VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“JnanaSangama”, Belgaum -590014, Karnataka.

LAB REPORT
on

UNIX Shell and Programming

Submitted by

Akansha Mehrotra (1BM20CS005)

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 **Akansha Mehrotra (1BM20CS005),** 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.

Name of the Lab-Incharge	**Dr. Jyothi S Nayak**
Designation	Professor and Head
Department of CSE	Department of CSE
BMSCE, Bengaluru	BMSCE, Bengaluru

`

Index

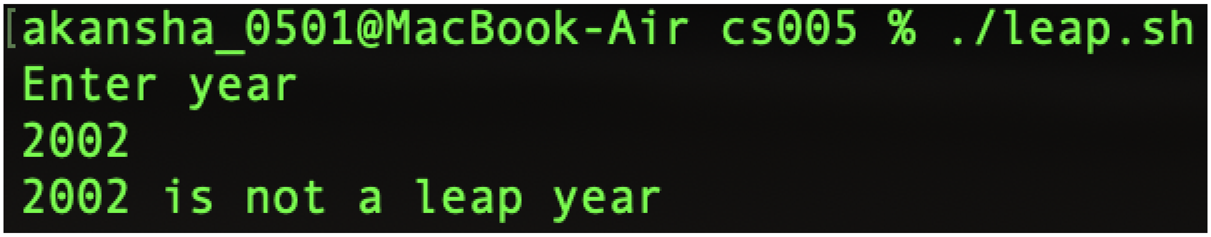
Experiment 1

Aim of the program - Shell script to find if the given year is leap or not

Program -

```
#!/bin/bash
echo "Enter year"
read y
c=`expr $y % 41`
d=`expr $y % 100`
e=`expr $y % 400`
if [ $c -eq 0 ] && [ $d -ne 0 ] || [ $e -eq 0 ]
then
echo "$y is a leap year"
else
echo "$y is not a leap year"
fi
```

Output -



```
[akansha_0501@MacBook-Air cs005 % ./leap.sh
Enter year
2002
2002 is not a leap year
```

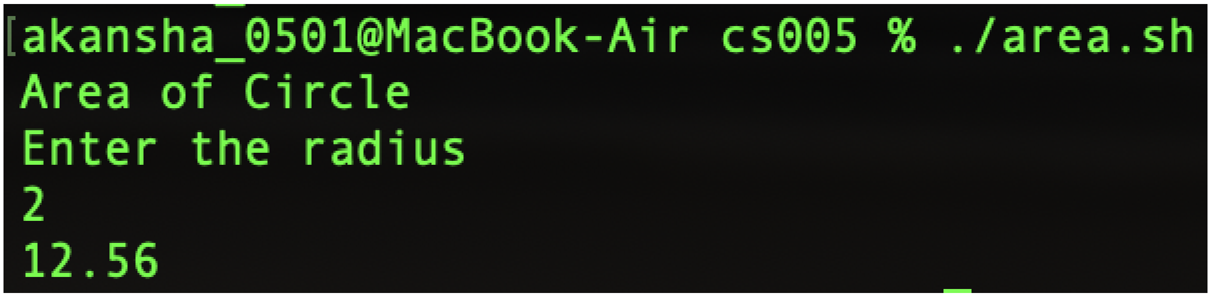
Experiment 2

Aim of the program - Shell script to find the area of a circle

Program -

```
#!/bin/bash

echo "Area of Circle"

echo "Enter the radius"

read r

echo "3.14 * $r * $r"|bc
```

Output -



```
[akansha_0501@MacBook-Air cs005 % ./area.sh
Area of Circle
Enter the radius
2
12.56
```

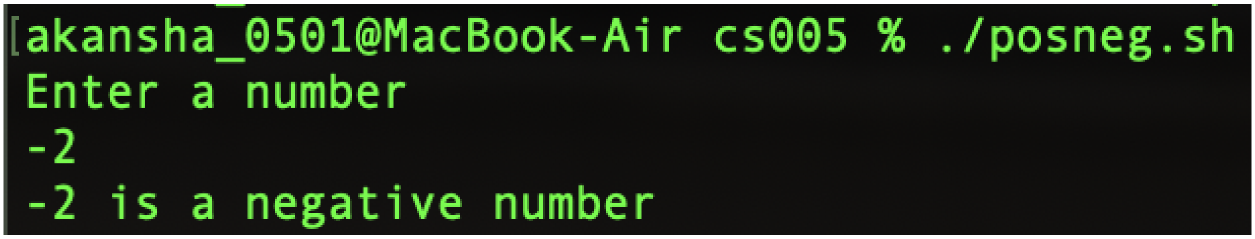
Experiment 3

Aim of the program - Shell script to check whether the number is zero/ positive/ negative

Program -

```
#!/bin/bash

echo "Enter a number"

read n

if [ $n -gt 0 ]

then

echo "$n is a positive number"

elif [ $n -lt 0 ]

then

echo "$n is a negative number"

else

echo "$n is zero"

fi
```

Output -



```
[akansha_0501@MacBook-Air cs005 % ./posneg.sh
Enter a number
-2
-2 is a negative number
```

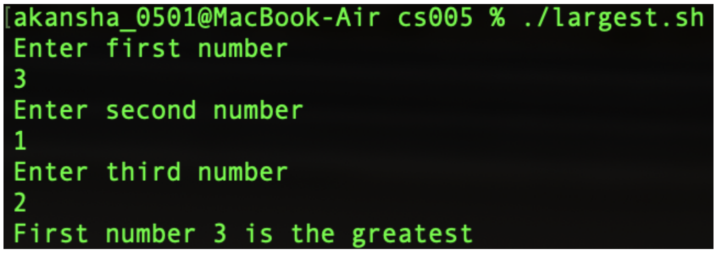
Experiment 4

Aim of the program - Shell script to find the biggest of three numbers

Program -

```
#!/bin/bash
echo "Enter first number"
read n1
echo "Enter second number"
read n2
echo "Enter third number"
read n3
if [ $n1 -ge $n2 ] && [ $n1 -ge $n3 ]
then
echo "First number $n1 is the greatest"
elif [ $n2 -ge $n1 ] && [ $n2 -ge $n3 ]
then
echo "Second number $n2 is the greatest"
else
echo "Third number $n3 is the greatest"
fi
```

Output -



```
[akansha_0501@MacBook-Air cs005 % ./largest.sh
Enter first number
3
Enter second number
1
Enter third number
2
First number 3 is the greatest
```

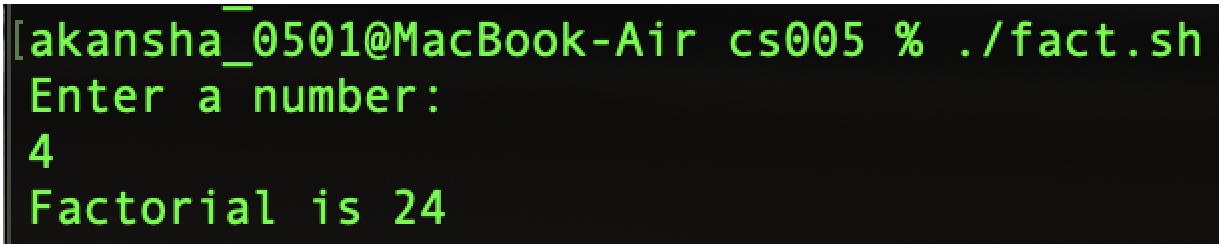
Experiment 5

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

Program -

```
#!/bin/bash
echo "Enter a number:"
read n
fact=1
while [ $n -gt 1 ]
do
fact=`expr $fact \* $n`
n=`expr $n - 1`
done
echo "Factorial is $fact"
```

Output -



```
[akansha_0501@MacBook-Air cs005 % ./fact.sh
Enter a number:
4
Factorial is 24
```

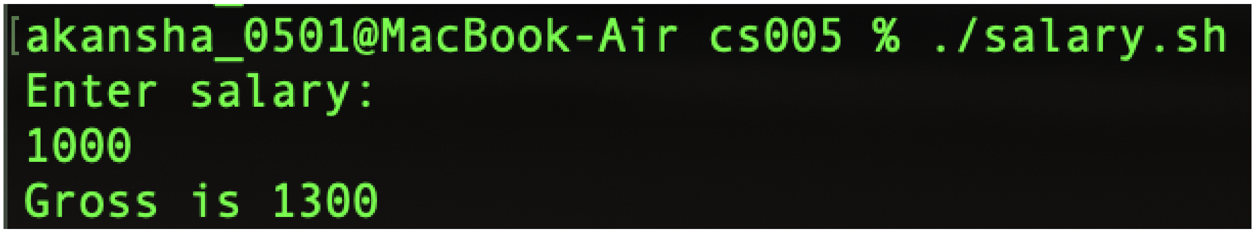
Experiment 6

Aim of the program - Shell script to compute the gross salary of an employee

Program -

```
#!/bin/bash
echo "Enter salary:"
read sal
total=$((($sal+(($sal/100)*20)+(($sal/100)*10))|bc))
echo "Gross is $total"
```

Output -



```
[akansha_0501@MacBook-Air cs005 % ./salary.sh
Enter salary:
1000
Gross is 1300
```

Experiment 7

Aim of the program - Shell script to convert the temperature Fahrenheit to Celsius

Program -

#!/bin/bash

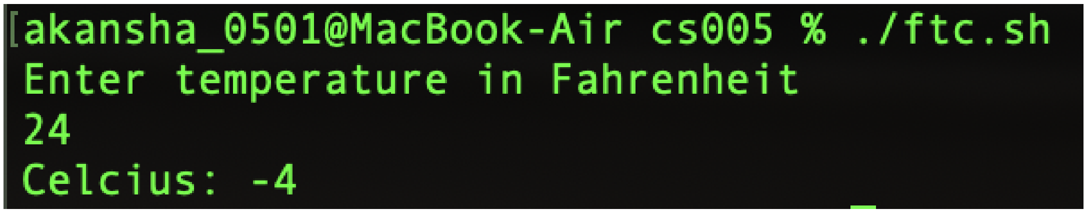
echo "Enter temperature in Fahrenheit"

read f

c=$(((($f-32)*5)/9))

echo "Celcius: $c"

Output -



```
[akansha_0501@MacBook-Air cs005 % ./ftc.sh
Enter temperature in Fahrenheit
24
Celcius: -4
```

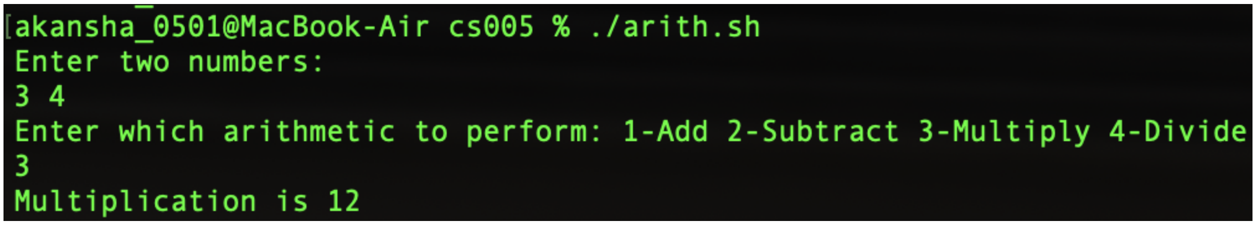
Experiment 8

Aim of the program - Shell script to perform arithmetic operations on given two numbers

Program -

```
#!/bin/bash

echo "Enter two numbers: "

read n1 n2

echo "Enter which arithmetic to perform: 1-Add 2-Subtract 3-Multiply 4-Divide"

read a

case $a in

1) echo "Addition is `expr $n1 + $n2`";;

2) echo "Difference is `expr $n1 - $n2`";;

3) echo "Multiplication is `expr $n1 \* $n2`";;

4) echo "Division is `expr $n1 / $n2`";;

*) echo "Invalid"

esac
```

Output -



```
[akansha_0501@MacBook-Air cs005 % ./arith.sh
 Enter two numbers:
 3 4
 Enter which arithmetic to perform: 1-Add 2-Subtract 3-Multiply 4-Divide
 3
 Multiplication is 12
```

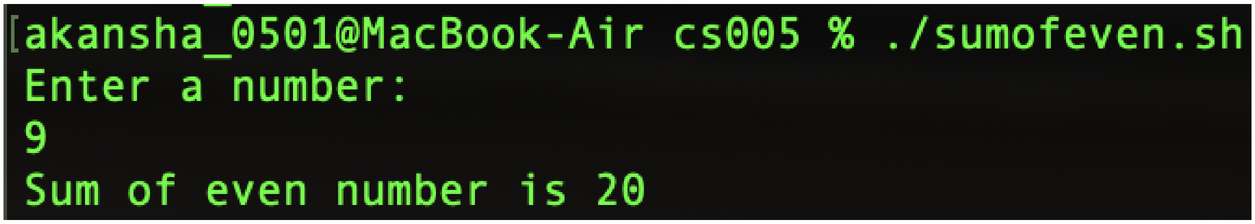
Experiment 9

Aim of the program - Shell script to find the sum of even numbers upto n

Program -

```
#!/bin/bash
echo "Enter a number:"
read n
sum=0
if [ `expr $n % 2` -ne 0 ]
then
n=`expr $n - 1`
fi
while [ $n -ge 0 ]
do
sum=`expr $sum + $n`
n=`expr $n - 2`
done
echo "Sum of even number is $sum"
```

Output -



```
[akansha_0501@MacBook-Air cs005 % ./sumofeven.sh
Enter a number:
9
Sum of even number is 20
```

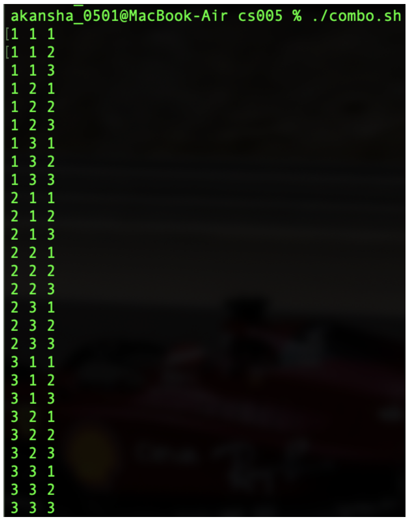
Experiment 10

Aim of the program - Shell script to print the combinations of numbers 123

Program -

```
#!/bin/bash
x=1 | y=2 | z=3
for i in $x $y $z
do
for j in $x $y $z
do
for k in $x $y $z
do
echo "$i $j $k"
done
done
done
```

Output -



```
 akansha_0501@MacBook-Air cs005 % ./combo.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
```

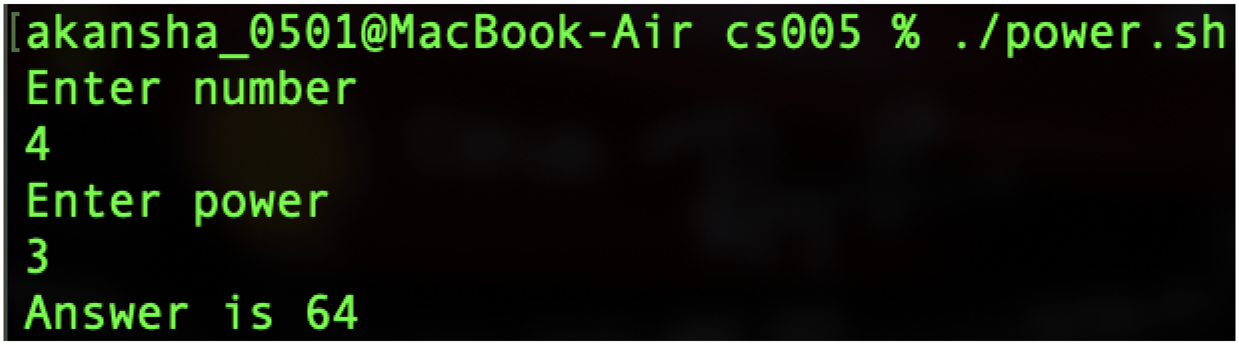
Experiment 11

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

Program -

```
#!/bin/bash
echo "Enter number"
read x
echo "Enter power"
read n
ans=1
while [ $n -gt 0 ]
do
ans=`expr $ans \* $x`
n=`expr $n - 1`
done
echo "Answer is $ans"
```

Output -



```
[akansha_0501@MacBook-Air cs005 % ./power.sh
Enter number
4
Enter power
3
Answer is 64
```

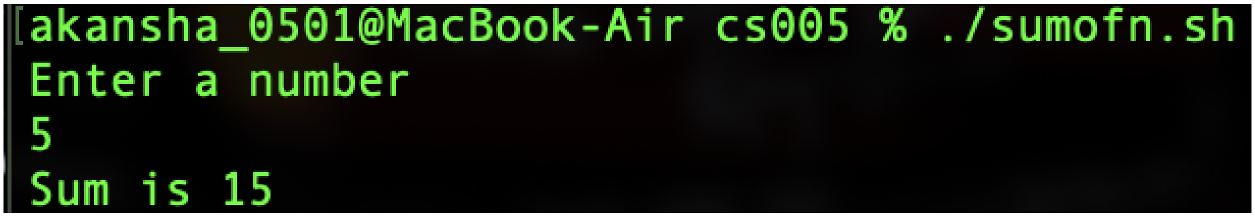
Experiment 12

Aim of the program - Shell script to find the sum of n natural numbers

Program -

```
#!/bin/bash
echo "Enter a number"
read n
sum=0
while [ $n -ge 1 ]
do
sum=`expr $sum + $n`
n=`expr $n - 1`
done
echo "Sum is $sum"
```

Output -



```
[akansha_0501@MacBook-Air cs005 % ./sumofn.sh
Enter a number
5
Sum is 15
```

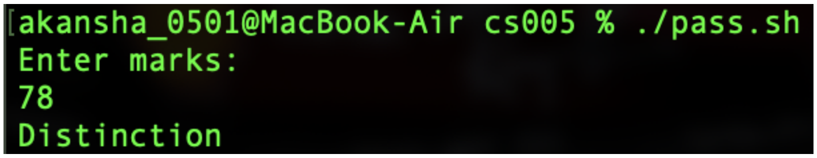
Experiment 13

Aim of the program - Shell script to display the pass class of a student

Program -

```
#!/bin/bash
echo "Enter marks: "
read n
if [ $n -gt 70 ]
then
echo "Distinction"
elif [ $n -gt 60 ] && [ $n -le 70 ]
then
echo "First class"
elif [ $n -gt 50 ] && [ $n -le 60 ]
then
echo "Second Class"
elif [ $n -gt 40 ] && [ $n -le 50 ]
then
echo "Pass"
else
echo "Fail"
fi
```

Output -



```
[akansha_0501@MacBook-Air cs005 % ./pass.sh
 Enter marks:
 78
 Distinction
```

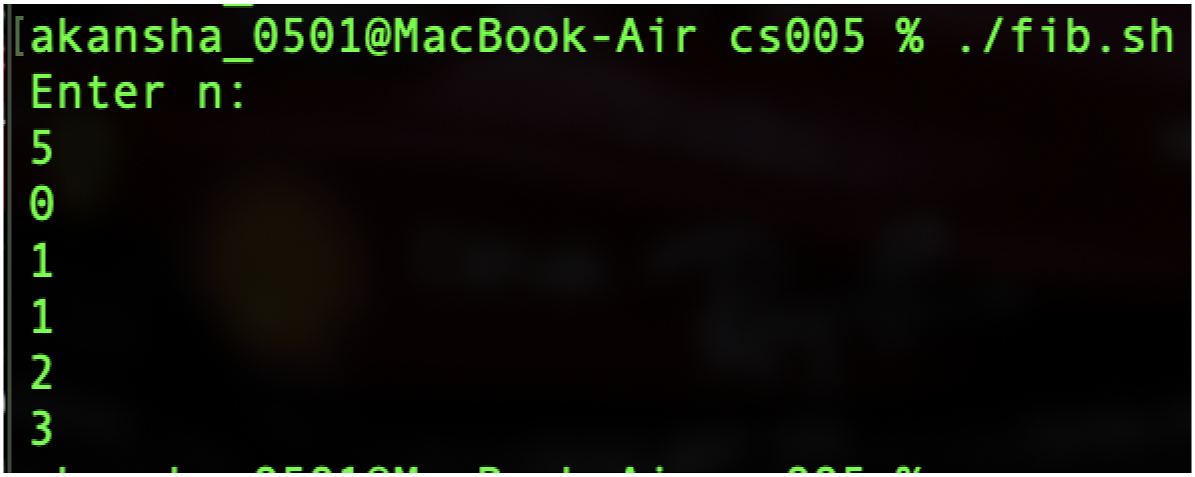
Experiment 14

Aim of the program - Shell script to find the Fibonacci series up to n

Program -

```
#!/bin/bash
echo "Enter n: "
read n
function fib
{
x=0
y=1
i=2
echo "$x"
echo "$y"
while [ $i -lt $n ]
do
i=`expr $i + 1`
z=`expr $x + $y`
echo "$z"
x=$y
y=$z
done
}
r=`fib $n`
echo "$r"
```

Output -



```
[akansha_0501@MacBook-Air cs005 % ./fib.sh
Enter n:
5
0
1
1
2
3
akansha_0501@MacBook-Air cs005 %
```

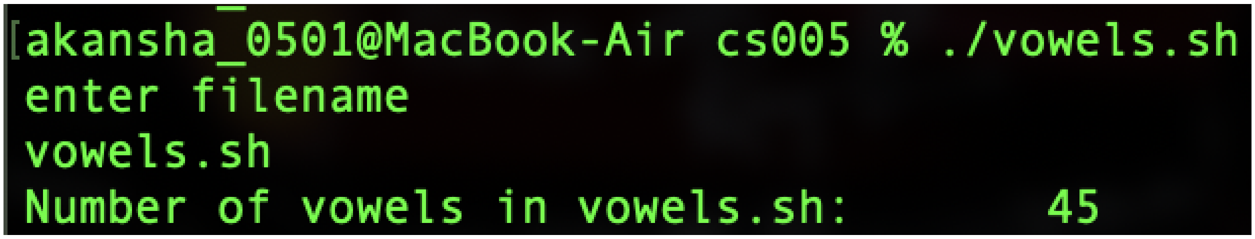
Experiment 15

Aim of the program - Shell script to count the number of vowels of a string

Program -

```
echo "enter filename"
read filename
vowels=`cat $filename | tr -cd 'aeiouAEIOU' | wc -c`
echo "Number of vowels in $filename: $vowels"
```

Output -



```
[akansha_0501@MacBook-Air cs005 % ./vowels.sh
enter filename
vowels.sh
Number of vowels in vowels.sh:       45
```

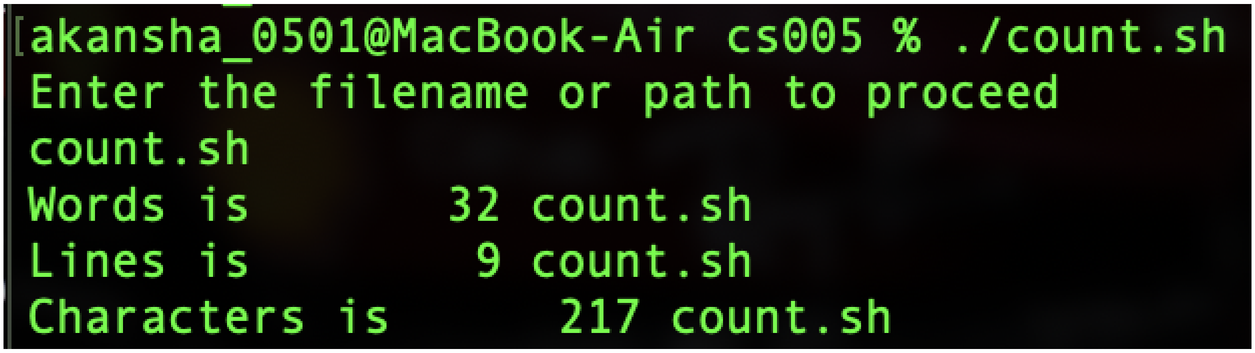
Experiment 16

Aim of the program - Shell script to check number of lines, words, characters in a file

Program -

```
#!/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 -



```
[akansha_0501@MacBook-Air cs005 % ./count.sh
Enter the filename or path to proceed
count.sh
Words is       32 count.sh
Lines is        9 count.sh
Characters is      217 count.sh
```

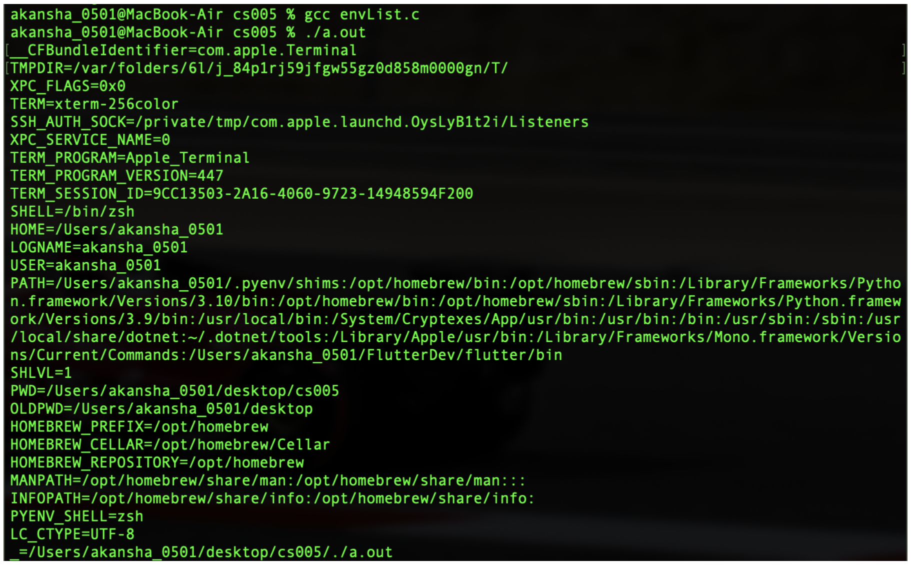
Experiment 17

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

Program -

```
#include<stdio.h>
int main(int argc, char* argv[ ])
{
int i;
char **ptr;
extern char **environ;
for( ptr = environ; *ptr != 0; ptr++ ) /*echo all env strings*/
printf("%s\n", *ptr);
return 0;
}
```

Output -



```
akansha_0501@MacBook-Air cs005 % gcc envList.c
akansha_0501@MacBook-Air cs005 % ./a.out
[__CFBundleIdentifier=com.apple.Terminal
[TMPDIR=/var/folders/6l/j_84p1rj59jfgw55gz0d858m0000gn/T/
XPC_FLAGS=0x0
TERM=xterm-256color
SSH_AUTH_SOCK=/private/tmp/com.apple.launchd.OysLyB1t2i/Listeners
XPC_SERVICE_NAME=0
TERM_PROGRAM=Apple_Terminal
TERM_PROGRAM_VERSION=447
TERM_SESSION_ID=9CC13503-2A16-4060-9723-14948594F200
SHELL=/bin/zsh
HOME=/Users/akansha_0501
LOGNAME=akansha_0501
USER=akansha_0501
PATH=/Users/akansha_0501/.pyenv/shims:/opt/homebrew/bin:/opt/homebrew/sbin:/Library/Frameworks/Pytho
n.framework/Versions/3.10/bin:/opt/homebrew/bin:/opt/homebrew/sbin:/Library/Frameworks/Python.framew
ork/Versions/3.9/bin:/usr/local/bin:/System/Cryptexes/App/usr/bin:/usr/bin:/bin:/usr/sbin:/sbin:/usr
/local/share/dotnet:~/.dotnet/tools:/Library/Apple/usr/bin:/Library/Frameworks/Mono.framework/Versio
ns/Current/Commands:/Users/akansha_0501/FlutterDev/flutter/bin
SHLVL=1
PWD=/Users/akansha_0501/desktop/cs005
OLDPWD=/Users/akansha_0501/desktop
HOMEBREW_PREFIX=/opt/homebrew
HOMEBREW_CELLAR=/opt/homebrew/Cellar
HOMEBREW_REPOSITORY=/opt/homebrew
MANPATH=/opt/homebrew/share/man:/opt/homebrew/share/man:::
INFOPATH=/opt/homebrew/share/info:/opt/homebrew/share/info:
PYENV_SHELL=zsh
LC_CTYPE=UTF-8
_=/Users/akansha_0501/desktop/cs005/./a.out
```

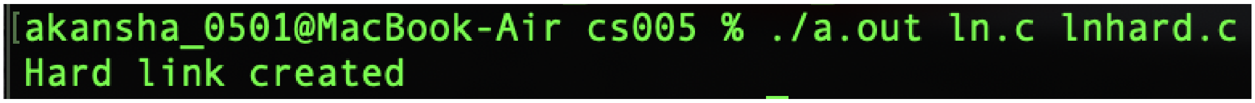
Experiment 18

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

Program -

```
#include<stdio.h>
#include<sys/types.h>
#include<unistd.h>
#include<string.h>
int main(int argc, char * argv[])
{
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 -



```
[akansha_0501@MacBook-Air cs005 % ./a.out ln.c lnhard.c
Hard link created
```

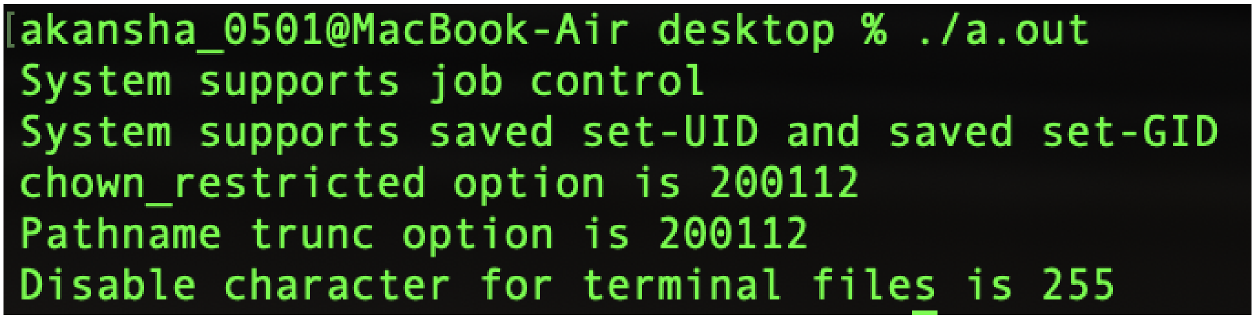
Experiment 19

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

Program -

```
#define _POSIX_SOURCE
#define _POSIX_C_SOURCE 199309L
#include<stdio.h>
#include<unistd.h>
int main()
{
if(_POSIX_JOB_CONTROL)
printf("System supports job control\n");
else
printf("System does not support job control \n");
if(_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");
if(_POSIX_CHOWN_RESTRICTED)
printf("chown_restricted option is %d\n",_POSIX_CHOWN_RESTRICTED);
else
printf("System does not support chown_restricted option \n");
if(_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”);
if(_POSIX_VDISABLE)
printf(“Disable character for terminal files is %d\n”,_POSIX_VDISABLE);
else
printf(“ System does not support _POSIX_VDISABLE \n”);
return 0;
}
```

Output -



```
[akansha_0501@MacBook-Air desktop % ./a.out
 System supports job control
 System supports saved set-UID and saved set-GID
 chown_restricted option is 200112
 Pathname trunc option is 200112
 Disable character for terminal files is 255
```

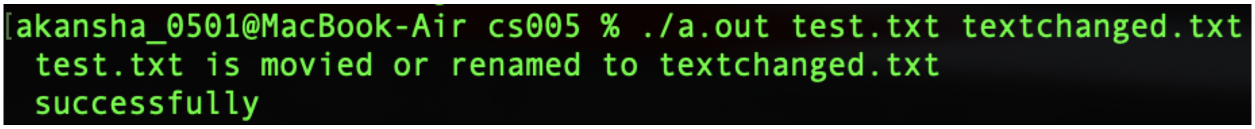
Experiment 20

Aim of the program - 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.

Program -

```
#include<fcntl.h>
#include<stdio.h>
#include<unistd.h>
#include<sys/stat.h>
int main(int argc, char **argv) {
if(argc>3 || argc<3)
{
printf("Please Provide two arugments \n");
} else{
int fd1,fd2;
int n,count=0; if(access(argv[1],F_OK)<0)
{
printf("%s not found \n ",argv[1]);
}
if(rename(argv[1],argv[2])==0)
printf(" %s is movied or renamed to %s \n successfully \n",argv[1],argv[2]); return (0);
}
}
```

Output -



```
[akansha_0501@MacBook-Air cs005 % ./a.out test.txt textchanged.txt
 test.txt is movied or renamed to textchanged.txt
 successfully
```