file with CR and LF has
been converted to UNIX file with only LF
root@Batman:~/Desktop/os#
```