VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

UNIX SHELL AND PROGRAMMING

Submitted by

ANEESH PUNICHITHAYA(1BM20CS013)

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
October-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 “LAB COURSE **UNIX SHELL AND PROGRAMMING**” carried out by **ANEESH PUNICHITHAYA (1BM20CS013),** who is bonafide 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. The Lab report has been approved as it satisfies the academic requirements in respect of a **Unix Shell and Programming - (20CS5PCUSP)** work prescribed for the said degree.

Seema Patil
Associate Professor
Department of CSE
BMSCE, Bengaluru
`

Dr. Jyothi S Nayak
Professor and Head
Department of CSE
BMSCE, Bengaluru

Index

19	16/1/23	Write a C/C++ POSIX compliant program that prints the POSIX defined Configuration options supported on any given system using feature test macros.	23
20	16/1/23	Write a C/C++ program which demonstrates Interprocess Communication between a reader process and a writer process. Use mkfifo, open, read, write and close apis in your program.	25

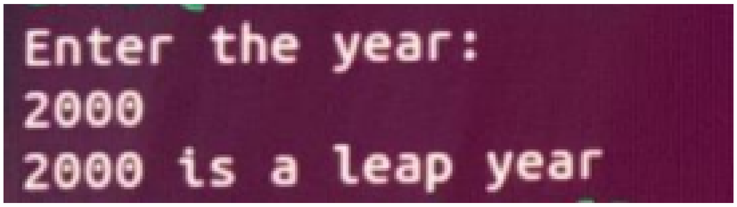
Experiment No 1

Shell script to find if the given year is leap or not

CODE:

```
#!/bin/bash
echo "Enter an 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:



```
Enter the year:
2000
2000 is a leap year
```

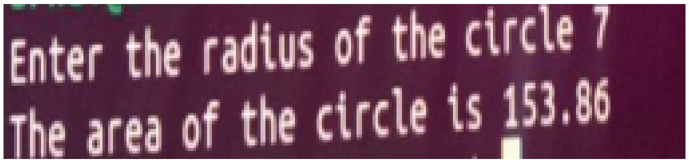
Experiment No 2

Shell script to find the area of a circle

CODE:

```
#!/bin/bash
echo "\nEnter the radius of a circle : "
read r
d=$(echo "scale=2;2 * $r"| bc) #Diameter
area=$(echo "scale=2; 22/7 * ($r * $r)" | bc)
circumference=$(echo "scale=2; 22/7 * $d"| bc)
echo "\nArea of circle is : $area"
echo "\nCircumference of circle is : $circumference \n"
```

OUTPUT:



```
Enter the radius of the circle 7
The area of the circle is 153.86
```

Experiment No 3

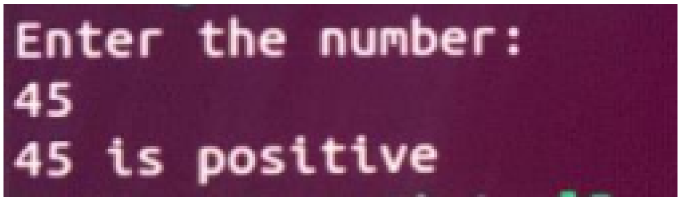
Shell script to check whether the number is zero/ positive/ negative

CODE:

```
#!/bin/bash
echo "Enter the number : "
read num
if [ $num -gt 0 ]
then
 echo "$num is positive"
elif [ $num -lt 0 ]
then
 echo "$num is negative"
else
 echo "$num is zero"
```

```
fi
```

OUTPUT:



```
Enter the number:
45
45 is positive
```

Experiment No 4

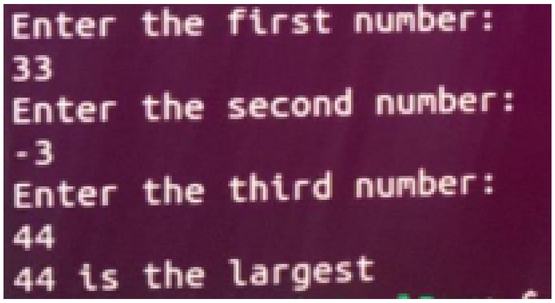
Shell script to find the biggest of three numbers

CODE:

```
#!/bin/bash
echo "Enter first number : "
read num1
echo "Enter second number : "
read num2
echo "Enter third number : "
read num3
if [ $num1 -gt $num2 ] && [ $num1 -gt $num3 ]
then
 echo "\n$num1 is the greatest"
elif [ $num2 -gt $num1 ] && [ $num2 -gt $num3 ]
then
```

```
 echo "\n$num2 is the greatest"
else
 echo "\n$num3 is the greatest"
fi
```

OUTPUT:



```
Enter the first number:
33
Enter the second number:
-3
Enter the third number:
44
44 is the largest
```

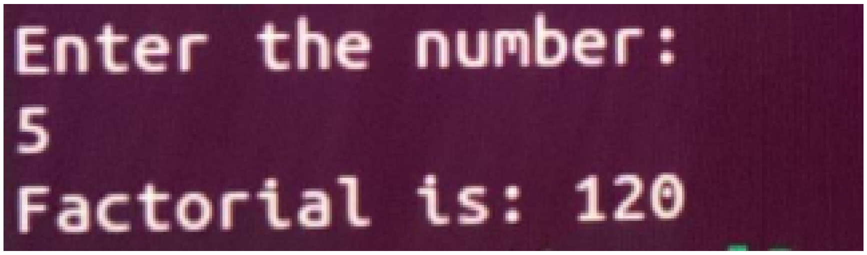
Experiment No 5

Shell script to find the factorial of a number

CODE:

```
#!/bin/bash
echo "ENTER THE NUMBER: "
read n
fact=1
while [ $n -gt 1 ]
do
 fact=$(( fact * n))
 n=$((n-1 ))
done
echo "FACTORIAL IS: "
echo $fact
```

OUTPUT:



```
Enter the number:
5
Factorial is: 120
```

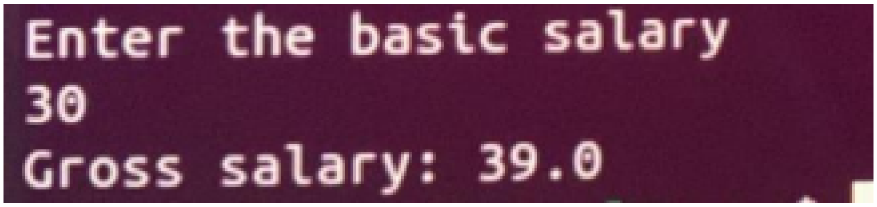
Experiment No 6

Shell script to compute the gross salary of an employee

CODE:

```
#!/bin/bash
echo "\nEnter name of Employee :"
read name
echo "\nEnter DA :"
read da
echo "\nEnter HRA:"
read hra
echo "\nEnter basic"
read basic
sal=$(( $da + $hra + $basic ))
echo "\nGross Salary of $name is $sal"
```

OUTPUT:



```
Enter the basic salary
30
Gross salary: 39.0
```

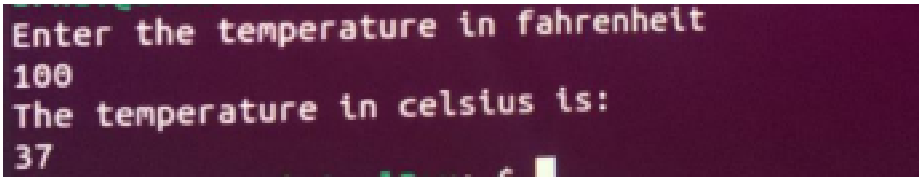
Experiment No 7

Shell script to convert the temperature Fahrenheit to Celsius

CODE:

```
#!/bin/bash
echo "Enter temperature in F : "
read f
c=$(echo "scale=2;(5/9)*($f-32)"|bc)
echo "$f °F = $c °C"
```

OUTPUT:



```
Enter the temperature in fahrenheit
100
The temperature in celsius is:
37
```

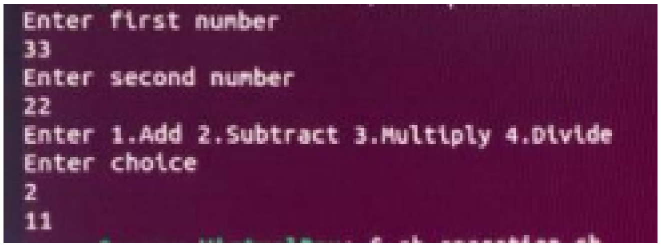
Experiment No 8

Shell script to perform arithmetic operations on given two numbers

CODE:

```
#!/bin/bash

echo "Enter 2 Numbers : "

read a

read b

echo "Enter Operation : \n"

echo "1) Addition"

echo "2) Subtraction"

echo "3) Multiplication"

echo "4) Division(Quotient)"
echo "5) Modulus(Remainder)\n"
read op
case $op in
 1)echo "scale=3; $a + $b" | bc -l ;;
 2)echo "scale=3; $a - $b" | bc -l ;;
 3)echo "scale=3; $a \* $b" | bc -l ;;
 4)echo "scale=3; $a / $b" | bc -l ;;
 5)echo "scale=3; $a % $b" | bc -l ;;
 *)echo "Choose a valid option"
esac
```

OUTPUT:



```
Enter first number
33
Enter second number
22
Enter 1.Add 2.Subtract 3.Multiply 4.Divide
Enter choice
2
11
```

Experiment No 9

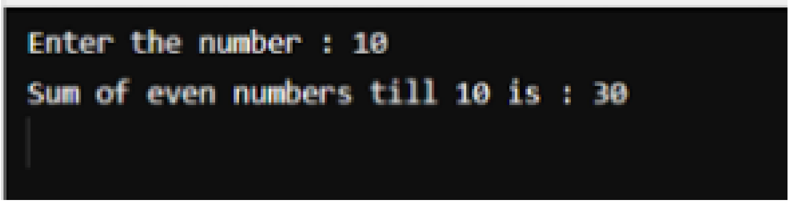
Shell script to find the sum of even numbers upto n

CODE:

```
#!/bin/bash
sum=0
read -p "Enter maximum limit of Even Numbers : " m
for ((i = 0; i < m; i++)); do
 if [[ $i%2 -eq 0 ]]; then
 sum=$(expr $sum + $i)
 fi
done
```

```
echo $sum
```

OUTPUT:



```
Enter the number : 10
Sum of even numbers till 10 is : 30
```

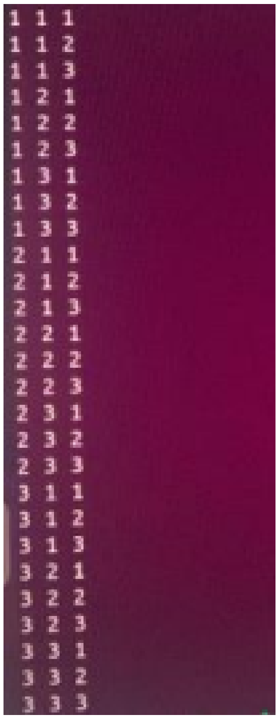
Experiment No 10

Shell script to print the combinations of numbers 123

CODE:

```
#!/bin/bash
echo "Combinations for 123 :"
for ((i = 1; i <= 3; i++)); do
 for ((j = 1; j <= 3; j++)); do
 for ((k = 1; k <= 3; k++)); do
 echo $i $j $k
 done
 done
done
```

OUTPUT:



```
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
```

Experiment No 11

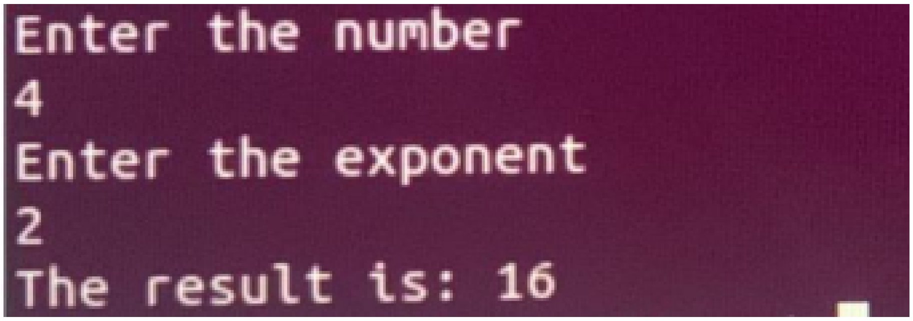
Shell script to find the power of a number

CODE:

```
#!/bin/bash
echo "Enter base"
read a
echo "Enter power"
read b
res=1
for ((i = 1; i <= b; i++)); do
 res=`expr $res \* $a`
```

```
done
echo $res
```

OUTPUT:



```
Enter the number
4
Enter the exponent
2
The result is: 16
```

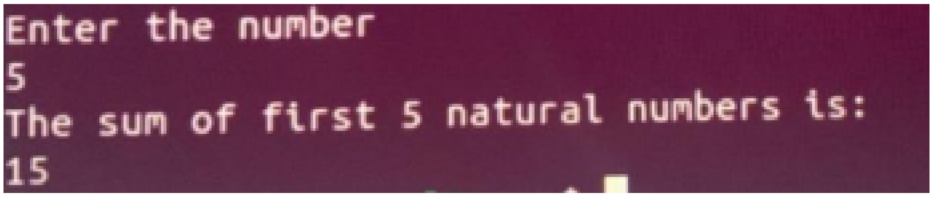
Experiment No 12

Shell script to find the sum of n natural numbers

CODE:

```
#!/bin/bash
echo "Enter a number"
read n
i=1
sum=0
while [ $i -le $n ]
 do
 echo "$i"
 sum=$(( $sum + $i ))
 i=$(( $i + 1 ))
 done
echo "Sum=$sum"
```

OUTPUT:



```
Enter the number
5
The sum of first 5 natural numbers is:
15
```

Experiment No 13

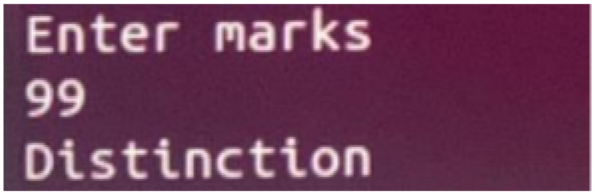
Shell script to display the pass class of a student

CODE:

```
#!/bin/bash
echo "Enter m1:\c and Enter m2:\c "
read m1
echo "Enter m3:\c"
read m3
echo "Enter m4:\c"
read m4
```

```
echo "Enter m5:\c"
read m5
tot=`expr $m1 + $m2 + $m3 + $m4 + $m5`;
avg=`expr $tot / 5`;
echo "total : $tot \n avg : $avg"
if [ $avg -gt 85 ];then
echo " Grade: Distinction "
elif [ $avg -gt 65 ];then
echo " Grade: First Class "
elif [ $avg -gt 50 ];then
echo " Grade: Second Class "
elif [ $avg -gt 35 ];then
echo " Grade: Pass "
else echo " Grade: Fail"
fi
```

OUTPUT:



```
Enter marks
99
Distinction
```

Experiment No 14

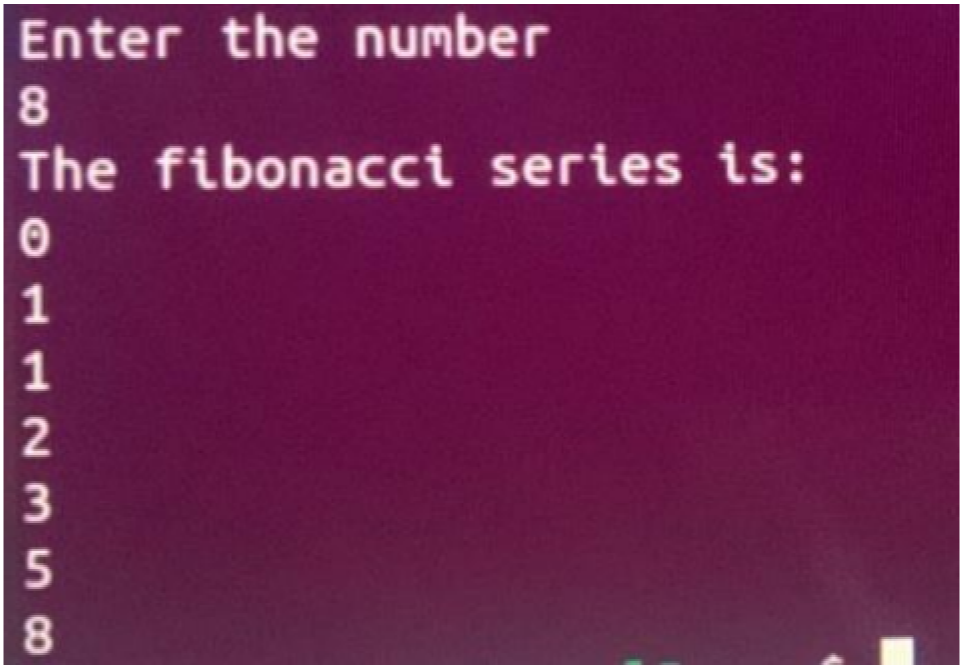
Shell script to find the Fibonacci series up to n

CODE:

```
#!/bin/bash
read N
a=0
b=1
echo "The Fibonacci series is : "
for (( i=0; i<N; i++ ))
```

```
do
 echo "$a"
 fib=$((a + b))
 a=$b
 b=$fib
done
```

OUTPUT:



```
Enter the number
8
The fibonacci series is:
0
1
1
2
3
5
8
```

Experiment No 15

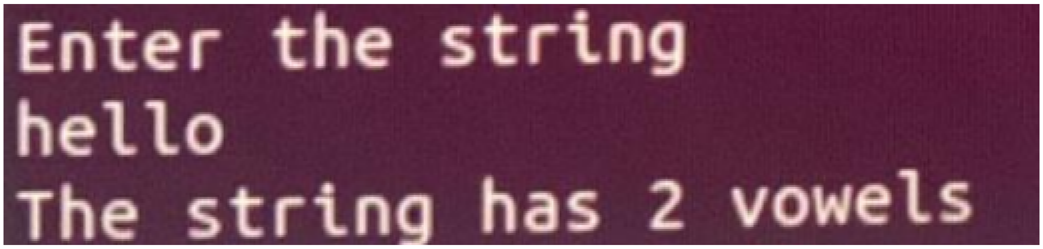
Shell script to count the number of vowels of a string

CODE:

```
#!/bin/bash
echo "enter filename"
read filename
vowels=`cat $filename | tr -cd 'aeiouAEIOU' | wc -c`
```

```
echo "Number of vowels in $filename: $vowels"
```

OUTPUT:



```
Enter the string
hello
The string has 2 vowels
```

Experiment No 16

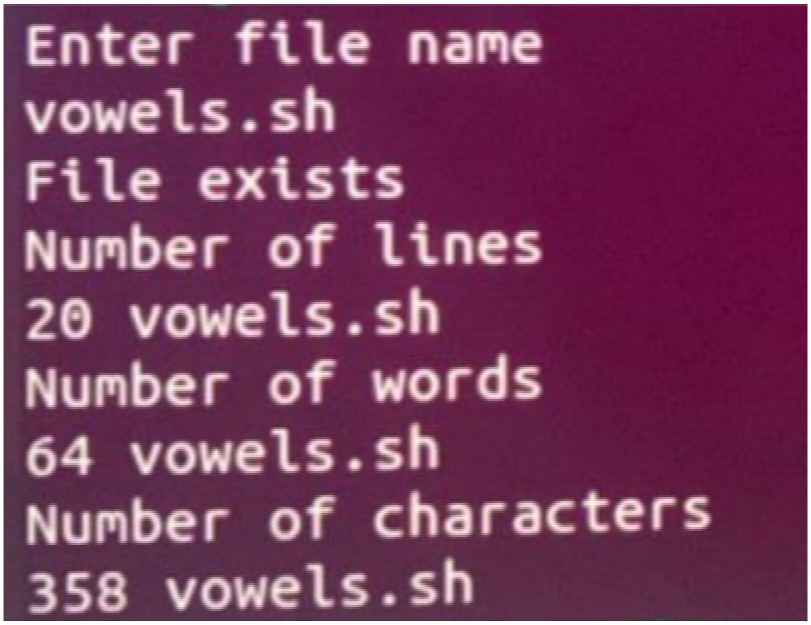
Shell script to check number of lines, words, characters in a file

CODE:

```
#!/bin/bash
echo "Enter the filename or path to proceed"
read filename
words=`wc -w $filename`
lines=`wc -l $filename`
```

```
chars=`wc -c $filename`
echo "Words is $words"
echo "Lines is $lines"
echo "Characters is $chars"
```

OUTPUT:



```
Enter file name
vowels.sh
File exists
Number of lines
20 vowels.sh
Number of words
64 vowels.sh
Number of characters
358 vowels.sh
```

Experiment No 17

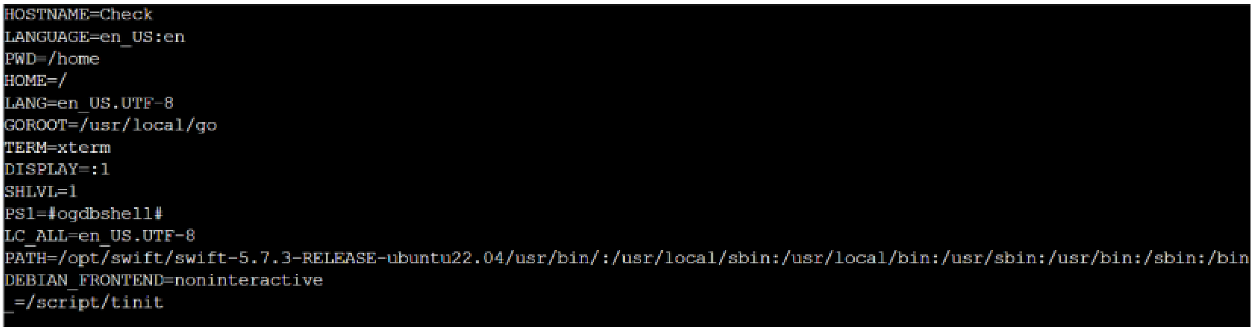
Write a C/C++ program to that outputs the contents of its environment list

CODE:

```
#include&lt;stdio.h&gt;

#include&lt;unistd.h&gt;
```

```
int main(int argc,char *argv[])

{

char **ptr;

extern char **environ;

for(ptr=environ; *ptr; ptr++)

printf(&quot;%s\n&quot;,*ptr);

return 0;

}
```

OUTPUT:



```
HOSTNAME=Check
LANGUAGE=en_US:en
PWD=/home
HOME=/
LANG=en_US.UTF-8
GOROOT=/usr/local/go
TERM=xterm
DISPLAY=:1
SHLVL=1
PS1=#ogdbshell#
LC_ALL=en_US.UTF-8
PATH=/opt/swift/swift-5.7.3-RELEASE-ubuntu22.04/usr/bin/:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
DEBIAN_FRONTEND=noninteractive
_=/script/tinit
```

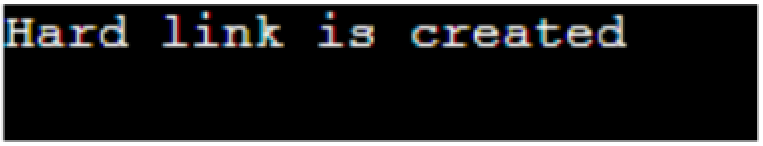
Experiment No 18

Write a C/C++ program to emulate the Unix ln command

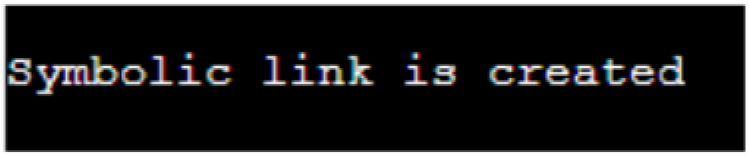
CODE:

```
#include<unistd.h>
#include<stdio.h>
#include<string.h>
int main(int argc , char * argv[])
{
        if(argc<3 || argc>4)
        {
              printf("Error in usage\n");
              return -1;
        }
        if(argc==4 && strcmp(argv[1],"-s")!=0)
        {
              printf("for symbolic link use -s option");
              return -1;
        }
        if(argc==4 && access(argv[2] , F_OK)==-1)
        {
                printf("Source file does not exist");
                return -1;
        }
        if(argc==3 && access(argv[1] , F_OK)==-1)
        {
                printf("Source file does not exist");
                return -1;
        }
        if(argc==4)
        {
                symlink(argv[2] , argv[3]);
                printf("Symbolic link is created");
                return 0;
        }
        if(argc==3)
        {
                link(argv[1] , argv[2]);
                printf("Hard link is created");
                return 0;
        }
}
```

OUTPUT:



```
Hard link is created
```



```
Symbolic link is created
```

Experiment No 19

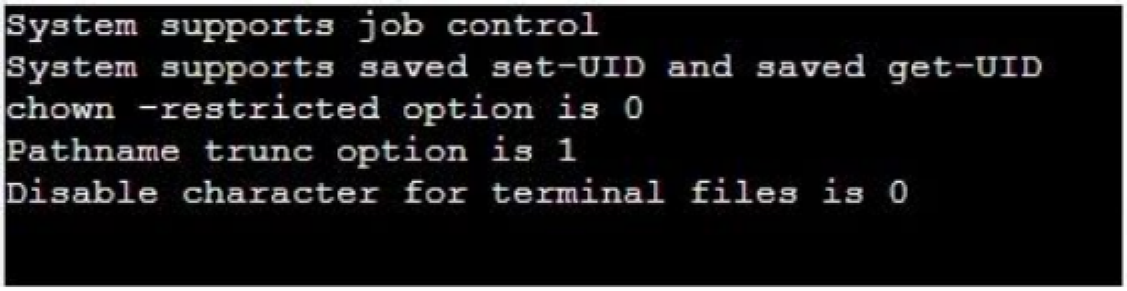
Write a C/C++ POSIX compliant program that prints the POSIX defined Configuration options supported on any given system using feature test macros.

CODE:

```
#define _POSIX_SOURCE

#define _POSIX_C_SOURCE 199309L

#include&lt;iostream&gt;

#include&lt;unistd.h&gt;

int main()

{

using namespace std;

#ifdef _POSIX_JOB_CONTROL

cout&lt;&lt;&quot;System Supports Job Control

feature&quot;&lt;&lt;endl; #else

cout&lt;&lt;&quot;System doesnot support job control\n&quot;;

#endif

#ifdef _POSIX_SAVED_IDS

cout&lt;&lt;&quot;System Supports saved set-UID and saved set-GID&quot;&lt;&lt;endl;

#else

cout&lt;&lt;&quot;System doesnot support saved set-UID\n&quot;;

#endif

#ifdef _POSIX_CHOWN_RESTRICTED

cout&lt;&lt;&quot;System Supports Change Ownership

feature:&quot;&lt;&lt;endl; #else

cout&lt;&lt;&quot;System doesnot support change Ownership feature\n&quot;;

#endif

#ifdef _POSIX_NO_TRUNC

cout&lt;&lt;&quot;System Supports Path truncation option:&quot;&lt;&lt;endl;
```

```
#else
cout&lt;&lt;&quot;System doesnot support Path truncation \n&quot;;
#endif
#ifdef _POSIX_VDISABLE
cout&lt;&lt;&quot;System Supports Disable Character for
files:&quot;&lt;&lt;endl; #else
cout&lt;&lt;&quot;System doesnot support Disable Characters \n&quot;;
#endif
return 0;
```

OUTPUT:



```
System supports job control
System supports saved set-UID and saved get-UID
chown -restricted option is 0
Pathname trunc option is 1
Disable character for terminal files is 0
```

Experiment No 20

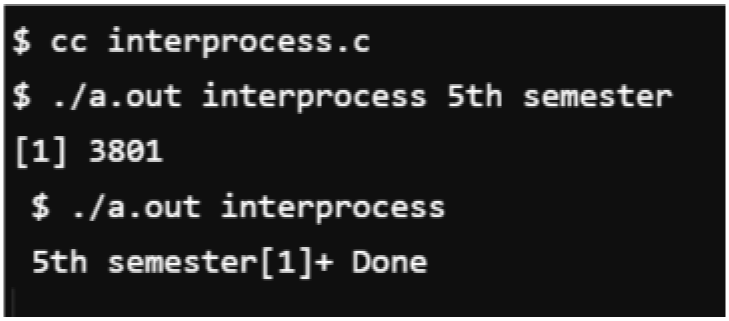
Write a C/C++ program which demonstrates Interprocess Communication between a reader process and a writer process. Use mkfifo, open, read, write and close apis in your program.

CODE:

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

        #include <string.h>

        #include <fcntl.h>

        #include <stdio.h>

        #include <unistd.h>

        int main(int argc, char *argv[])

         {

        char buf[100];

        int fd,n;

        mkfifo (argv[1], S_IFIFO |0777);

        if (argc == 3){
```

```
fd = open (argv[1], O_WRONLY);
write (fd, argv[2], strlen(argv[2]));
close(fd);}
if (argc ==2){
fd = open (argv[1], O_RDONLY);
n= read (fd, buf, sizeof(buf));
buf[n]='\0';
printf ("%s", buf);
close(fd);
 }
```

OUTPUT:



```
$ cc interprocess.c
$ ./a.out interprocess 5th semester
[1] 3801
 $ ./a.out interprocess
 5th semester[1]+ Done
```