VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"Jnana Sangama", Belgaum -590014, Karnataka.

LAB REPORT
On

Unix Shell Programming

Submitted by

RAKESH JADHAV(1BM20CS122)

in partial fulfillment for the award of the degree of
BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING
(Autonomous Institution under VTU)
BENGALURU-560019

Oct 2022 to Feb 2023

B. M. S. College of Engineering,

Bull Temple Road, Bangalore 560019

(Affiliated To Visvesvaraya Technological University, Belgaum)

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the Lab work entitled **"Unix Shell Programming"** carried out by **RAKESH JADHAV(1BM20CS122)**, who is Bonafede student of **B. M. S. College of Engineering**. It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2022-23. The Lab report has been approved as it satisfies the academic requirements in respect of a **Unix Shell Programming - (20CS5PCUSP)** work prescribed for the said degree.

Dr. Latha N R
Assistant Professor
Department of CSE
BMSCE,Bengaluru

Dr.JYOTHI S NAYAK
Professor & Head of
Department of CSE
BMSCE,Bengaluru

Index

System Programs

Program 1

Aim of the program: A shell script which displays the number of words and lines in a file.

Program:

```
#!/bin/bash

ls

echo Enter a filename :
read fname

echo -n Number of lines is :
wc -l $fname | head -c 3

echo

echo -n Number of words is :
wc -w $fname | head -c 3
echo
```

Output:

```
rakj13@RakeshJadhav:~/UnixLab$ sh 1usp.sh
1usp.sh  2usp.sh  3usp.sh  4usp.sh  5usp.sh  6usp.sh  7usp.sh  rsj
Enter a filename :
rsj
Number of lines is :15
Number of words is :48
```

Program 2

Aim of the program: Shell script which displays list of files in a given directory.

Program:

```
#!/bin/bash

echo "Enter the directory path: "
read directory

if [ ! -d "$directory" ]; then
  echo "$directory is not a directory."
  exit 1
fi

files=$(ls "$directory")

echo "Files in $directory:"
echo "$files"
```

Output:

```
rakj13@RakeshJadhav:~$ sh 2usp.sh
Enter the directory path:
UnixLab
Files in UnixLab:
1usp.sh
2usp.sh
rsj
```

Program 3

Aim of the program: Shell script to display area of a circle

Program:

```
#!/bin/bash

echo "Enter the radius: "
read radius

area=$(echo "3.141592653589 * ($radius * $radius)" | bc -l)

echo "The area of the circle is: $area"
```

Output:

```
rakj13@RakeshJadhav:~/UnixLab$ sh 3usp.sh
Enter the radius:
10
The area of the circle is: 314.159265358900
rakj13@RakeshJadhav:~/UnixLab$
```

Program 4

Aim of the program: Shell script to display largest of three numbers

Program:

```
#!/bin/bash

echo "Enter the first number: "
read num1

echo "Enter the second number: "
read num2

echo "Enter the third number: "
read num3

if [ "$num1" -gt "$num2" ] && [ "$num1" -gt "$num3" ]; then
  largest=$num1
elif [ "$num2" -gt "$num1" ] && [ "$num2" -gt "$num3" ]; then
  largest=$num2
else
  largest=$num3
fi

echo "The largest number is: $largest"
```

Output:

```
rakj13@RakeshJadhav:~/UnixLab$ sh 4usp.sh
Enter the first number:
13
Enter the second number:
100
Enter the third number:
9
The largest number is: 100
```

Program 5

Aim of the program: Shell script to display whether the entered number is a positive, negative or a zero number

Program:

```
#!/bin/bash

echo "Enter a number: "
read num

if [ "$num" -gt  0 ]; then
  echo "$num is a positive number."
elif [ "$num" -lt 0 ]; then
  echo "$num is a negative number."
else
  echo "$num is zero."
fi
```

Output:

```
rakj13@RakeshJadhav:~/UnixLab$ sh 5usp.sh
Enter a number:
5
5 is a positive number.
rakj13@RakeshJadhav:~/UnixLab$ sh 5usp.sh
Enter a number:
-9
-9 is a negative number.
rakj13@RakeshJadhav:~/UnixLab$ sh 5usp.sh
Enter a number:
0
0 is zero.
rakj13@RakeshJadhav:~/UnixLab$
```

Program 6

Aim of the program: Shell script to display whether the entered arguments are equal or not

Program:

```
#!/bin/bash

if [ "$#" -ne 2 ]; then
  echo "Error: Two arguments are required."
  exit 1
fi

arg1="$1"
arg2="$2"

if [ "$arg1" = "$arg2" ]; then
  echo "The arguments are equal."
else
  echo "The arguments are not equal."
fi
```

Output:

```
rakj13@RakeshJadhav:~/UnixLab$ sh 6usp.sh rakesh rakesh
The arguments are equal.
rakj13@RakeshJadhav:~/UnixLab$ sh 6usp.sh 10 20
The arguments are not equal.
```

Program 7

Aim of the program: Shell script to display whether the entered year is a leap year or not

Program:

```
#!/bin/bash

echo "Enter a year: "
read year

if [ "$((year % 4))" -eq 0 ]; then
  if [ "$((year % 100))" -eq 0 ]; then
    if [ "$((year % 400))" -eq 0 ]; then
      echo "$year is a leap year."
    else
      echo "$year is not a leap year."
    fi
  else
    echo "$year is a leap year."
  fi
else
  echo "$year is not a leap year."
fi
```

Output:

```
rakj13@RakeshJadhav:~/UnixLab$ sh 7usp.sh
Enter a year:
2020
2020 is a leap year.
rakj13@RakeshJadhav:~/UnixLab$ sh 7usp.sh
Enter a year:
2021
2021 is not a leap year.
rakj13@RakeshJadhav:~/UnixLab$
```

Program 8

Aim of the program: Shell script to display the gross salary of an employee

Program:

```
#!/bin/bash
echo -n "Enter the basic pay : "
read b
da=`echo "scale=4; $b * 0.1" | bc`
hra=`echo "scale=4; $b * 0.2" | bc`
echo "DA : $da"
echo "HRA : $hra"
echo "GROSS : `echo "scale=4; $b + $da + $hra" | bc`
```

Output:

```
rakj13@RakeshJadhav:~/UnixLab$ sh 8usp.sh
Enter the basic pay : 10000
DA : 1000.0
HRA : 2000.0
GROSS : 13000.0
rakj13@RakeshJadhav:~/UnixLab$
```

Program 9

Aim of the program: Shell script to convert from fahrenheit to celsius

Program:

```
#!/bin/bash
echo -n "Enter the temperature (in deg F) : "
read t
echo "scale=4; ($t - 32)*5/9" | bc
```

Output:

```
rakj13@RakeshJadhav:~/UnixLab$ sh 9usp.sh
Enter the temperature (in deg F) : 25
-3.8888
rakj13@RakeshJadhav:~/UnixLab$ sh 9usp.sh
Enter the temperature (in deg F) : 100
37.7777
rakj13@RakeshJadhav:~/UnixLab$
```

Program 10

Aim of the program: Shell script to perform arithmetic operations using CASE statements

Program:

```
#!/bin/bash

read -p "Enter the first number: " num1
read -p "Enter the second number: " num2
read -p "Enter the operation (+, -, x, /): " operation

result=0

case $operation in
  "+") result=$(echo $num1 + $num2 | bc) ;;
  "-") result=$(echo $num1 - $num2 | bc) ;;
  "x") result=$(echo $num1 \* $num2 | bc) ;;
  "/") result=$(echo "scale=2; $num1 / $num2" | bc) ;;
  *) echo "Invalid operation" ;;
esac

echo "Result: $result"
```

Output:

```
rakj13@RakeshJadhav:~/UnixLab$ sh 10usp.sh
Enter the first number: 20
Enter the second number: 10
Enter the operation (+, -, x, /): +
Result: 30
rakj13@RakeshJadhav:~/UnixLab$ sh 10usp.sh
Enter the first number: 23
Enter the second number: 5
Enter the operation (+, -, x, /): /
Result: 4.60
rakj13@RakeshJadhav:~/UnixLab$
```

Program 11

Aim of the program: Shell script to find the factorial of a number

Program:

```
#!/bin/bash

read -p "Enter a number: " num

factorial=1
i=1

while [ $i -le $num ]
do
  factorial=$(echo $factorial \* $i | bc)
  i=$(($i + 1))
done

echo "Factorial of $num: $factorial"
```

Output:

```
rakj13@RakeshJadhav:~/UnixLab$ sh 11usp.sh
Enter a number: 13
Factorial of 13: 6227020800
rakj13@RakeshJadhav:~/UnixLab$ sh 11usp.sh
Enter a number: 5
Factorial of 5: 120
rakj13@RakeshJadhav:~/UnixLab$
```

Program 12

Aim of the program: Shell script to find the sum of even number

Program:

```
#!/bin/bash

read -p "Enter a number: " num

sum=0
i=2

while [ $i -le $num ]
do
  sum=$(echo $sum + $i | bc)
  i=$(($i + 2))
done

echo "Sum of even numbers up to $num: $sum"
```

Output:

```
rakj13@RakeshJadhav:~/UnixLab$ nano 12usp.sh
rakj13@RakeshJadhav:~/UnixLab$ sh 12usp.sh
Enter a number: 25
Sum of even numbers up to 25: 156
rakj13@RakeshJadhav:~/UnixLab$
```

Program 13

Aim of the program: Shell script to find the power of a number

Program:

```
echo -n "Enter a number : "
read n

echo -n "Enter the exponent : "
read exp

res=1

while [ $exp -gt 0 ]
do
res=`expr $res \* $n`
exp=`expr $exp - 1`
done

echo "Result is : $res"
```

Output:

```
rakj13@RakeshJadhav:~/UnixLab$ sh 13usp.sh
Enter a number : 25
Enter the exponent : 2
Result is : 625
rakj13@RakeshJadhav:~/UnixLab$
```

Program 14

Aim of the program: Shell script to find the sum of 'N' numbers

Program:

```
#!/bin/bash
echo -n "Enter a number : "
read num
ans=0
i=$num
while [ $i -gt 0 ]
do
ans=`expr $i + $ans`
i=`expr $i - 1`
done
echo "Sum of $num numbers is : $ans"
```

Output:

```
rakj13@RakeshJadhav:~/UnixLab$ sh 14usp.sh
Enter a number : 25
Sum of 25 numbers is : 325
```

Program 15

Aim of the program: Shell script to print all combinations of ‘1 2 3’

Program:

```
#!/bin/bash

numbers="1 2 3"

for i in $numbers; do
  for j in $numbers; do
    for k in $numbers; do
      echo "$i $j $k"
    done
  done
done
```

Output:

```
rakj13@RakeshJadhav:~/UnixLab$ sh 15usp.sh
1 1 1
1 1 2
1 1 3
1 2 1
1 2 2
1 2 3
1 3 1
1 3 2
1 3 3
2 1 1
2 1 2
2 1 3
2 2 1
2 2 2
2 2 3
2 3 1
2 3 2
2 3 3
3 1 1
3 1 2
3 1 3
3 2 1
3 2 2
3 2 3
3 3 1
3 3 2
3 3 3
```

Program 16

Aim of the program: Shell script to find GCD and LCM of the entered number

Program:

```
#!/bin/bash

gcd() {
  if [ $2 -eq 0 ]; then
    echo $1
  else
    gcd $2 $(($1 % $2))
  fi
}

lcm() {
  echo $(($1 * $2 / $(gcd $1 $2)))
}

read -p "Enter first number: " num1
read -p "Enter second number: " num2

gcd_result=$(gcd $num1 $num2)
lcm_result=$(lcm $num1 $num2)

echo "GCD of $num1 and $num2: $gcd_result"
echo "LCM of $num1 and $num2: $lcm_result"
```

Output:

```
rakj13@RakeshJadhav:~/UnixLab$ sh 16usp.sh
Enter 2 numbers : 25 27
GCD(25,27)=1
LCM(25,27)=675
rakj13@RakeshJadhav:~/UnixLab$ sh 16usp.sh
Enter 2 numbers : 16 24
GCD(16,24)=8
LCM(16,24)=48
rakj13@RakeshJadhav:~/UnixLab$
```

Program 17

Aim of the program: Shell script to check the number of line, words and characters on an entered file

Program:

```
#!/bin/bash
echo "Enter the name of the file:" \c
read name
echo "The number of lines are:"
wc -l $name
echo "The number of words are:"
wc -w $name
echo "The number of characters are:"
wc -c $name
```

Output:

```
rakj13@RakeshJadhav:~/UnixLab$ sh 18usp.sh
Enter the name of the file:
rsj
The number of lines are:
15 rsj
The number of words are:
48 rsj
The number of characters are:
288 rsj
```

Program 18

Aim of the program: Shell script to display count, sum of positive number and sum of negative numbers separately on an entered number list

Program:

```
#!/bin/bash

read -p "Enter the number of elements: " n

positive_sum=0
negative_sum=0
positive_count=0
negative_count=0
i=1

while [ $i -le $n ]; do
  read -p "Enter element $i: " num
  if [ $num -lt 0 ]; then
    negative_sum=$(($negative_sum + $num))
    negative_count=$(($negative_count + 1))
  else
    positive_sum=$(($positive_sum + $num))
    positive_count=$(($positive_count + 1))
  fi
  i=$(($i + 1))
done

echo "Positive count: $positive_count"
echo "Negative count: $negative_count"
echo "Sum of positive numbers: $positive_sum"
echo "Sum of negative numbers: $negative_sum"
```

Output:

```
rakj13@RakeshJadhav:~/UnixLab$ sh 19usp.sh
Enter the number of elements: 10
Enter element 1: 45
Enter element 2: 75
Enter element 3: 78
Enter element 4: 10
Enter element 5: -9
Enter element 6: -6
Enter element 7: -85
Enter element 8: -6
Enter element 9: 8
Enter element 10: 9
Positive count: 6
Negative count: 4
Sum of positive numbers: 225
Sum of negative numbers: -106
```

Program 19

Aim of the program: Shell script to find the sum of the last two prime numbers before the entered number

Program:

```
#!/bin/bash

read -p "Enter a number N: " n

count=0
sum=0
i=$(($n - 1))

while [ $count -lt 2 ]; do
  factor=2
  flag=0
  while [ $((i % factor)) -ne 0 ] && [ $factor -lt $i ]; do
    factor=$(($factor + 1))
  done
  if [ $factor -eq $i ]; then
    count=$(($count + 1))
    sum=$(($sum + $i))
  fi
  i=$(($i - 1))
done

echo "Sum of last two prime numbers before $n: $sum
```

Output:

```
rakj13@RakeshJadhav:~/UnixLab$ sh 20usp.sh
Enter a number N: 13
Sum of last two prime numbers before 13: 18
rakj13@RakeshJadhav:~/UnixLab$ sh 20usp.sh
Enter a number N: 50
Sum of last two prime numbers before 50: 90
rakj13@RakeshJadhav:~/UnixLab$
```

Program 20

Aim of the program: Shell script to print a pattern with the specified number of stars

Program:

```
#!/bin/sh

echo "Enter the number of rows: \c"
read n

k=1
i=1
l=1

while [ $i -ge 1 ]
do
   j=1
   k=1
   while [ $j -ge 1 ]
   do
      if [ $j -gt $(( $n - $i )) ]; then
         echo "*\c"
      else
         echo " \c"
      fi
      if [ $j -eq $n ]; then
         k=-1
      fi
      j=$(( $j + $k ))
   done
   echo ""
   if [ $i -eq $n ]; then
      l=-1
   fi
   i=$(( $i + $l ))
done
```

Output:

```
rakj13@RakeshJadhav:~/UnixLab$ sh 23usp.sh
Enter the number of rows: 5
    *
   ***
  *****
 *******
*********
 *******
  *****
   ***
    *
rakj13@RakeshJadhav:~/UnixLab$
```

System Programs

Program 1

Aim of the program: A program that outputs the given contents of its environment

Program:

```
#include <stdio.h>

int main(int argc, char *argv[], char * envp[])
{
    int i;
    for (i = 0; envp[i] != NULL; i++)
        printf("\n%s", envp[i]);
    getchar();
    return 0;
}
```

Output:

```
rakj13@RakeshJadhav:~/UnixLab$ gcc sp1.c
rakj13@RakeshJadhav:~/UnixLab$ ./a.out

SHELL=/bin/bash
SESSION_MANAGER=local/RakeshJadhav:@/tmp/.ICE-unix/2381,unix/RakeshJadhav:/tmp/.ICE-unix/2381
QT_ACCESSIBILITY=1
COLORTERM=truecolor
XDG_CONFIG_DIRS=/etc/xdg/xdg-ubuntu:/etc/xdg
SSH_AGENT_LAUNCHER=gnome-keyring
XDG_MENU_PREFIX=gnome-
GNOME_DESKTOP_SESSION_ID=this-is-deprecated
LANGUAGE=en_IN:en
GNOME_SHELL_SESSION_MODE=ubuntu
SSH_AUTH_SOCK=/run/user/1000/keyring/ssh
XMODIFIERS=@im=ibus
DESKTOP_SESSION=ubuntu
GTK_MODULES=gail:atk-bridge
PWD=/home/rakj13/UnixLab
LOGNAME=rakj13
XDG_SESSION_DESKTOP=ubuntu
XDG_SESSION_TYPE=wayland
SYSTEMD_EXEC_PID=2442
XAUTHORITY=/run/user/1000/.mutter-Xwaylandauth.MVPEZ1
HOME=/home/rakj13
USERNAME=rakj13
IM_CONFIG_PHASE=1
LANG=en_IN
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:or=40;31;01:mi=00:su=37;41:sg=30;43:ca=30;41:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tgz=01;31:*.arc=01
;31:*.arj=01;31:*.taz=01;31:*.lha=01;31:*.lz4=01;31:*.lzh=01;31:*.lzma=01;31:*.tlz=01;31:*.txz=01;31:*.tzo=01;31:*.t7z=01;31:*.zip=01;31:*.z=01;31:*.dz=01;31:*.gz=01;31:*.lrz=01;31:*.lz=01;31:*.lzo=01;31
:*.xz=01;31:*.zst=01;31:*.tzst=01;31:*.bz2=01;31:*.bz=01;31:*.tbz=01;31:*.tbz2=01;31:*.tz=01;31:*.deb=01;31:*.rpm=01;31:*.jar=01;31:*.war=01;31:*.ear=01;31:*.sar=01;31:*.rar=01;31:*.alz=01;31:*.ace=01;31
:*.zoo=01;31:*.cpio=01;31:*.7z=01;31:*.rz=01;31:*.cab=01;31:*.wim=01;31:*.swm=01;31:*.dwm=01;31:*.esd=01;31:*.jpg=01;35:*.jpeg=01;35:*.mjpg=01;35:*.mjpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=0
1;35:*.ppm=01;35:*.tga=01;35:*.xbm=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.png=01;35:*.svg=01;35:*.svgz=01;35:*.mng=01;35:*.pcx=01;35:*.mov=01;35:*.mpg=01;35:*.mpeg=01;35:*.m2v=01;35:*.mkv=01;35:*.w
ebm=01;35:*.webp=01;35:*.ogm=01;35:*.mp4=01;35:*.m4v=01;35:*.mp4v=01;35:*.vob=01;35:*.qt=01;35:*.nuv=01;35:*.wmv=01;35:*.asf=01;35:*.rm=01;35:*.rmvb=01;35:*.flc=01;35:*.avi=01;35:*.fli=01;35:*.flv=01;35:
*.gl=01;35:*.dl=01;35:*.xcf=01;35:*.xwd=01;35:*.yuv=01;35:*.cgm=01;35:*.emf=01;35:*.ogv=01;35:*.ogx=01;35:*.aac=00;36:*.au=00;36:*.flac=00;36:*.m4a=00;36:*.mid=00;36:*.midi=00;36:*.mka=00;36:*.mp3=00;36:
*.mpc=00;36:*.ogg=00;36:*.ra=00;36:*.wav=00;36:*.oga=00;36:*.opus=00;36:*.spx=00;36:*.xspf=00;36:
XDG_CURRENT_DESKTOP=ubuntu:GNOME
VTE_VERSION=6800
WAYLAND_DISPLAY=wayland-0
GNOME_TERMINAL_SCREEN=/org/gnome/Terminal/screen/14d54195_af54_4d08_b362_b7b5be5a2f49
GNOME_SETUP_DISPLAY=:1
LESSCLOSE=/usr/bin/lesspipe %s %s
XDG_SESSION_CLASS=user
TERM=xterm-256color
LESSOPEN=| /usr/bin/lesspipe %s
USER=rakj13
GNOME_TERMINAL_SERVICE=:1.129
DISPLAY=:0
SHLVL=1
QT_IM_MODULE=ibus
```

Program 2

Aim of the program: A program to emulate the Unix Line Command

Program:

```
#include<stdio.h>
#include<sys/types.h>
#include<unistd.h>
#include<string.h>
int main(int argc, char * argv[])
{
printf("count of arguments:%d\n",argc);
if(argc < 3 || argc > 4 || (argc == 4 && strcmp(argv[1],"-s")))
{
printf("Usage: ./a.out [-s] <org_file> <new_link>\n");
return 1;
}
if(argc == 4)
{
if((symlink(argv[2], argv[3])) == -1)
printf("Cannot create symbolic link\n") ;
else
printf("Symbolic link created\n") ;
}
else
{
if((link(argv[1], argv[2])) == -1)
printf("Cannot create hard link\n") ;
else
printf("Hard link created\n") ;
}
return 0;
```

}

Output:

```
rakj13@RakeshJadhav:~/UnixLab$ gcc sp2.c
rakj13@RakeshJadhav:~/UnixLab$ ./a.out -s rsj 1.txt
count of arguments:4
Symbolic link created
rakj13@RakeshJadhav:~/UnixLab$ ./a.out -s rsj 1.txt 2.txt
count of arguments:5
Usage: ./a.out [-s] <org_file> <new_link>
rakj13@RakeshJadhav:~/UnixLab$
```

Program 3

Aim of the program: A program POSIX compliant program that prints the POSIX defined configuration options supported on any given system using feature test macro

Program:

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

int main() {
  #ifdef _POSIX_JOB_CONTROL
    printf("System supports job control\n");
  #else
    printf("System does not support job control\n");
  #endif

  #ifdef _POSIX_SAVED_IDS
    printf("System supports saved set-UID and saved set-GID\n");
  #else
    printf("System does not support saved set-UID and saved set-GID\n");
  #endif

  #ifdef _POSIX_CHOWN_RESTRICTED
    printf("chown_restricted option is %d\n", _POSIX_CHOWN_RESTRICTED);
  #else
    printf("System does not support chown_restricted option\n");
  #endif

  #ifdef _POSIX_NO_TRUNC
    printf("Pathname trunc option is %d\n", _POSIX_NO_TRUNC);
  #else
    printf("System does not support system-wide pathname trunc option\n");
```

```
#endif

#ifdef _POSIX_VDISABLE
  printf("Disable character for terminal files is %d\n", _POSIX_VDISABLE);
#else
  printf("System does not support _POSIX_VDISABLE\n");
#endif

return 0;
}
```

Output:

```
rakj13@RakeshJadhav:~/UnixLab$ gcc sp2.c
rakj13@RakeshJadhav:~/UnixLab$ ./a.out -s rsj 1.txt
count of arguments:4
Symbolic link created
rakj13@RakeshJadhav:~/UnixLab$ ./a.out -s rsj 1.txt 2.txt
count of arguments:5
Usage: ./a.out [-s] <org_file> <new_link>
rakj13@RakeshJadhav:~/UnixLab$
```

Program 4

Aim of the program: Write a program which demonstrates Interprocess Communication between a reader process and a writer process. Use mkfifo, open, read, write and close apis in your program.

Program:

```
#include
#include
 #include
#include
#include
#include
#include
int main(int argc, char* argv[])
{
int fd;
char buf[256];
if(argc != 2 && argc != 3)

{
   printf("Usage: %s <file> [<arg>]\n",argv[0]);
   return 1;
}

if(mkfifo(argv[1],S_IFIFO | S_IRWXU | S_IRWXG | S_IRWXO) ==
-1)
{
   printf("Error creating named pipe %s: %s\n", argv[1],
strerror(errno));
   return 1;
}

if(argc == 2)
{
   fd = open(argv[1], O_RDONLY | O_NONBLOCK);
   if(fd == -1)
   {
      printf("Error opening named pipe %s: %s\n", argv[1],
strerror(errno));
      return 1;
```

```
    }
    while(read(fd, buf, sizeof(buf)) > 0)
        printf("%s",buf);
}
else
{
    fd = open(argv[1], O_WRONLY);
    if(fd == -1)
    {
        printf("Error opening named pipe %s: %s\n", argv[1],
strerror(errno));
        return 1;
    }
    if(write(fd,argv[2],strlen(argv[2])) == -1)
    {
        printf("Error writing to named pipe %s: %s\n", argv[1],
strerror(errno));
        return 1;
    }
}
close(fd);
return 0;
```

Output:

```
rakj13@RakeshJadhav:~/UnixLab$ gcc sp4.c
rakj13@RakeshJadhav:~/UnixLab$ ./a.out
USAGE ./a.out <file> [<arg>]
rakj13@RakeshJadhav:~/UnixLab$ ./a.out RESULT "Unix OS"
```