



CMR ENGINEERING COLLEGE

(Approved By AICTE-New Delhi, Affiliated to JNTUH)

Kandlakoya(V), Medchal Road, Hyderabad-501401

Department of Computer Science & Engineering

LAB MANUAL

Name of the Lab : Linux Programming

Class : IV Year I Semester

Branch : Computer Science & Engineering

Year : IV Year I semester

Regulation : R13

A.Y. : 2017-18

Vision of the Department

To produce globally competent and industry ready graduates in Computer Science & Engineering by imparting quality education with a know-how of cutting edge technology and holistic personality.

Mission of the Department

M1. To offer high quality education in Computer Science & Engineering in order to Build core competence for the students by laying solid foundation in Applied Mathematics, and program framework with a focus on concept building.

M2. The department promotes excellence in teaching, research, and collaborative Activities to prepare students for professional career or higher studies.

M3. Creating intellectual environment for developing logical skills and problem solving Strategies, thus to develop, able and proficient computer engineer to compete in the Current global scenario.

Linux programming Lab

System Requirements

1. Intel based desktop PC of 166MHz or faster processor with at least 64 MB RAM and 100 MB free disk space.
2. LINUX server supporting 60 systems
3. Putty software in each terminal to connect to server and use BASH SHELL for SHELL SCRIPTS.

Lab Objectives

Upon successful completion of this Lab the student will be able to:

1. Demonstrate how to use the following Bourne Shell commands: cat, grep, ls, more, ps, chmod, finger, ftp, etc.
2. Use the following Bourne Shell constructs: test, if then, if then else, if then elif, for, while, until, and case.
3. Learn tracing mechanisms (for debugging), user variables, Bourne Shell variables, read-only variables, positional parameters, reading input to a Bourne Shell script, command substitution, comments, and exporting variables. In addition, test on numeric values, test on file type, and test on character strings are covered.
4. Copy, move, and delete files and directories
5. Write moderately complex Shell scripts.
6. Make a Shell script executable.
7. Create a ".profile" script to customize the user environment.
8. Use advanced features of File Transfer Protocol (FTP)
9. Compile source code into object and executable modules.
10. Execute programs written in C under UNIX environment

SYLLABUS

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD

IV Year B.Tech. CSE I –Sem

L	T/P/D	C
-	-/3/-	2

(A70596) LINUX PROGRAMMING LAB

Objectives:

- To write shell scripts to solve problems
- To implement some standard Linux utilities such as ls,cp,etc using system calls.
- To develop network based applications using C.

List of Sample Problems:

Note: Use Bash for Shell scripts

1. Write a Shell script that accepts a filename, starting and ending line numbers as arguments and displays all the lines between the given line numbers.
2. Write a Shell script that deletes all lines containing a specified word in one or more files supplied as arguments to it.
3. Write a Shell script that displays list of all the files in the current directory to which the user has read, Write and execute permissions.
4. Write a Shell script that receives any number of file names as arguments checks if every argument supplied is a file or a directory and reports accordingly. Whenever the argument is a file, the number of lines on it is also reported.
5. Write a Shell script that accepts a list of file names as its arguments, counts and reports the occurrence of each word that is present in the first argument file on other argument files.
6. Write a Shell script to list all of the directory files in a directory
7. Write a Shell script to find factorial of a given integer.
8. Write a Shell script to count the number of lines in a file that do not contain vowels.
9. Write an awk script to find the number of characters, words and lines in a file.
10. Write a C Program that makes a copy of a file using standard I/O and system calls.
11. Write in C the following Unix commands using system calls
A.cat B.mv
12. Write a C program to list files in a directory
13. Write a C program to emulate the Unix ls-l command.
14. Write a C program to list for every file in a directory, its inode number and file name.
15. Write a C Program that demonstrates redirection of standard output to a file .EX:
ls>f1.

16. Write a C program to create a child process and allow the parent to display "parent" and the child to display "child" on the screen.
17. Write a C program to create a Zombie process.
18. Write a C program that illustrates how an orphan is created.
19. Write a program that illustrates how to execute two commands concurrently with a command pipe
20. Write C programs that illustrate communication between two unrelated processes using named pipe(FIFO file)
21. Write a C program in which a parent writes a message to a pipe and the child reads the message.
22. Write a C program (sender.c) to create a message queue with read and write permissions to write three messages to it with different priority numbers.
23. Write a C Program (receiver.c) that receives the messages (from the above message queue as specified in (22)) and displays them.
24. Write a C program that illustrates suspending and resuming processes using signals
25. Write a client and server programs in C for connection oriented communication between server and client process using Linux domain sockets to perform the following:
Client process sends a message to the server process. The server receives the message, reverses it and sends it back to the client. The client will then display the message to the standard output device.
26. Write a client and server programs in C for connection oriented communication between server and client process using Internet domain socket to perform the following:
Client process sends a message to the server process. The server receives the message, reverses it and sends it back to the client. The client will then display the message to the standard output device.
27. Write a C program to perform the following:
One process creates a shared memory segment and writes a message "Hello" into it. Another process open the segment and reads the message that is "Helllo". It will then display the message "Hello" to the standard output device.

Text Books:

1. Beginning Linux Programming 4th Edition, N.Matthew,R.Stones, Wrox, Wily India Edition.
2. Advanced Unix Programming N.B.Venkateswarlu, BS Publications.
3. Unix & Shell Programming, M.G.Venkatesh Murthy, Pearson Education
4. Unix shells by example,4th Edition Ellie Quigley, Pearson Education.
5. Sed andawk, O.Dougherty & A.R.Robbins , 2nd Edition, SPD

Outcomes:

- Ability to Understand the Linux environment
- Ability to perform to perform the file management and multiple tasks using shell scripts in Linux environment.

INDEX

Sl.No	Experiment	Page No's
1	Write a Shell script that accepts a filename, starting and ending line numbers as arguments and displays all the lines between the given line numbers.	8-9
2	Write a Shell script that deletes all lines containing a specified word in one or more files supplied as arguments to it.	10-11
3	Write a Shell script that displays list of all the files in the current directory to which the user has read, Write and execute permissions	12-13
4	Write a Shell script that receives any number of file names as arguments checks if every argument supplied is a file or a directory and reports accordingly. Whenever the argument is a file, the number of lines on it is also reported.	14-15
5	Write a Shell script that accepts a list of file names as its arguments, counts and reports the occurrence of each word that is present in the first argument file on other argument files.	16-17
6	Write a Shell script to list all of the directory files in a directory	18-19
7	Write a Shell script to find factorial of a given integer.	20-21
8	Write a Shell script to count the number of lines in a file that do not contain vowels.	22
9	Write an awk script to find the number of characters, words and lines in a file.	23
10	Write a C Program that makes a copy of a file using standard I/O and system calls.	24-25
11	Write in C the following Unix commands using system calls A.cat B.mv	26-29
12	Write a C program to list files in a directory	30-31
13	Write a C program to emulate the Unix ls-l command.	32-34
14	Write a C program to list for every file in a directory, its inode number and file name.	35-36

15	Write a C Program that demonstrates redirection of standard output to a file .EX: ls>f1.	37-38
16	Write a C program to create a child process and allow the parent to display “parent” and the child to display “child” on the screen.	39-40
17	Write a C program to create a Zombie process.	41-42
18	Write a C program that illustrates how an orphan is created.	43-44
19	Write a program that illustrates how to execute two commands concurrently with a command pipe	45-46
20	Write C programs that illustrate communication between two unrelated processes using named pipe(FIFO file)	47-50
21	Write a C program in which a parent writes a message to a pipe and the child reads the message.	51-53
22	Write a C program (sender.c) to create a message queue with read and write permissions to write three messages to it with different priority numbers.	54-57
23	Write a C Program (receiver.c) that receives the messages (from the above message queue as specified in (22)) and displays them.	58-60
24	Write a C program that illustrates suspending and resuming processes using signals	61-62
25	Write a client and server programs in C for connection oriented communication between server and client process using Linux domain sockets to perform the following: Client process sends a message to the server process. The server receives the message, reverses it and sends it back to the client. The client will then display the message to the standard output device.	63-66
26	Write a client and server programs in C for connection oriented communication between server and client process using Internet domain socket to perform the following: Client process sends a message to the server process. The server receives the message, reverses it and sends it back to the client. The client will then display the message to the standard output device.	67-72
27	Write a C program to perform the following: One process creates a shared memory segment and writes a message “Hello” into it. Another process open the segment and reads the message that is “Hello”. It will then display the message “Hello” to the standard output device.	71-73
28	References	74

EXPERIMENT NO: 1.

NAME OF THE EXPERIMENT: Write a Shell Script that accepts a file name, starting and ending line numbers as arguments and displays all lines between the given line numbers.

AIM: By using bash shell for writing the shell scripts, first create any file in that file read any line number as starting and ending line number and print all lines between that line numbers.

ALGORITHM:

Step 1: Create a file with 5-6 lines of data

File can be created by vi text.txt or cat text.txt

Step 2: Now write a shell script with

vi prg1.sh

step3: Check the no of arguments for shell script

if 0 arguments then print no arguments

else if 1 argument then print 1 argument

else if 2 arguments then print 2 arguments

else check for file is there or not (if file is not there print file does not exist)

else sed -ne "\$2','\$3' p' \$1

sed is one of powerful filter (stream editor)

-e default option (script on command line)

-n suppresses automatic output

\$2 first line number passed \$3 2nd line number passed

p is a print command (prints current content to the pattern space).

\$1 is name of file from which we are printing data between the line numbers.

SOURCE CODE

```
_[lakshmi@localhost ~]$ vi prg1.sh
if [ $# -eq 0 ]
then
echo no arguments
elif [ $# -eq 1 ]
then
echo one argument
elif [ $# -eq 2 ]
then
echo two arguments
else
if [ ! -e $1 ]
then
echo file does not exist
else
sed -ne "$2','$3' p' $1
fi
fi
```


EXPECTED OUTPUT

```
lakshmi@localhost:~  
[lakshmi@localhost ~]$ vi prg1.txt  
[lakshmi@localhost ~]$ vi prg1.sh  
[lakshmi@localhost ~]$ vi prg1.sh  
[lakshmi@localhost ~]$ sh prg1.sh text.txt 2 5  
fine thank u  
bye  
hgfdgfh  
jff  
[lakshmi@localhost ~]$
```

Viva Questions

1. Who developed UNIX?

- | | |
|---------------------|--|
| 1. Ken Thompson | 2. Dennis Ritchie, |
| 3. Douglas MacIlroy | 4. Ken Thompson, Dennis Ritchie |

2. What is 'ps' command for?

- | | |
|--------------------------------|-------------------------------------|
| 1. prints the status | 2. prints the process status |
| 3. prints the execution status | 4. none |

3. Which command is used to terminate the process?

- | | |
|---------|----------------|
| 1. ps | 2. who |
| 3. grep | 4. kill |

4. Shell is an interface between user and

- | | |
|------------------|-------------|
| 1. Kernel | 2. hardware |
| 3. I/O | 4. none |

5. Which of the following is not a type shell

- | | |
|------------|---------------|
| 1. korn | 2. bourne |
| 3. C-shell | 4. All |

EXPERIMENT NO: 2

NAME OF THE EXPERIMENT: Write a shell script that deletes all lines containing a specific word in one or more files supplied as arguments to it.

AIM: By using bash shell for writing the shell scripts, first create any file with 5-10 lines of data, then specify any particular word, if that word is present in any lines of file those lines will be deleted.

ALGORITHM:

Step 1: Create a file with 5-6 lines of data

 Create the 2 file f1 and f2 as vi f1 and vi f2

Step2: Now write a shell script with

 vi prg2.sh

step3: Check the no of arguments for shell script

 if 0 arguments then print no arguments

 else pattern=\$1 (word will be stored in pattern)

 for fname in \$*

 for every filename in given files

 if it is a file if [-f \$fname] then

 print DELETING \$pattern FROM \$fname

 sed '/\$pattern/d' \$fname

 sed acts as filter if word is a file in any line that will be deleted

 '/' is used to represent regular expressions

 '/d' is a delete command in sed

 else print file NOT FOUND

SOURCE CODE:

```
[lakshmi@localhost ~]$ vi prg2.sh
```

```
if [ $# -eq 0 ]
```

```
then
```

```
echo NO ARGUMENTS
```

```
else
```

```
pattern=$1
```

```
shift
```

```
for fname in $*
```

```
do
```

```
if [ -f $fname ]
```

```
then
```

```
echo DELETING $pattern FROM $fname
```

```
sed '/$pattern/d' $fname
```

```
else
echo $fname NOT FOUND
fi
done
fi
```

EXPECTED OUTPUT

```
akshmi@localhost ~]$ sh prg2.sh
ARGUMENTS
akshmi@localhost ~]$ vi prg2.sh
akshmi@localhost ~]$ vi prg2.sh
akshmi@localhost ~]$ sh prg2.sh hai f1 f2
LETING hai FROM f1
ne
e

LETING hai FROM f2
ce color
ce day

akshmi@localhost ~]$
```

Viva Questions

1. Which command is use for the copy in Unix?
 1. copy
 2. cp
 3. cpy
 4. none
- 2 .What is stand for IPC?
 1. Inter procedure communication
 2. Inter process communication
 3. Inter part compare
 4. None
- 3 .What is the stand for FIFO?
 1. First in First out
 2. File in File out
 3. First Inter First Out
 4. None

EXPERIMENT: 3

NAME OF THE EXPERIMENT: Write a shell script that displays a list of files in current directory to which the user has read, write and execute permissions.

AIM: In the current directory we are able know which files have read, write and execute permissions.

ALGORITHM

Step1: Here we are working with current directory

Shell script will be written at command line

`ls -l | grep '^.rwx'`

ls is list command (list all the files in current directory)

-l long listing the files

grep acts as filter

it doesn't print the files which does not have read, write and execute permissions

^ represents negation.

SOURCE CODE:

```
$ls -l | grep '^.rwx'
```

EXPECTED OUTPUT

```

[lakshmi@localhost ~]$ ls -l|grep '^.rwx'
[lakshmi@localhost ~]$ ls -l
total 36
-rw-rw-r-- 1 lakshmi lakshmi 26 Jun 29 14:42 f1
-rw-rw-r-- 1 lakshmi lakshmi 39 Jun 29 14:41 f2
-rw-rw-r-- 1 lakshmi lakshmi 22 Jun 29 15:01 f3
-rw-rw-r-- 1 lakshmi lakshmi 212 Jun 29 14:34 prg1.sh
-rw-rw-r-- 1 lakshmi lakshmi 203 Jun 29 14:58 prg2.sh
-rw-rw-r-- 1 lakshmi lakshmi 234 Jun 29 15:06 prg3.sh
-rw-rw-r-- 1 lakshmi lakshmi 159 Jun 29 15:23 prg4.sh
-rw-rw-r-- 1 lakshmi lakshmi 211 Jun 29 14:15 prg.txt
-rw-rw-r-- 1 lakshmi lakshmi 54 Jun 29 14:16 text.txt
[lakshmi@localhost ~]$ ls -l|grep '^.rw'
-rw-rw-r-- 1 lakshmi lakshmi 26 Jun 29 14:42 f1
-rw-rw-r-- 1 lakshmi lakshmi 39 Jun 29 14:41 f2
-rw-rw-r-- 1 lakshmi lakshmi 22 Jun 29 15:01 f3
-rw-rw-r-- 1 lakshmi lakshmi 212 Jun 29 14:34 prg1.sh
-rw-rw-r-- 1 lakshmi lakshmi 203 Jun 29 14:58 prg2.sh
-rw-rw-r-- 1 lakshmi lakshmi 234 Jun 29 15:06 prg3.sh
-rw-rw-r-- 1 lakshmi lakshmi 159 Jun 29 15:23 prg4.sh
-rw-rw-r-- 1 lakshmi lakshmi 211 Jun 29 14:15 prg.txt
-rw-rw-r-- 1 lakshmi lakshmi 54 Jun 29 14:16 text.txt
[lakshmi@localhost ~]$ chmod u+x f1
[lakshmi@localhost ~]$ ls -l|grep '^.rwx'
-rwxrw-r-- 1 lakshmi lakshmi 26 Jun 29 14:42 f1
[lakshmi@localhost ~]$ chmod g+x f1
[lakshmi@localhost ~]$ ls -l|grep '^.rwx'
-rwxrwxr-- 1 lakshmi lakshmi 26 Jun 29 14:42 f1
[lakshmi@localhost ~]$ █

```

METHOD 2:

echo "enter the directory name"

```

read dir
if [ -d $dir ]
then
cd $dir
ls > f
exec < f
while read line
do
if [ -f $line ]
then
if [ -r $line -a -w $line -a -x $line ]
then
echo "$line has all permissions"
else
echo "files not having all permissions"
fi
fi
done
fi

```

EXPECTED OUTPUT

student@ubuntu:~\$sh prg3.sh
enter the directory name
dir1
ff has all permissions
files not having permissions

Method3:

```
echo "List of Files which have Read, Write and Execute Permissions in Current Directory"  
for file in *  
do if [ -r $file -a -w $file -a -x $file ]  
then  
echo $file  
fi  
done
```

INPUT: sh prog3.sh

OUTPUT: List of Files which have Read, Write and Execute Permissions in Current Directory f1 f2 f3

Viva Questions

1.What are states that the page can be in, after causing a page fault?

1. On a swap device and not in memory 2. On the free page list in the main memory,
3. In an executable file, 4. all

2.At what mode the fault handler executes?

1. execution mode 2. kernal mode
3. operation mode 4. none

3.Which one data structure is used for Demand Paging?

1. Page table entries, 2. Disk block descriptors,
3. Page frame data table (pfdata), 4. Sw

EXPERIMENT NO: 4

NAME OF THE EXPERIMENT: Write a shell script that receives any number of file names as arguments checks if every argument is a file or directory, when it is a file, report no of lines in it.

AIM: In this shell script we will supply the file names or directory names as arguments if it is a file name then it prints no of lines in file, if it is a directory then it prints it is a directory file.

ALGORITHM:

Step1: check files and directories in the current directory by ls command

Step2: create a shell script

vi prg3.sh

step3: Check the no of arguments for shell script

if 0 arguments then print no arguments

for fname in \$*

for every filename in given files

if it is a file if [-f \$fname] then

print it is a regular file

count=`wc -l \$fname`

wc is used to count no of lines words characters

-l is used only to print lines.

print count

else

print it is a directory file.

SOURCE CODE

```
[lakshmi@localhost ~]$ vi prg3.sh
if [ $# -eq 0 ]
then
echo NO ARGUMENTS
else
for fname in $*
do
if [ -f $fname ]
then
echo $fname IS A REGULAR FILE
count=`wc -l $fname`
echo NUMBER OF LINES IN $fname ARE :
echo $count
else
echo $fname IS A DIRECTORY FILE
fi
done
fi
```

EXPECTED OUTPUT

```
lakshmi@localhost ~
[lakshmi@localhost ~]$ sh prg3.sh f3
f3 IS A REGULAR FILE
NUMBER OF LINES IN f3 ARE :
3 f3
[lakshmi@localhost ~]$ ls
f1 f2 f3 prg1.sh prg2.sh prg3.sh prg4.sh prg.txt text.txt
[lakshmi@localhost ~]$ sh prg3.sh lakshmi
lakshmi IS A DIRECTORY FILE
[lakshmi@localhost ~]$
```

Viva Questions

1.What is stand for BSS in Unix?

- | | |
|----------------------------|----------------------------|
| 1. Block Started by Symbol | 2. Black Started by Symbol |
| 3. Black Started by Symbol | 4. none |

2.Which one is best in action between fork() and vfork()?

- | | |
|-----------|------------|
| 1. fork() | 2. vfork() |
| 3. both | 4. none |

3.Which symbol will be used with grep command to match the pattern pat at the beginning of a line?

- | | |
|----------------------|----------|
| A. ^pat | B. \$pat |
| C. pat\$ | D. pat^ |
| E. None of the above | |

EXPERIMENT NO: 5

NAME OF THE EXPERIMENT: Write a shell script that accepts a list of file names as its arguments, count and reports the occurrence of each word that is present in the first argument file on other argument files.

AIM: In this shell script we will supply the file names as arguments, any word that is present in first argument file can counted and printed if it present in other argument files.

ALGORITHM:

Step1: Check the no of arguments for shell script
if 0 arguments then print no arguments
step2: else translate each word in the first file is to be on separate line
which will be stored in temp file
step3: for i in \$*
for every filename in given files
step 4: translate each word in the file is to be on separate line
which will be stored in temp1 file
step5: count no of lines in temp file assign it to j
step6: initialize j=1
step 7: while i<j
extract the line that are common in both the file by using
head and tail commands
then apply the filter grep to count and print the lines
which are common to files
increment j
step 8: end

SOURCE CODE

```
if [ $# -eq 0 ]
then
echo "no arguments"
else
tr " " "\n"<$1>temp
shift
for i in $*
do
tr "" "\n"<$i>temp1
y=`wc -l < temp`
j=1
while [ $j le $y ]
do
x=`head -n $j temp | tail -l`
c=`grep -c "$x" temp1`
echo $x $c
j=`expr $j + 1`
```

done
done
fi

EXPECTED OUTPUT

```
[lakshmi@localhost labprograms]$ sh laxmi5.sh sampletext2 sampletext3
hair 3
laxmi5.sh: line 12: [: 1+1: integer expression expected
[lakshmi@localhost labprograms]$ sh laxmi5.sh sampletext2 sampletext3 sampletex
4
hair 3
laxmi5.sh: line 12: [: 1+1: integer expression expected
hair 2
laxmi5.sh: line 12: [: 1+1: integer expression expected
[lakshmi@localhost labprograms]$
```

VIVA QUESTIONS

1. What is Shell Scripting ?

Shell scripting, in Linux or Unix, is programming with the shell using which you can automate your tasks. A shell is the command interpreter which is the interface between the User and the kernel. A shell script allows you to submit a set of commands to the kernel in a batch. In addition, the shell itself is very powerful with many properties on its own, be it for string manipulation or some basic programming stuff.

2. The command "cat file" gives error message "--bash: cat: Command not found". Why?

It is because the PATH variable is corrupt or not set appropriately. And hence the error because the cat command is not available in the directories present PATH variable.

3. How to find the length of a string in Linux?

```
$ x="welcome"      $ echo ${#x} 7
```

4. What are the different timestamps associated with a file?

- Modification time:- Refers to the time when the file is last modified.
- Access time :- The time when the file is last accessed.
- Changed time :- The time when the attributes of the file are last changed.

5. How to get the list of files alone in a directory in Linux?

```
$ ls -lrt | grep ^-
```

EXPERIMENT NO: 6

NAME OF THE EXPERIMENT: Write a shell script to list all of the directory files in a directory.

AIM: In this shell script we will supply the directory name as an argument then our program will return us the list of directories in the given directory

ALGORITHM:

Step1: enter the name of the directory

Read dir

Step2: if it is a directory

Then list the files present in that directory

By using ls command with -p option to list all directory files in a given directory

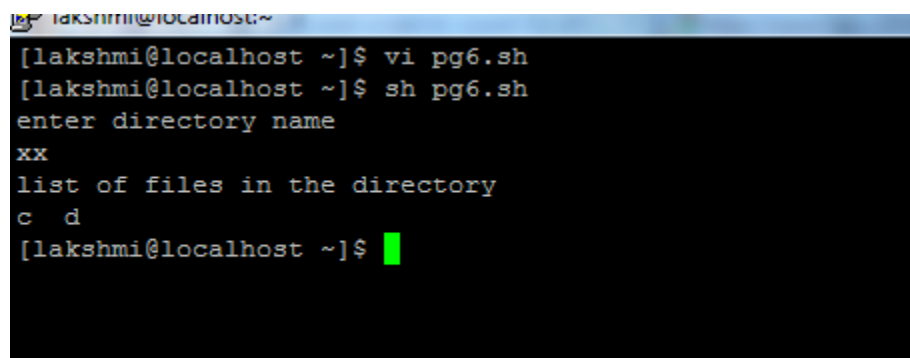
Step 3: else enter the directory name

Step 4: stop

SOURCE CODE

```
echo "enter directory name"
read dir
if [ -d $dir ]
then
echo "list of files in the directory"
ls $dir
else
echo "enter proper directory name"
fi
```

EXPECTED OUTPUT:



```
[lakshmi@localhost ~]$ vi pg6.sh
[lakshmi@localhost ~]$ sh pg6.sh
enter directory name
xx
list of files in the directory
c d
[lakshmi@localhost ~]$
```

Viva Questions

1. What are the different timestamps associated with a file?

1)Modification time 2) Access time 3)Changed time

2. How to get the list of files alone in a directory in Linux?

\$ ls -lrt | grep ^-

3. Which command is used to sort the lines of data in a file in reverse order

A.sort B.sh C.st D.sort -r

EXPERIMENT NO: 7

NAME OF THE EXPERIMENT: Write a shell script to find factorial of a given number.

3.AIM: By using bash shell we will write a shell script to find the factorial of a given number.

ALGORITHM

Step 1: read any number to find factorial

Step 2: initialize fact=1 and i=1

Step 3: while i less than

do

fact=fact* i

i=i+1

done

step 4:print fact

step 5:stop.

SOURCE CODE:

```
echo ENTER A NUMBER :
```

```
read n
```

```
echo THE FACTORIAL OF A GIVEN $n IS :
```

```
fact=1
```

```
i=1
```

```
while [ $i -le $n ]
```

```
do
```

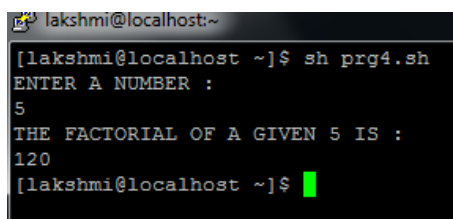
```
fact=`expr $fact \* $i`
```

```
i=`expr $i + 1`
```

```
done
```

```
echo $fact
```

EXPECTED OUTPUT



```
lakshmi@localhost:~$ sh prg4.sh
[ lakshmi@localhost ~]$ sh prg4.sh
ENTER A NUMBER :
5
THE FACTORIAL OF A GIVEN 5 IS :
120
[ lakshmi@localhost ~]$
```

Viva Questions

1. Which command is used to display the top of the file?

- A. cat B. head C. more D. grep
E. None of the above

2. Which command is used to copy all files having the string chap and any two characters after that to the progs directory?

- A. cp chap?? Progs B. cp chap* progs C. cp chap[12] /progs/*.* D. cp chap?? /progs/*

3. Which command is used to change protection mode of files starting with the string emp and ending with 1,2, or 3?

- A. chmod u+x emp[1-3] B. chmod 777 emp*
C. chmod u+r ??? emp D. chmod 222 emp?

EXPERIMENT NO: 8

NAME OF THE EXPERIMENT: Write awk script to count no of lines in a file that do not contain vowels.

AIM: By using bash shell we will write awk script to count the number of lines that do not contain vowels

ALGORITHM:

Step 1: create a file with 5-10 lines of data

Step 2: write an awk script by using grep command to filter the lines that do not contain vowels

```
awk ' $0 !~/aeiou/ {print $0}' file1
```

step3: count=count+1

step4:print count

step5:stop

5. SOURCE CODE:

```
awk ' $0 !~/aeiou/ {print $0}' file1
```

6. EXPECTED OUTPUT:

```
$vi file1
```

```
pvwxyz sssx
```

```
hhhh kkkk
```

```
zzzzz bbbbb
```

```
$ awk -f prg.awk file1
```

```
3
```

Viva Questions

1.Which command is used to remove a directory?

- A. rd B. rmdir C. dldir D. rdir
E. None of the above

2.Which of the following keys is used to replace a single character with new text?

- A. S B. s C. r D. C
E. None of the above

3.Which command is used to extract specific columns from the file?

- A. cat B. cut C. grep D. paste

EXPERIMENT NO: 9

NAME OF THE EXPERIMENT: Write awk script to count no of characters, words, lines in a given file.

AIM: By using bash shell we will write awk script to count the number of lines, words, characters in a given file.

ALGORITHM

Step 1: create a file with 5-10 lines of data

Step 2: write an awk script

find the length of file

store it in x

step3: count the no of fields (NF), store it in y

step4: count the no of records (NR), store it in NR

step5: print NR,x,y

step6: stop

SOURCE CODE:

```
awk '{x+=length($0); y+=NF} end {print NR, x,y}' filename
```

EXPECTED OUTPUT

```
vi filename.txt
```

```
hai how r u
```

```
file thank u
```

```
bye
```

```
3 28 8
```

VIVA QUESTIONS:

1. How to find the last modified file or the newest file in a directory?

```
$ ls -lrt | grep ^- | awk 'END{print $NF}'
```

2. How to access the 10th command line argument in a shell script in Linux?

\$1 for 1st argument, \$2 for 2nd, etc... For 10th argument, \${10}, for 11th, \${11} and so on.

3. How to find the sum of all numbers in a file in Linux? \$ awk '{x+=\$0}END{print x}' file

4. How to delete a file which has some hidden characters in the file name?

Since the rm command may not be able to delete it, the easiest way to delete a file with some hidden characters in its name is to delete it with the find command using the inode number of the file.

```
$ ls -li
```


EXPERIMENT NO: 10

NAME OF THE EXPERIMENT: Write C program that makes a copy of a file using standard I/O and System calls.

AIM: By using bash shell we will write awk script to count the number of lines, words, characters in a given file.

ALGORITHM:

```
Step 1: define BUFSIZE 1024
        char buf[BUFSIZE];
step 2: if (argc!=3)
        printf("Usage: cp <src> <dst>\n");
        return 1;
step 3: src = open(argv[1]);
step 4: if (src==-1)
        print "Unable to open %s\n", argv[1]
        return 1;
step 5: creat(argv[2]);
        dst = open(argv[2]);
step 6: if (dst==-1)
        print Unable to create %s\n", argv[2]
        return 1;
step 7: while ((amount = read(src, buf, BUFSIZE))>0)
        write(dst, buf, amount);
step 8: close(src);
        close(dst);
```

SOURCE CODE:

```
#include "syscall.h"
#include "stdio.h"
#include "stdlib.h"
#define BUFSIZE 1024
char buf[BUFSIZE];
int main(int argc, char** argv) {
    int src, dst, amount;
    if (argc!=3) {
        printf("Usage: cp <src> <dst>\n");
        return 1;
    }
    src = open(argv[1]);
    if (src==-1) {
```

```

    printf("Unable to open %s\n", argv[1]);
    return 1;
}
creat(argv[2]);
dst = open(argv[2]);
if (dst==-1) {
    printf("Unable to create %s\n", argv[2]);
    return 1;
}
while ((amount = read(src, buf, BUFSIZE))>0) {
    write(dst, buf, amount);
}
close(src);
close(dst);
return 0;
}

```

EXPECTED OUTPUT:

```

[lakshmi@localhost labprograms]$ vi prg10.c
[lakshmi@localhost labprograms]$ cc prg10.c
[lakshmi@localhost labprograms]$ ./a.out sampletext2 file10
Data copied from sampletext2 to file10
[lakshmi@localhost labprograms]$

```

Viva Questions

1. Using the grep command, how can you display or print the entire file contents?

```
$ grep '.*' file
```

2. What is the difference between a local variable and environment variable in Linux?

A local variable is the one in which the scope of the variable is only in the shell in which it is defined. An environment variable has scope in all the shells invoked by the shell in which it is defined.

EXPERIMENT NO: 11

NAME OF THE EXPERIMENT: Implement in C the following UNIX commands using system calls.

A.cat B.mv

A.cat:

AIM: By using system calls in linux we will write the c program to implement cat command(cat command is used to display the content of file)

ALGORITHM:

Step 1: if argument count not equal to 2
Print enter correct no of arguments
Step 2: open the file in read mode
if not opened for reading
print error
if((fd=open(argv[1],4))<0)
printf("ERROR");
step 3: read the bytes from files
read(fd,buff,sizeof(buff))
step 4:print the data to the output
write(1,buff,n)
step 5:stop

SOURCE CODE

```
#include<stdio.h>
#include<sys/types.h>
#include<stdlib.h>
#include<fcntl.h>
#include<sys/stat.h>
int main(int argc,char *argv[])
{
    int fd,n;
    char buff[512];
    if(argc!=2)
        printf("ENTER CORRECT ARGUMENTS :");
    if((fd=open(argv[1],4))<0)
    {
        printf("ERROR");
        return 0;
    }
}
```

```
        while(n=read(fd,buff,sizeof(buff))>0)
            write(1,buff,n);
    }
```

EXPECTED OUTPUT:

```
[1828@localhost]$ vi catcommand.c
[1828@localhost]$ cat 123
hai
how
are
you
welcome
to
linux lab
[1828@localhost]$ cc catcommand.c
[1828@localhost]$ ./a.out 123
hai
how
are
you
welcome
to
linux lab
```

B.mv:

AIM: By using system calls in linux we will write the c program to implement mv command(mv command is used to rename the file or to move the content of one file to another file)

ALGORITHM:

```
Step 1: if argument count not equal to 3
        print enter correct no of arguments
Step 2: open the first file in read mode
        f1=open(argv[1],O_RDONLY)
step 3: create a new file with 0644 permissions
        f2=creat(argv[2],0644)
step 4: read the data from file1 and write it to file 2
        read(f1,buff,sizeof(buff))
        write(f2,buff,n)
step 5: remove first file
step 6: stop
```

SOURCE CODE:

```
#include<stdio.h>
#include<fcntl.h>
#include<sys/types.h>
#include<stdlib.h>
int main(int argc,char *argv[])
{
    char buff[512];
    int f1,f2,n;
    if(argc!=3)
    {
        printf("USAGE %s FROM TO : ",argv[0]);
        return 0;
    }
    if((f1=open(argv[1],O_RDONLY))<0)
        printf("ERROR IN OPENING");
    if((f2=creat(argv[2],0644))<0)
        printf("ERROR IN CREATING");
    while((n=read(f1,buff,sizeof(buff)))>0)
        if((write(f2,buff,n))<n)
            printf("WRITE ERROR");
    close(f2);
    remove(argv[1]);
}
```

EXPECTED OUTPUT:

```
[1828@localhost]$ vi mvcommand.c
[1828@localhost]$ cat unix
welcome
we are using linux
[1828@localhost]$ cc mvcommand.c
[1828@localhost]$ ./a.out unix 12345
[1828@localhost]$ cat 12345
welcome
we are using linux
```

Viva Questions:

1. What does the 'execute' permission in a directory stand for?

Without the execute permission on a directory, the user will not be able to traverse or in other words, do a "cd" to the directory.

2. How to find the total number of arguments in a shell script in Linux?

The shell special variable, \$# ,contains the total number of arguments passed to a shell script.

3 How to remove the Control-M character from a file in Linux?

\$ dos2unix file

4. In which file should a variable be set in order to make the setting permanent?

The variable should be set in the profile file to make the setting permanent. The appropriate profile depends on the default shell being set for the user..

5. What is a she-bang line in a shell script?

She-bang line in a shell script is the first line, if present. It starts with '#' and followed up with a full path of a shell. The shell specified indicates the shell in which this script will be run. The entry of she-bang is not mandatory, however, if present, should be the first line of the script. With a number of shells available and syntax being specific for a given shell, it is always good to specify the she-bang line in a shell script.

Experiment: 12.

Name of the Experiment: Write a C Program to list files in a directory

AIM: By using system calls in Linux we will write the c program to list all files in a current directory.

ALGORITHM:

Step 1: if argument count is 1

Then print correct no of arguments

Step 2: by using stat system call retstat=stat(argv[i], &fs);

Which gives the information about file

1) Name of file

2) Type of file 1)directory file

3) List all the files in a given directory with long format

Step 3: stop

Source Code:

```
#include<stdlib.h>
#include<stdio.h>
#include<string.h>
main(int argc, char *argv[])
{
char d[50];
if(argc==2)
{
bzero(d,sizeof(d));
strcat(d,"ls ");
strcat(d,argv[1]);
system(d);
}
else
printf("\nInvalid No. of inputs");
}
```

Expected Output:

```
$gcc exp12.c dir1
File1
File2
File3
```

Viva Questions:

1. What is an internal command in Linux?

Internal commands are also called shell built-in commands. Example: cd,fg. Since these are shell built-in, no process is created while executing these commands, and hence are considered to be much faster.

2. x and y are two variables containing numbers? How to add these 2 numbers?

```
$ expr $x + $y
```

3. How to add a header record to a file in Linux?

```
$ sed -i '1i HEADER' file
```

4. How to find the list of files modified in the last 30 mins in Linux?

```
$ find . -mmin -30
```

5. How to find the list of files modified in the last 20 days?

```
$ find . -mtime -20
```


Experiment No 13

NAME OF THE EXPERIMENT: Write C program to implement ls -l command.

AIM: By using system calls in Linux we will write the c program to implement ls -l command (which is used to long listing the files)

ALGORITHM

```
Step 1: void rec_open( char *dname, int lev )
        struct dirent *dbuf;
        DIR *dp;
        struct stat stbuf;
        char cwd[128];
step 2: if (i > 1 && dname[i - 1] == '/')
        dname[i - 1] = 0;
        getcwd( cwd, sizeof( cwd) );
step 3: dp = opendir( dname );
        if (dp == NULL)
            write( 1, "opendir ", 8 );
            perror( dname );
step 4: if (chdir( dname ) < 0)
        write( 1, "chdir ", 6 );
        perror( dname );
step 5: while ((dbuf = readdir( dp )))
        if (! strcmp( dbuf->d_name, "." ) || ! strcmp( dbuf->d_name, ".." ))
            continue;
step 6: for (i = 0; i < lev; ++i)
        printf( "\t|" );
step 7: if (stat( dbuf->d_name, &stbuf ) < 0)
        write( 1, "stat ", 5 );
        perror( dbuf->d_name );
step 8: switch (stbuf.st_mode & S_IFMT)
        case S_IFREG:
            print ( "--%s\n", dbuf->d_name );
            break;
        case S_IFDIR:
            print ( "\033[01;34m--%s \033[0m\n", dbuf->d_name ); //Display in Blue color
            rec_open( dbuf->d_name, lev + 1 );
            break;
        default:
step 9: closedir( dp );
        chdir( cwd );
```

SOURCE CODE:

```
#include <sys/types.h>
#include <dirent.h>
#include <sys/stat.h>
#include <stdio.h>
#include <string.h>

static char wd[128];

void rec_open( char *dname, int lev )
{
    struct dirent *dbuf;
    DIR *dp;
    struct stat stbuf;
    char cwd[128];
    int i = strlen( dname );
    if (i > 1 && dname[i - 1] == '/')
        dname[i - 1] = 0;
    getcwd( cwd, sizeof( cwd ) );
    // printf( "CWD = %s\n", cwd );
    dp = opendir( dname );
    if (dp == NULL) {
        write( 1, "opendir ", 8 );
        perror( dname );
        return;
    }
    if (chdir( dname ) < 0) {
        write( 1, "chdir ", 6 );
        perror( dname );
        return;
    }
    while ((dbuf = readdir( dp ))) {
        if (! strcmp( dbuf->d_name, "." ) || ! strcmp( dbuf->d_name, ".." ))
            continue;
        for (i = 0; i < lev; ++i)
            printf( "\\t|" );
        if (stat( dbuf->d_name, &stbuf ) < 0) {
            write( 1, "stat ", 5 );
            perror( dbuf->d_name );
            continue;
        }
        switch (stbuf.st_mode & S_IFMT) {
            case S_IFREG:
```

```

printf( "--%s\n", dbuf->d_name );
break;
case S_IFDIR:
printf( "\033[01;34m--%s \033[0m\n", dbuf->d_name ); //Display in Blue color
rec_open( dbuf->d_name, lev + 1 );
break;
default:
printf( "\n" );
}
}
closedir( dp );
chdir( cwd );
}
int main(int argc, char *argv[])
{
if (argc < 2)
rec_open( ".", 1 );
else
rec_open( argv[1], 1 );
return 0;
}

```

EXPECTED OUTPUT

```

[lakshmi@localhost ~]$ vi laxmi5.sh
[lakshmi@localhost ~]$ vi laxmi12.c
[lakshmi@localhost ~]$ cc laxmi12.c
[lakshmi@localhost ~]$ ./a.out xx
|--d
|--c

```

Viva Questions

1. How to find the list of files modified in the last 30 mins in Linux? \$ find . -mmin -30
2. How to find the list of files modified in the last 20 days? \$ find . -mtime -20
3. 4. How to print the contents of a file line by line in Linux? \$ while read line do
> echo \$line
> done < file

EXPERIMENT NO 14

NAME OF THE EXPERIMENT : Write C program to list every file in a directory, inode number and file name

AIM: By using system calls in linux we will write the c program to create a child process, And the parent process will parent and child processes.

ALGORITHM:

Step 1: open directory

```
DIR *opendir(char *dirname)
DIR *dp;
```

Step 2: if directory is opened

```
if ((fd = open(dirname, O_RDONLY, 0)) == -1
    || fstat(fd, &stbuf) == -1
    || (stbuf.st_mode & S_IFMT) != S_IFDIR
    || (dp = (DIR *) malloc(sizeof(DIR))) == NULL)
    return NULL;
```

Step 3: Displays the files present in directory

```
dp->fd = fd;
return dp;
```

step 4: for each file displaying inode number

```
struct stat statbuf;
if (stat(dirname(argv[1]), &statbuf) != -1)
    process_inode_number(statbuf.st_ino);
here inode number of each file will be displayed
```

step 5: stop

SOURCE CODE:

```
#include<stdlib.h>
#include<stdio.h>
#include<string.h>
main(int argc, char *argv[])
{
    char d[50];
    if(argc==2)
    {
        bzero(d,sizeof(d));
        strcat(d,"ls ");
        strcat(d,"-i ");
        strcat(d,argv[1]);
        system(d);
    }
    else
        printf("\nInvalid No. of inputs");
}
```

EXPECTED OUTPUT:

```
$vi prg3.c
$cc prg3.c labprograms
a.out      laxmi100.c laxmi4.sh  sampletext2 temp  yy
datafile.dat laxmi1.sh laxmi5.sh sampletext3 temp1
f3         laxmi2.sh prg10.c   sampletext4 xx
file2      laxmi3.sh sampletext1 sampletext5 xx.txt
```

Viva Questions

1. How to find the files modified exactly before 30minutes?

```
$ find . -mmin 30
```

2. A string contains a absolute path of a file. How to extract the filename alone from the absolute path in Linux?

```
$ x="/home/guru/temp/f1.txt"
$ echo $x | sed 's^.*/^'
```

3. How to find all the files created after a pre-defined date time, say after 10th April 10AM?

This can be achieved in 2 steps:

1. Create a dummy file with the time stamp, 10th April 10AM.
2. Find all the files created after this dummy file.

```
$ touch -t 1004101000 file
$ find . -newer file
```

4. How to print the contents of a file line by line in Linux?

```
$ while read line
> do
> echo $line
> done < file
```

5. The word "Unix" is present in many .txt files which is present across many files and also files present in sub directories. How to get the total count of the word "Unix" from all the .txt files?

```
$ find . -name *.txt -exec grep -c Unix '{}' \; | awk '{x+=$0;}END{print x}'
```

EXPERIMENT NO 15

Name of the Experiment .Write a C program to demonstrates redirection of standard output to a file ls>f1

AIM: By using system calls in linux we will write the c program to demonstrate the redirection operator content of ls command will copied to file f1

ALGORITHM:

Step 1: create a file pointer

```
FILE *fd;
```

Step 2:open the pipe for reading the data of ls -l command

```
fd=popen("ls -l","r");
```

step 3: read the data from pipe and store it in line buffer

```
while((fgets(line,200,fd))!=NULL)
```

```
print ("%s\n",line);
```

step 4: create a file

```
if((f1=creat("xx.txt",0644))<0)
```

```
print ERROR IN CREATING
```

step 5:read the data from line and store it to file

```
while((n=read(line,buff,sizeof(buff)))>0)
```

```
write(f1,line,n);
```

step 6:stop

Source Code:

```
#include<stdio.h>
```

```
int main()
```

```
{
```

```
int n,f1;
```

```
FILE *fd;
```

```
char line[200],buff[200];
```

```
fd=popen("ls -l","r");
```

```
while((fgets(line,200,fd))!=NULL)
```

```
printf("%s\n",line);
```

```
if((f1=creat("xx.txt",0644))<0)
```

```
printf("ERROR IN CREATING");
```

```
while((n=read(line,buff,sizeof(buff)))>0)
```

```
write(f1,line,n);
```

```
//printf("WRITE ERROR");
```

```
close(f1);
```

```
return 0;
```

```
}
```

Expected Output:

```
1828@localhost]$ vi lscommand.c
[1828@localhost]$ cc lscommand.c
Data of ls copied to f1
[1828@localhost]$ cat f1
cat unix
welcome
we are using linux
```

Viva Questions

1. The word "Unix" is present in many .txt files which is present across many files and also files present in sub directories. How to get the total count of the word "Unix" from all the .txt files?

```
$ find . -name *.txt -exec grep -c Unix '{}' \; | awk '{x+=$0;}END{print x}'
```

2. How to join every 2 lines in a file in Linux? \$ sed 'N;s/\n/' file

3. A shell script will ask for 3 inputs. The user will not be physically present to be able to give it manually. How can the script be run without having to manually give the input?

Put those 3 input values in a file, and make the script to read this file as input. For example:

Assume the 3 values to be : 3, 10 and 20: \$ cat file

```
3
10
20
```

EXPERIMENT NO 16

Name of the Experiment .Write a C programs to create a child process and allow the parent to display parent and child to display child on screen.

AIM: By using system calls in Linux we will write the c program to create a child process, And the parent process will parent and child processes.

ALGORITHM:

Step 1: start the main function

Step2: call the fork() function to create a child process

fork function returns 2 values

step 3: which returns 0 to child process

step 4:which returns process id to the parent process

step 5:stop

5. Source Code:

```
#include <stdio.h>
#include <stdlib.h>
int main(void)
{
    int pid;
    switch(pid = fork())
    {
        case -1:
            printf("fork error");
            exit(0);
            break; /* parent exits */
        case 0:
            printf("HI FRIENDS... IAM CHILD PROCESS....!\n");
            exit(0);
            break;
        default:
            printf("HI FRIENDS... IAM PARENT PROCESS....!\n");
            exit(0);
            break;
    }
}
```


6.OUTPUT:

```
[1828@localhost]$ vi parent_child.c
[1828@localhost]$ cc parent_child.c
[1828@localhost]$ ./a.out
HI FRIENDS... IAM PARENT PROCESS....!
HI FRIENDS... IAM CHILD PROCESS....!
```

7.Viva Questions:

- 1. How to get tomorrow's date in Linux?** `$ date -d "1 day"`
- 2. How to join all lines in a file using comma?** `$ paste -s -d, file`
- 3. How to join all lines in a file without any delimiter?** `$ paste -s --delimiter="" file`
- 4. How to join every 2 lines in a file in Linux?** `$ sed 'N;s/\n//' file`
- 5. A shell script will ask for 3 inputs. The user will not be physically present to be able to give it manually. How can the script be run without having to manually give the input?**
Put those 3 input values in a file, and make the script to read this file as input. For example:
Assume the 3 values to be : 3, 10 and 20: `$ cat file`
3
10
20

EXPERIMENT NO 17

Name of the Experiment. Write a C program to create Zombie process

AIM: By using system calls in Linux we will write the c program to create a Zombie process(the process which is left by the parent)

ALGORITHM:

Step 1:call fork function to create a child process

Step 2:if fork(>0)

Then creation of Zombie

By applying sleep function for 10 seconds

Step3: now terminate the child process

Step 4: exit status child process not reported to parent

Step 5: status any process which is zombie can known by

Applying ps(1) command

Step 6: stop

SOURCE CODE:

```
#include<stdio.h>
#include<stdlib.h>
int main()
{
    if(fork(>0)
    {
        printf("*****CREATION OF ZOMBIE*****\n");
        sleep(10);
        printf("AFTER 10 SECUNDS\n");
        printf("PARENT\n");
    }
}
```

EXPECTED OUTPUT:

```
[1828@localhost]$ vi zombie.c
```

```
[1828@localhost]$ cc zombie.c
```

```
[1828@localhost]$ ./a.out
```

```
*****CREATION OF ZOMBIE*****
```

```
AFTER 10 SECUNDS
```

```
PARENT
```

Viva Questions:

1.What is the kernel in UNIX?

Ans:Kernel is the name of Operating System in UNIX. It is the kernel Job to keep each process and user separate and to regulate to access the system hardware

2.What is system calls? Or How does user program interact with Kernel?

Ans:User programs interact with kernel through set of standard system calls. These system calls provide requested service by kernel. Services like open, close, read, write or execute file and changing accessibility of file and enable access of hardware devices.

EXPERIMENT NO 18

Name of the Experiment : Write a c program that illustrates how an orphan process is created.

AIM: By using system calls in Linux we will write the c program to create a orphan process
(the process whose parent is dead)

ALGORITHM:

Step 1: call the fork function to create the child process

Step 2:if (pid==0)

Then print child id and parent id

Step 3:Then sleep(10)

Print child id and parent id

Step 4: else

Print child id and parent id

Step 5:which gives the information of arphan process

Step 6:stop

SOURCE CODE:

```
#include<stdio.h>
#include<stdlib.h>
int main()
{
    int pid;
    pid=fork();
    if(pid==0)
    {
        printf("IAM CHILD PROCESS,MY PROCESS ID IS: %d",getpid());
        printf("\n THE CHILDS'S PARENT PROCESS ID IS : %d\n",getppid());
        sleep(10);
        printf("\n *****AFTER 10 SECUNDS*****");
        printf("\n IAM CHILD PROCESS,MY PROCESS ID IS: %d",getpid());
        printf("\n THE CHILDS'S PARENT PROCESS ID IS : %d\n",getppid());
    }
    else
    {
        printf("\nIAM PARENT PROCESS,MY PROCESS ID IS: %d",getpid());
        printf("\n THE PARENT'S PARENT PROCESS ID IS : %d\n",getppid());
    }
}
```

EXPECTED OUTPUT:

```
lakshmi@localhost:~  
[lakshmi@localhost ~]$ cc prg18.c  
[lakshmi@localhost ~]$ ./a.out  
IAM CHILD PROCESS,MY PROCESS ID IS: 4403  
THE CHILDS'S PARENT PROCESS ID IS : 4402  
  
IAM PARENT PROCESS,MY PROCESS ID IS: 4402  
THE PARENT'S PARENT PROCESS ID IS : 3830  
[lakshmi@localhost ~]$  
*****AFTER 10 SECUNDS*****  
IAM CHILD PROCESS,MY PROCESS ID IS: 4403  
THE CHILDS'S PARENT PROCESS ID IS :1
```

Viva Questions:

1.How to find the total number of a lines in a file in Linux?

\$ wc -l file | awk '{print \$1}'

2.How to print the first line or the header record in a file?

\$ head -1 file

3. How to replace all occurrences of "Unix" to "Linux" in a file?

\$ sed 's/Unix/Linux/g' file

4. How to make the above changes permanent in the file?

\$ sed -i 's/Unix/Linux/g' file

5. How to replace only the first occurrence of "Unix" to "Linux" in a string in Linux?

\$ sed 's/Unix/Linux/' file

6. How to replace only the second occurrence of "Unix" to "Linux" in a string in Linux?

\$ sed 's/Unix/Linux/2' file

EXPERIMENT NO 19

Name of the Experiment. : Write a C program that illustrates how to execute two commands concurrently with a command pipe

AIM: By using system calls in Linux we will write the c program to execute two commands at a time by using pipe system call

ALGORITHM:

Step 1:call the fork function to create child process

Step2 :close writing option

Step 3:copy the old file descriptor

Step 4:exec the command `ls -l`(long listing files)

 This process will be executed by child

Step 5: else

 Close reading option

 Copy old file descriptor

 exec the command `wc` t count no of lines, words, characters

step 6: both commands concurrently executed by pipe

step 7:stop

SOURCE CODE:

```
#include<stdio.h>
#include<fcntl.h>
main()
{
    int pfd[2],p;
    pipe(pfd);
    p=fork();
    if(p==0)
    {
        close(pfd[0]);
        close(1);
        dup(pfd[1]);
        execlp("ls","ls","-l",(char*)0);
    }
    else
    {
        close(pfd[1]);
        close(0);
        dup(pfd[0]);
        execlp("wc","wc",(char*)0);
    }
}
```

```
}  
}
```

Expected Output:

```
[lakshmi@localhost ~]$ cc prg19.c  
prg19.c: In function âmainâ:  
prg19.c:13: warning: incompatible implicit declaration of built-in  
prg19.c:20: warning: incompatible implicit declaration of built-in  
[lakshmi@localhost ~]$ ./a.out  
23      200      1193  
[lakshmi@localhost ~]$ ls -l  
total 92  
-rwxrwxr-x 1 lakshmi lakshmi 5156 Jul  2 13:58 a.out  
-rwxrwxr-- 1 lakshmi lakshmi  26 Jun 29 14:42 f1  
-rw-rw-r-- 1 lakshmi lakshmi  39 Jun 29 14:41 f2  
-rw-rw-r-- 1 lakshmi lakshmi  22 Jun 29 15:01 f3  
-rw-rw-r-- 1 lakshmi lakshmi  164 Jun 30 15:08 pg6.sh  
-rw-rw-r-- 1 lakshmi lakshmi 1727 Jul  2 11:49 prg12.c  
-rw-rw-r-- 1 lakshmi lakshmi  173 Jun 30 15:43 prg16.c  
-rw-rw-r-- 1 lakshmi lakshmi  239 Jul  2 13:28 prg17.c  
-rw-rw-r-- 1 lakshmi lakshmi  545 Jul  2 13:31 prg18.c  
-rw-rw-r-- 1 lakshmi lakshmi  242 Jul  2 13:55 prg19.c  
-rw-rw-r-- 1 lakshmi lakshmi  212 Jun 29 14:34 prg1.sh  
-rw-rw-r-- 1 lakshmi lakshmi  947 Jun 30 15:43 prg23.c  
-rw-rw-r-- 1 lakshmi lakshmi  203 Jun 29 14:58 prg2.sh  
-rw-rw-r-- 1 lakshmi lakshmi  234 Jun 29 15:49 prg3.sh  
-rw-rw-r-- 1 lakshmi lakshmi  159 Jun 29 15:47 prg4.sh  
-rw-rw-r-- 1 lakshmi lakshmi   77 Jun 30 12:25 prg66.sh  
-rw-rw-r-- 1 lakshmi lakshmi  234 Jun 30 15:04 prg6.sh  
-rw-rw-r-- 1 lakshmi lakshmi  159 Jun 30 15:09 prg7.sh  
-rw-rw-r-- 1 lakshmi lakshmi  211 Jun 29 14:15 prg.txt  
-rw-rw-r-- 1 lakshmi lakshmi   54 Jun 29 14:16 text.txt  
drwxrwxr-x 2 lakshmi lakshmi 4096 Jun 30 12:30 xx  
drwxrwxr-x 2 lakshmi lakshmi 4096 Jun 30 12:20 yy  
[lakshmi@localhost ~]$
```

Viva Questions:

1. How to make the above changes permanent in the file?
\$ sed -i 's/Unix/Linux/g' file
2. How to replace only the first occurrence of "Unix" to "Linux" in a string in Linux?
\$ sed 's/Unix/Linux/' file
3. How to replace only the second occurrence of "Unix" to "Linux" in a string in Linux?
\$ sed 's/Unix/Linux/2' file

EXPERIMENT NO 20

Name of the Experiment. : Write a C program that illustrates communication between 2 unrelated processes using named pipe (FIFO file)

AIM: By using system calls in linux we will write the c program to provide the communication between the 2 different processes by using named pipes

ALGORITHM:

Algorithm for server:

- 1.Start
- 2.Create a first named pipe by using mkfifo system call
Pipe1=mkfifo(NP1,0666).
- 3.if mkfifo returns -1 then print a message that error in creating the pipe.
- 4.Create a second named pipe by using mkfifo system call
Pipe2=mkfifo(NP2,0666).
- 5.if mkfifo returns -1 then print a message that error in creating the pipe.
- 6.Open the first pipe for reading by open system call by setting O-RDONLY
Fd=open(NP1,O-RDONLY)
7. Open the second pipe for writing by open system call by setting O-WRONLY
Fd=open(NP2,O-WRONLY)
- 8.read the data from the first pipe by using read system call
numread=Read(fd,buf,MAX-BUF-SIZE)
buf[numread]='\0'
- 9.print the data that we have read from pipe
- 10.convert the data to the upper case.
- 11.write the converted string back to second pipe by
write(fd,buf, strlen(buf))
- 12.stop.

Algorithm for client :

- 1.start
- 2.check whether the no of arguments specified were correct or not
- 3.if no of arguments are less then print error message
- 4.Open the first named pipe for writing by open system call by setting O-WRONLY
Fd=open(NP1,O-WRONLY)
5. .Open the second named pipe for reading by open system call by setting O-RDONLY
Fd=open(NP2,O-RDONLY)
6. write the data to the pipe by using write system call
write(fd,argv[1],strlen(argv[1]))
7. read the data from the first pipe by using read system call


```
        numread=Read(fd,buf,MAX-BUF-SIZE)
        buf[numread]='\0'
```

8. print the data that we have read from pipe

9.stop

SOURCE CODE:

```
#include<stdio.h>
#include<stdlib.h>
#include<sys/types.h>
#include<sys/stat.h>
#include<string.h>
#include<fcntl.h>
void server(int,int);
void client(int,int);
int main()
{
    int p1[2],p2[2],pid;
    pipe(p1);
    pipe(p2);
    pid=fork();
    if(pid==0)
    {
        close(p1[1]);
        close(p2[0]);
        server(p1[0],p2[1]);
        return 0;
    }
    close(p1[0]);
    close(p2[1]);
    client(p1[1],p2[0]);
    wait();
    return 0;
}
void client(int wfd,int rfd)
{
    int i,j,n;
    char fname[2000];
    char buff[2000];
    printf("ENTER THE FILE NAME :");
    scanf("%s",fname);
    printf("CLIENT SENDING THE REQUEST.....PLEASE WAIT\n");
    sleep(10);
    write(wfd,fname,2000);
```

```

        n=read(rfd,buff,2000);
        buff[n]='\0';
        printf("THE RESULTS OF CLIENTS ARE.....\n");
        write(1,buff,n);
    }
void server(int rfd,int wfd)
{
    int i,j,n;
    char fname[2000];
    char buff[2000];
    n=read(rfd,fname,2000);
    fname[n]='\0';
    int fd=open(fname,O_RDONLY);
    sleep(10);
    if(fd<0)
        write(wfd,"can't open",9);
    else
        n=read(fd,buff,2000);
        write(wfd,buff,n);
}

```

EXPECTED OUTPUT:

```

[1828@localhost]$ vi pipes.c
[1828@localhost]$ cat smec
hello
welcome to
smec
[1828@localhost]$ cc pipes.c
[1828@localhost]$ ./a.out
ENTER THE FILE NAME :smec
CLIENT SENDING THE REQUEST.....PLEASE WAIT THE RESULTS OF CLIENTS ARE.....
hello
welcome to
smec

```

Viva Questions

1. How to add leading zeros to every line in a file in Linux?

```
$ sed 's/^/0000/' file
```

2. How to add trailing zeros to every line in a file in Linux?

```
$ sed 's/$/00/' file
```

3. How to get yesterday's date in Linux?

```
$ date -d "1 day ago"
```

4. I have a file with SQL commands. How can I open a sqlplus session in Linux and run this SQL file?

```
$ sqlplus guru/unix11@XE @file.txt
```

where file.txt is the ASCII file containing the sql instructions.

5. The ps command will disclose the sqlplus connect string if any sqlplus session is ON. How to prevent the sqlplus connect string from appearing in the ps command in Linux?

While connecting to sqlplus, instead of connecting in the normal way, connect as below:

```
$ sqlplus /nolog
> connect guru/unix11@XE
```

6. How to rename a group of files from .txt to .exe in Linux?

```
for i in *.txt
do
x=`basename $i .txt`
mv $i $x.exe
done
.
```

EXPERIMENT NO 21

Name of the Experiment: Write a C program in which a parent writes a message to a pipe and the child reads the message.

AIM: By using system calls in Linux we will write the c program to implement parent and child communication (parent is going to writes the message and consumer is going to reads the message)

ALGORITHM:

Algorithm for parent:

- 1.Start
- 2.Create a named pipe by using mkfifo system call
Pipe1=mkfifo(NP1,0666)
- 3.if mkfifo returns -1 then print a message that error in creating the pipe
- 4.Open the pipe for reading by open system call by setting O-RDONLY
Fd=open(NP1,O-RDONLY)
- 5.read the data from the pipe by using read system call
numread=Read(fd,buf,MAX-BUF-SIZE)
- 6.print the data that we have read from pipe
- 7.convert the data to the upper case.
- 8.print the converted data
- 9.stop.

Algorithm for child:

- 1.start
- 2.check whether the no of arguments specified were correct or not
- 3.if no of arguments are less then print error message
- 4.Open the pipe for writing by open system call by setting O-WRONLY
Fd=open(NP1,O-WRONLY)
5. write the data to the pipe by using write system call
write(fd,argv[1],strlen(argv[1]))
- 6.stop

Source code: Child:

```
#include<stdio.h>
#include<unistd.h>
#include<fcntl.h>
```

```

#define MAXSIZE 10
#define FIFO_NAME    "myfifo"

main()
{

int fifoid;
int fd, n;
char *r;
system("clear");
r=(char *)malloc(sizeof(char)*MAXSIZE);
int open_mode=O_RDONLY;
if( (fd=open(FIFO_NAME, open_mode)) < 0 )
{
printf("\nError: Named pipe cannot be opened\n");
exit(0);
}
while(1)
{
    n=read(fd, r, MAXSIZE);
    if(n > 0)
        printf("\nConsumer read: %s", r);
}
} /*main close*/

```

parent program

```

#include<stdio.h>
#include<unistd.h>
#include<fcntl.h>
#define MAXSIZE 10
#define FIFO_NAME    "myfifo"
main()
{
int fifoid;
int fd, n;
char *w;
system("clear");
w=(char *)malloc(sizeof(char)*MAXSIZE);
int open_mode=O_WRONLY;
fifoid=mkfifo(FIFO_NAME, 0755);
if(fifoid==-1)
{
printf("\nError: Named pipe cannot be Created\n");

```

```

exit(0);
}
if( (fd=open(FIFO_NAME, open_mode)) < 0 )
{
printf("\nError: Named pipe cannot be opened\n");
exit(0);
}
while(1)
{
    printf("\nProducer :");
    fflush(stdin);
    read(0, w, MAXSIZE);
    n=write(fd, w, MAXSIZE);
    if(n > 0)
        printf("\nProducer sent: %s", w);
}
} /*main close*/

```

Expected Output:

```

$ cc -o producer producer.c      #first window
$ cc -o consumer consumer.c      # second window

```

```

$ ./producer #first window
$ ./consumer # second window
Producer:
Producer sent: hai #first window
Consumer read: hai # second window
Producer sent: good morning #first window
Consumer read: good morning # second window
Producer sent: welcome #first window
Consumer read: welcome # second window

```

Viva Questions

1.What are Pipes? Explain use of pipes. *Answer - A pipe is a chain of processes so that output of one process (stdout) is fed an input (stdin) to another.....*

2.Explain trap command; shift Command, getopt command of linux. *Answer - Trap command: controls the action to be taken by the shell when a signal is received.*

3.What Stateless Linux server? What feature it offers? *Answer - A stateless Linux server is a centralized server in which no state exists on the single workstations.*

EXPERIMENT NO 22

Name of the Experiment: Write a C program (sender.c) to create a message queue with read and write permissions to write 3 messages to it with different priority numbers.

AIM: By using system calls in linux we will write the c program to create a message queue with read and write permissions and write 3 messages with different priority numbers and receive the messages at other terminal

ALGORITHM:

- 1.Start
- 2.Declare a message queue structure

```
typedef struct msgbuf {  
    long mtype;  
    char mtext[MSGSZ];  
} message_buf;
```

- Mtype =0 Retrieve the next message on the queue, regardless of its *mtype*.
Positive Get the next message with an *mtype equal to* the specified *msgtyp*.
Negative Retrieve the first message on the queue whose *mtype* field is less than or equal to the absolute value of the *msgtyp* argument.

Usually mtype is set to 1

mtext is the data this will be added to the queue.

- 3.Create message queue and store result in msgflg

```
msgflg = IPC_CREAT | 0666
```

msgflg argument must be an octal integer with settings for the queue's permissions and control flags.

4. Get the message queue id for the "name" 1234, which was created by the server

```
key = 1234
```

5. if ((msqid = msgget(key, msgflg)) < 0)

Then print error

The *msgget()* function shall return the message queue identifier associated with the argument *key*.

6. otherwise

Print msgget succeeded: msqid

7. Now we will send message of type 1 through message queue

message_buf sbuf

sbuf.mtype=1

8.mtext contains data to be send through message queue

By using strcpy function we we will copy some message
Into mtext

strcpy(sbuf.mtext, "Did you get this?")

9. then find the length of text by using strlen function

buf_length = strlen(sbuf.mtext) + 1

10.now we will send message to message to the message queue

By using msgsnd function

#include <[sys/msg.h](#)>

int msgsnd(int *msqid*, const void **msgp*, size_t *msgsz*, int *msgflg*);

msgsnd(msqid, &sbuf, buf_length, IPC_NOWAIT)

msqid: message queue id

&sbuf: pointer to user defined structure

Buf_length :length of data

IPC_NOWAIT: returns error on wait

11.if msgsnd <0 return error

12.otherwise print message sent is sbuf.mext

13.stop

SOURCE CODE:

```
#include<stdio.h>
```

```
#include<sys/ipc.h>
```

```
#include<sys/msg.h>
```

```
#include<sys/types.h>
```

```
#include<stdlib.h>
```

```
#define SIZE 2000
```

```
main()
```

```
{
```

```
    int mfd,mfd2,mfd3;
```

```
    struct
```

```
    {
```

```
        double mtype;
```

```
        char mtext[2000];
```

```
    }s1,s2,s3;
```

```
    if((mfd=msgget(1000,IPC_CREAT|0666))== -1)
```

```
    {
```

```
        perror("msgget:");
```

```
        exit(1);
```

```
    }
```



```

s1.mtype=1;
sprintf(s1.mtext,"%s","Hi friends... My name is message1");
if(msgsnd(mfd,&s1,1000,0)==-1)
{
    perror("msgsnd");
    exit(1);
}

if((mfd2=msgget(1000,IPC_CREAT|0666))==1)
{
    perror("msgget:");
    exit(1);
}
s2.mtype=1;
sprintf(s2.mtext,"%s","Hi friends... My name is message2");
if(msgsnd(mfd2,&s2,1000,0)==-1)
{
    perror("msgsnd");
    exit(1);
}
}

if((mfd3=msgget(1000,IPC_CREAT|0666))==1)
{
    perror("msgget:");
    exit(1);
}
s3.mtype=1;
sprintf(s3.mtext,"%s","Hi friends... My name is message3");
if(msgsnd(mfd3,&s3,1000,0)==-1)
{
    perror("msgsnd");
    exit(1);
}
printf("Your message has been sent successfully...\n");
printf("Please visit another (receiver's) terminal...\n");
printf("Thank you.... For using LINUX");
}

```

EXPECTED OUTPUT:

```

[1828@localhost]$ cc sender1.c
[1828@localhost]$ ./a.out
Your message has been sent successfully...
Please visit another (receiver's) terminal...
Thank you.... For using LINUX

```

Viva Questions

1. What is difference between external command and internal command in UNIX?

Ans: In UNIX system, external commands resides in computer disks. For ex- /bin/ls is an external command. On other hand, internal command are built into the shell program. For ex- exit is an internal command.

2. How does the UNIX system indentify user?

Ans: The system identifies you by the user id and group id assigned to you by system administrator. You don't need to remember your user id

EXPERIMENT NO 23

Name of the Experiment: Write a C program (receiver.c) that receives the messages from message queue and display them.

AIM: By using system calls in linux we will write the c program to create a message queue with read and write permissions and write 3 messages with different priority numbers and receive the messages at other terminal

ALGORITHM:

- 1.Start
- 2.Declare a message queue structure

```
typedef struct msgbuf {  
    long mtype;  
    char mtext[MSGSZ];  
} message_buf;
```

- Mtype =0 Retrieve the next message on the queue, regardless of its *mtype*.
Positive Get the next message with an *mtype equal* to the specified *msgtyp*.
Negative Retrieve the first message on the queue whose *mtype* field is less than or equal to the absolute value of the *msgtyp* argument.

Usually mtype is set to 1

mtext is the data this will be added to the queue.

- 3.Get the message queue id for the "name" 1234, which was created by the server

key = 1234

- 4 if ((msqid = msgget(key, 0666 < 0))

Then print error

The *msgget()* function shall return the message queue identifier associated with the argument *key*.

5. Receive message from message queue by using *msgrcv* function

```
Int msgrcv(int msqid, void *msgp, size_t msgsz, long msgtyp, int msgflg);
```

```
#include <sys/msg.h>
```

```
(msgrcv(msqid, &rbuf, MSGSZ, 1, 0)
```

msqid: message queue id

&rbuf: pointer to user defined structure

MSGSZ: message size

Message type: 1

Message flag: The *msgflg* argument is a bit mask constructed by ORing together zero or more of the following flags: **IPC_NOWAIT** or **MSG_EXCEPT** or **MSG_NOERROR**

6.if msgrcv <0 return error

7.otherwise print message sent is sbuf.mext

8.stop

SOURCE CODE:

```
include<stdio.h>
#include<stdlib.h>
#include<sys/ipc.h>
#include<sys/msg.h>
#include<sys/types.h>
#define SIZE 40
main()
{
    int mfd,mfd2,mfd3;
    struct
    {
        long mtype;
        char mtext[6];
    }s1,s2,s3;
    if((mfd=msgget(1000,0))== -1)
    {
        perror("msgget");
        exit(1);
    }
    if(msgrcv(mfd,&s1,SIZE,0,IPC_NOWAIT|MSG_NOERROR)== -1)
    {
        perror("msgrcv");
        exit(1);
    }
    printf("Message from client is :%s\n",s1.mtext);

    if((mfd2=msgget(1000,0))== -1)
    {
        perror("msgget");
        exit(1);
    }
}
```

```

if(msgrcv(mfd2,&s2,SIZE,0,IPC_NOWAIT|MSG_NOERROR)==-1)
{
    perror("msgrcv");
    exit(1);
}
printf("Message from client is :%s\n",s2.mtext);
if((mfd3=msgget(1000,0))== -1)
{
    perror("msgget");
    exit(1);
}
if(msgrcv(mfd3,&s3,SIZE,0,IPC_NOWAIT|MSG_NOERROR)==-1)
{
    perror("msgrcv");
    exit(1);
}
printf("Message from sender is :%s\n",s3.mtext);
}

```

EXPECTED OUTPUT:

```

[1828@localhost]$ vi receiver1.c
[1828@localhost]$ cc receiver1.c
[1828@localhost]$ ./a.out
Message from sender is :Hi friends... My name is message1
Message from sender is :Hi friends... My name is message2
Message from sender is :Hi friends... My name is message3

```

Viva Questions:

1. After logging in to your account in Linux, you did "cd log". There was no "log" directory under the current directory, still the "cd" command traversed to a log directory under a different location? How it happened?

It is because the CDPATH variable is set.

2. How to zero pad a number in Linux?

Say, to zero pad a number to 4 places:

```

$ x=20
$ printf "%04d\n" $x

```

EXPERIMENT NO 24

Name of the Experiment: .Write a C program that illustrates suspending and resuming processes using signals.

AIM: By using system calls in Linux we will write the c program to illustrate the signals to suspend and resume the execution of process.

ALGORITHM:

Step 1: call the signal function to generate the signal
Step 2: execution of process will be started
Step 3: call alarm function to suspend the execution of current process
Step 4: then it will execute the signal function
Step 5: again the process will be resumed
Step 6: stop

Source code:

```
main()
{
int n;
if(signal(SIGALRM,sig_alarm)==SIG_ERR)
printf("Signal error");
alarm(5);
for(n=0;n<=15;n++)
printf("from for loop n=%d",n);

printf("main program terminated");

}
void sig_alarm(int signo)
{
printf("from sigalarm function");
}
```

Expected Output:

```
$cc alarm.c
$./a.out
```

```
from for loop n=0
from for loop n=1
from for loop n=2
```

from for loop n=3
from for loop n=4
from sigalarm function
from for loop n=5
from for loop n=6
.....
from for loop n=13
from for loop n=14
from for loop n=15

Viva Questions:

1. **How do you find out what's your shell? - echo \$SHELL**
2. **What's the command to find out today's date? - date**
3. **What's the command to find out users on the system? - who**
4. **How do you find out the current directory you're in? - pwd**
5. **How do you remove a file? - rm**
6. **How do you remove a <="" b=""> - rm -rf**

EXPERIMENT No: 25

Write a client and server programs in C for connection oriented communication between server and client process using Linux domain sockets to perform the following:

Client process sends a message to the server process. The server receives the message, reverses it and sends it back to the client. The client will then display the message to the standard output device.

Name of the Experiment: Write a Client-Server program using UNIX domain Sockets

AIM: By using system calls in Linux we will write the c program to implement client-server communication using UNIX domain sockets

ALGORITHM:

Sample UNIX server

Step 1: define NAME "socket"

Step 2: sock = socket(AF_UNIX, SOCK_STREAM, 0);

Step 3: if (sock < 0)

 perror("opening stream socket");

 exit(1);

step4: server.sun_family = AF_UNIX;

 strcpy(server.sun_path, NAME);

 if (bind(sock, (struct sockaddr *) &server, sizeof(struct sockaddr_un))) {

 perror("binding stream socket");

 exit(1);

step 5: print ("Socket has name %s\n", server.sun_path);

 listen(sock, 5);

step 6: for (;;) {

 msgsock = accept(sock, 0, 0);

 if (msgsock == -1)

 perror("accept");

 else do {


```

        bzero(buf, sizeof(buf));

        if ((rval = read(msgsock, buf, 1024)) < 0)

            perror("reading stream message");

        else if (rval == 0)

            else          print ("-->%s\n", buf);

    } while (rval > 0);

    close(msgsock);

}

close(sock);

unlink(NAME);

}

```

lines 11-14 Create the socket

lines 16-20 Name and bind the socket using file system name

lines 23-37 Start accepting connections, and start reading on the accepted connection

lines 39-40 These statements are not executed, because they follow an infinite loop. However, most ordinary programs will not run forever. In the UNIX domain it is necessary to tell the file system that one is through using NAME. In most programs one uses the call **unlink** as in line 40. Because the user has to kill this program, it is necessary to remove the name by a command from the shell.

Client code (UNIX domain)

This program connects to the socket named in the command line and sends a one line message to that socket. The form of the command line is:

Sample UNIX client

```

#define DATA "Half a league, half a league . . ."

main(argc, argv)

int argc;

char *argv[];

{

int sock;

```

```

struct sockaddr_un server;

char buf[1024];

if (argc < 2) {

printf("usage:%s <pathname>", argv[0]);

exit(1);

}

sock = socket(AF_UNIX, SOCK_STREAM, 0);

if (sock < 0) {

perror("opening stream socket");

exit(1);

}

server.sun_family = AF_UNIX;

strcpy(server.sun_path, argv[1]);

if (connect(sock, (struct sockaddr *) &server, sizeof(struct sockaddr_un)) < 0) {

close(sock);

perror("connecting stream socket");

exit(1);

}

if (write(sock, DATA, sizeof(DATA)) < 0)

perror("writing on stream socket");

close(sock);

}

```

lines 13-16 Check command-line arguments for validity

lines 17-21 Create socket

lines 22-28 Connect socket using name specified by command line
lines 29-31 Write data on the connected socket and close the socket before exiting the program

Expected Output:

Execution Steps:

2.

TCP

a) Client Server Application.

1. Compiling and running server.

```
[user@localhost week9]$ cc tcpserver.c
```

```
[user@localhost week9]$ mv a.out tcpserver
```

```
[user@localhost week9]$ ./tcpserver
```

Server:I am waiting-----Start of Main Loop

Connection from 127.0.0.1

enter the string

Server :Received Network Programming

Rev string is gnimmargorP krowteN

Finished Serving One Client

Server:I am waiting-----Start of Main Loop

2. Compiling and running client.

```
[user@localhost week9]$ cc tcpclient.c
```

```
[user@localhost week9]$ mv a.out tcpclient
```

```
[user@localhost week9]$ ./tcpclient 127.0.0.1 13153
```

enter sentence to end enter #Network Programming#

String : Network Programming sent to server

Viva Questions:

1. Define producer consumer problem
2. Define half duplex pipe
3. Define full duplex pipe
4. Define client server communication
5. Define IPC
6. Define UNIX domain sockets

EXPERIMENT No: 26

Write a client and server programs in C for connection oriented communication between server and client process using Internet domain socket to perform the following:

Client process sends a message to the server process. The server receives the message, reverses it and sends it back to the client. The client will then display the message to the standard output device.

Name of the Experiment: Write a Client-Server program using internet domain sockets.

AIM: By using system calls in Linux we will write the c program to provide the client server communication using internet domain sockets

ALGORITHM: server.c

```
#define MYPORT 13154 /*The port users will be connecting to*/
void readstring(int,char *);
int main(int C, char *V[] )
{
    int listensocket,connectionsocket,retbind;
    struct sockaddr_in serveraddress,cliaddr;
    socklen_t len;
    char buf[100],databuf[1024];

    listensocket = socket(AF_INET, SOCK_STREAM, 0 );
    if (listensocket < 0 )
    {
        perror("socket" );
        exit(1);
    }
    memset(&serveraddress, 0, sizeof(serveraddress) );
    serveraddress.sin_family = AF_INET;
    serveraddress.sin_port = htons(MYPORT);/*PORT NO*/
    serveraddress.sin_addr.s_addr = htonl(INADDR_ANY);/*ADDRESS*/
    retbind=bind(listensocket,(struct sockaddr*)&serveraddress,
    sizeof(serveraddress));
    /*Check the return value of bind for error*/
    if(-1==retbind)
    {
        perror("BIND ERROR\n");
        exit(1);
    }
    listen(listensocket,5);
    /*Beginning of the Main Server Processing Loop*/
```

```

for (;;)
{
printf("Server:I am waiting-----Start of Main Loop\n");
len=sizeof(cliaddr);
connectionsocket=accept(listensocket,
(struct sockaddr*)&cliaddr,&len);
if (connectionsocket < 0)
{
if (errno == EINTR)
printf("Interrupted system call ??");
continue;
}
printf("Connection from %s\n", inet_ntop(AF_INET,&cliaddr.sin_addr,buf,sizeof(buf)));
readstring(connectionsocket , databuf);
close(connectionsocket);
printf("Finished Serving One Client\n");
}
}

```

```

void readstring(
int connectionsocket,
/*Socket Descriptor*/
char *fname) /*Array , to be populated by the string from client*/
{
int pointer=0,n;
int len=0,a,b;
char rev[50],temp[50],temp1[50];
int k,i;
while ((n=read(connectionsocket,(fname + pointer),1024))>0)
{
pointer=pointer+n;
}
fname[pointer]='\0';
printf("enter the string\n");
printf("Server :Received  %s\n ",fname);
//strcpy(temp,fname);
k=strlen(fname);
//      for(k=0;temp[k]!=0;k++);
//      len=k;
a=0;
for(i=k-1;i>=0;i--)
temp[a++]=fname[i];
temp[a]='\0';
printf("\nrev is %s\n", temp);

```

```
}
```

Client:

```
#define MAXBUFFER 1024
void sendstring(int , char *);
int main( int C, char *V[] )
{
    Int sd,fd;
    char c;
    struct sockaddr_in serveraddress;
    char text[100];
    int i=0;
    sd = socket( AF_INET, SOCK_STREAM, 0 );
    if( sd < 0 ) {
        perror( "socket" );
        exit( 1 );
    }
    memset( &serveraddress, 0, sizeof(serveraddress) );
    serveraddress.sin_family = AF_INET;
    serveraddress.sin_port = htons(atoi(V[2]));//PORT NO
    serveraddress.sin_addr.s_addr = inet_addr(V[1]);//ADDRESS
    if (connect(sd,(struct sockaddr*)&serveraddress,
        sizeof(serveraddress))<0)
    {
        printf("Cannot Connect to server");
        exit(1);
    }
    printf("enter sentence to end enter #");
    while(1)
    {
        c=getchar();
        if(c=='#')
            break;
        text[i++]=c;
    }
    text[i]='\0';
    sendstring(sd,text);
    close(sd);
    return 0;
}

void sendstring(
int sd,
/*Socket Descriptor*/
```

```

char *fname) /*Array Containing the string */
{
int n , byteswritten=0 , written ;
char buffer[MAXBUFFER];
strcpy(buffer , fname);
n=strlen(buffer);
while (byteswritten<n)
{
written=write(sd , buffer+byteswritten,(n-byteswritten));
byteswritten+=written;
} printf("String : %s sent to server \n",buffer);
}

```

Expected Output:

Execution Steps:

2.

TCP

a) Client Server Application.

1.Compiling and running server.

[user@localhost week9]\$ cc tcpserver.c

[user@localhost week9]\$ mv a.out tcpserver

[user@localhost week9]\$./tcpserver

Server:I am waiting-----Start of Main Loop

Connection from 127.0.0.1

enter the string

Server :Received Network Programming

Rev string is gnimmargorP krowteN

Finished Serving One Client

Server:I am waiting-----Start of Main Loop

2. Compiling and running client.

[user@localhost week9]\$ cc tcpclient.c

[user@localhost week9]\$ mv a.out tcpclient

[user@localhost week9]\$./tcpclient 127.0.0.1 13153

enter sentence to end enter #Network Programming#

String : Network Programming sent to server

Viva Questions

1.[How does the linux file system work?](#)

Answer - Linux file structure is a tree like structure. It starts from the root directory, represented by '/', and then expands into sub-directories.....

2.[What are the process states in Linux?](#)

Answer - Process states in Linux.....

3.[What is a zombie?](#)

Answer - Zombie is a process state when the child dies before the parent process. In this case the structural information of the process is still in the process table.....

EXPERIMENT No: 27

Name of the Experiment: Write a C program that illustrates 2 processes communicating using shared memory.

AIM: By using system calls in Linux we will write the c program to implement to provide the communication using shared memory

ALGORITHM:

- 1.Start
- 2.Include header files required for the program are

```
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <unistd.h>
#include <string.h>
#include <errno.h>
```

- 3.Declare the variable which are required as

```
pid_t pid
int *shared /* pointer to the shm */
int shmid
```

- 4.Use shmget function to create shared memory

```
#include <sys/shm.h>
```

```
int shmget(key_t key, size_t size, int shmflg)
```

The shmget() function shall return the shared memory identifier associated with key
The argument key is equal to IPC_PRIVATE. so that the operating system selects the next available key for a newly created shared block of memory.

Size represents size of shared memory block

Shmflg shared memory permissions which are represented by octal integer

```
shmid = shmget(IPC_PRIVATE, sizeof(int), IPC_CREAT | 0666);
```

print the shared memory id

- 5.if fork()==0

Then

begin

```
shared = shmat(shmid, (void *) 0, 0)
```

```
print the shared variable(shared)
```

```
*shared=2
```

```
print *shared
```



```

    sleep(2)
    print *shared
end
6.else
begin
    shared = shmat(shmid, (void *) 0, 0)
    print the shared variable(shared)
    print *shared
    sleep(1)
    *shared=30
    printf("Parent value=%d\n", *shared);
    sleep(5)
    shmctl(shmid, IPC_RMID, 0)
end
7.stop.

```

Source code:

```

#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <unistd.h>
#include <string.h>
#include <errno.h>

int main(void) {

    pid_t pid;
    int *shared; /* pointer to the shm */
    int shmid;

    shmid = shmget(IPC_PRIVATE, sizeof(int), IPC_CREAT | 0666);
    printf("Shared Memory ID=%u",shmid);
    if (fork() == 0) { /* Child */
        /* Attach to shared memory and print the pointer */
        shared = shmat(shmid, (void *) 0, 0);
        printf("Child pointer  %u\n", shared);
        *shared=1;
        printf("Child value=%d\n", *shared);
        sleep(2);
        printf("Child value=%d\n", *shared);

    } else { /* Parent */
        /* Attach to shared memory and print the pointer */

```

```

        shared = shmat(shmid, (void *) 0, 0);
        printf("Parent pointer %u\n", shared);
        printf("Parent value=%d\n", *shared);
        sleep(1);
        *shared=42;
        printf("Parent value=%d\n", *shared);
        sleep(5);
        shmctl(shmid, IPC_RMID, 0);
    }
}

```

Expected Output:

```

[sampath@localhost ipc]$cc shared_mem.c
[sampath@localhost ipc]$ ./a.out
Shared Memory ID=65537Child pointer 3086680064
Child value=1
Shared Memory ID=65537Parent pointer 3086680064
Parent value=1
Parent value=42
Child value=42

```

Viva Questions

1. Define shared memory
2. What are file locking functions?
3. What are shared memory system calls?
4. Define internet domain sockets
5. Difference between internet and UNIX domain sockets.

References:

Books:

1. Advanced UNIX programming by B.Venkateswarlu PHI publications 3rd edition.
2. Unix System Programming using C++, T.Chan, PHI
3. UNIX Concepts and Applications, 4th Edition, Sumitabha Das, TMH.

WEBSITES

1. www.advancedlinuxprogramming.com/.../advanced-linux-programming.
2. fita.hua.edu.vn/pttien/Setups/.../beginning-linux-programming.pdf
3. www.cse.hcmut.edu.vn/~hungnq/courses/nap/alp.pdf
4. <https://sites.google.com/site/linuxlabjntu09/home>
5. <http://youtube.com/linux>
6. <http://www.nptel.com/computerscience/Linuxprogramming>
7. <http://nptel.iitm.ac.in/courses.php?disciplineId=106>
8. <http://manuals.bioinformatics.ucr.edu/home/linux-basics>