**LAB PROGRAMS USP**

**Lab Program 1:**

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

**PROGRAM:**

echo "Check whether the year is a leap year or not" echo "Enter year: "

read year

if [ `expr $year % 4` -eq 0 ] then

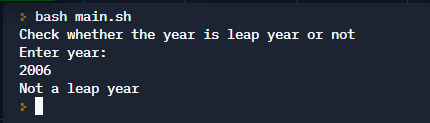
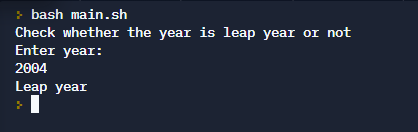
else

fi

echo "Leap year"

echo "Not a leap year"

# OUTPUT:



**LAB PROGRAM 2**

**Q.** Shell script to find the area of a circle.

**PROGRAM:**

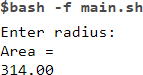
echo "Enter radius: " read r

echo "Area = "

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

# OUTPUT:

(STDIN = 10)



**LAB PROGRAM 3**

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

**PROGRAM:**

echo "Check whether the no is given is positive or negative or zero" echo "Read n1"

read n1

if [ $n1 -gt 0 ] then

echo "Positve" elif [ $n1 -lt 0 ]

then

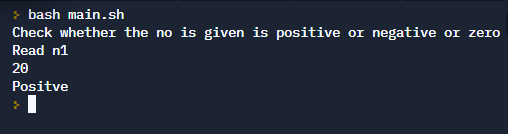
else

fi

echo "Negative"

echo "Zero"

# OUTPUT:



**LAB PROGRAM 4**

**Q.** Shell script to find the biggest of three numbers.

**PROGRAM:**

echo "Enter n1" read n1

echo "Enter n2" read n2

echo "Enter n3" read n3

if [ $n1 -gt $n2 -a $n1 -gt $n3 ] then

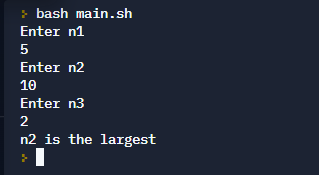
echo "n1 is the largest"

elif [ $n2 -gt $n1 -a $n2 -gt $n3 ] then

echo "n2 is the largest" else

echo "n3 is largest" fi

# OUTPUT:



**LAB PROGRAM 5**

**Q.** Shell script to find the factorial of a number.

**PROGRAM:**

echo "Enter a number: " read n

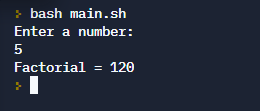
f=1

while [ $n -gt 1 ] do

f=`expr $f \\* $n` n=`expr $n - 1` done

echo "Factorial = $f"

# OUTPUT:



**LAB PROGRAM 6**

**Q.** Shell script to compute the gross salary of an employee.

**PROGRAM:**

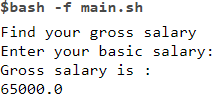
echo "Find your gross salary" echo "Enter your basic salary: " read s

echo "Gross salary is :" da=`echo "0.1 \* $s" | bc` hra=`echo "0.2 \* $s" | bc`

gross=`echo "$da + $hra + $s" | bc` echo $gross

# OUTPUT:

(STDIN = 50000)



**LAB PROGRAM 7**

**Q.** Shell script to convert the temperature Fahrenheit to Celsius.

**PROGRAM:**

echo "Enter the temperatire in fahrenheit" read f

x=`expr $f - 32`

c=`echo "scale=2; $x \* 5/9" | bc`

echo "$f in Fahrenheit = $c in Celcius"

# OUTPUT:

(STDIN = 68)



**LAB PROGRAM 8**

**Q.** Shell script to perform arithmetic operations on given two numbers.

**PROGRAM:**

echo "Enter 2 numbers: " read a b

echo "1. Add

1. Subtract
2. Multiply
3. Divide
4. Remainder"

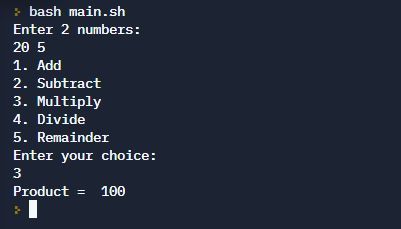
echo "Enter your choice: " read ch

case $ch in

1. echo "Sum = " `expr $a + $b`;;
2. echo "Difference = " `expr $a - $b`;;
3. echo "Product = " `expr $a \\* $b`;;
4. echo "Quotient = " `expr $a / $b`;;
5. echo "Remainder = " `expr $a % $b`;;

\*) echo "Invalid option" esac

# OUTPUT:



**LAB PROGRAM 9**

**Q.** Shell script to find the sum of even numbers up to n.

**PROGRAM:**

echo "Enter the value of n: " read n

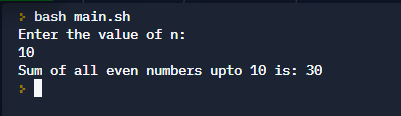
sum=0

for (( i=0; i<=n; i=i+2 )) do

sum=`expr $sum + $i` done

echo "Sum of all even numbers upto $n is: $sum"

# OUTPUT:



**LAB PROGRAM 10**

**Q.** Shell script to print the combinations of numbers 123.

**PROGRAM:**

for i in 1 2 3 do

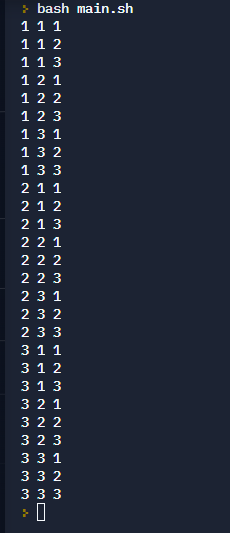
for j in 1 2 3 do

for k in 1 2 3 do

echo $i $j $k done

done done

# OUTPUT:



**LAB PROGRAM 11**

**Q.** Shell script to find the power of a number.

**PROGRAM:**

echo "Enter the value of the number: " read n

echo "Enter the value of the power: " read p

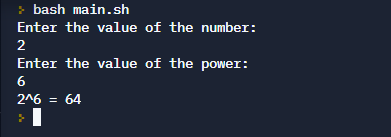
total=1

for (( i=1; i<=p; i++ )) do

total=`expr $total \\* $n` done

echo "$n^$p = $total"

# OUTPUT:



**LAB PROGRAM 12**

**Q.** Shell script to find the sum of n natural numbers.

**PROGRAM:**

echo "Enter the value of n:" read n

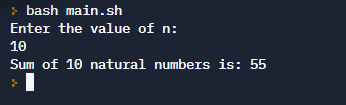
sum=0

for (( i=1; i<=n; i++ )) do

sum=`expr $sum + $i` done

echo "Sum of $n natural numbers is: $sum"

# OUTPUT:



**LAB PROGRAM 13**

**. Q** Shell script to display the pass class of a student.

**PROGRAM:**

#!/bin/sh pass=0 fail=0

i=1

while [ $i -le 6 ] do

echo "Enter the cie and see marks(out of 50 for see) of the sub$i " read cie see

total=`expr $cie+$see|bc` case $total in

100) echo "S grade "

pass=$((pass+1)) ;;

9[0-9]) echo "S grade "

pass=$((pass+1)) ;;

8[0-9]) echo "A grade "

pass=$((pass+1)) ;;

7[0-9]) echo "B grade "

pass=$((pass+1)) ;;

6[0-9]) echo "C grade "

pass=$((pass+1)) ;;

5[0-9]) echo "D grade "

pass=$((pass+1)) ;;

4[0-9]) echo "E grade "

pass=$((pass+1)) ;;

[0123][0-9]) echo "F grade "

fail=$((fail+1)) ;;

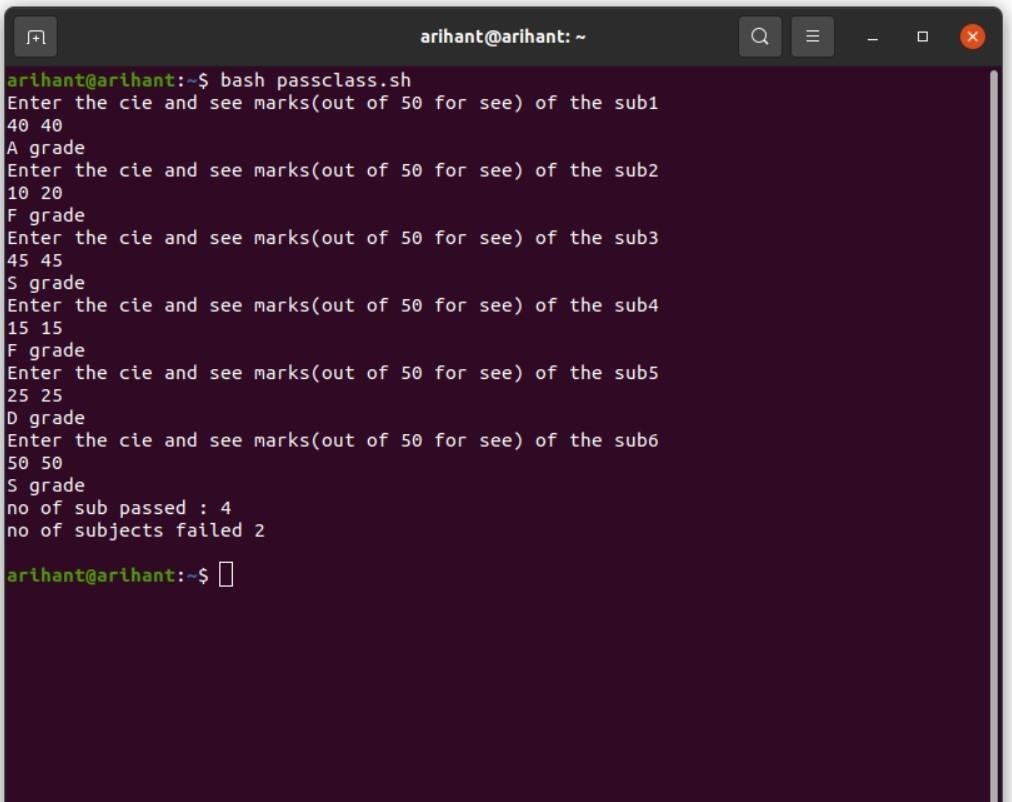
\*)echo "error in input"

esac i=$((i+1))

done

echo -e "no of sub passed : $pass\nno of subjects failed $fail\n"

**OUTPUT:**



**LAB PROGRAM 14**

**Q.** Shell script to find the Fibonacci series up to n

**PROGRAM:**

#!/bin/sh

echo "Enter the no" read no

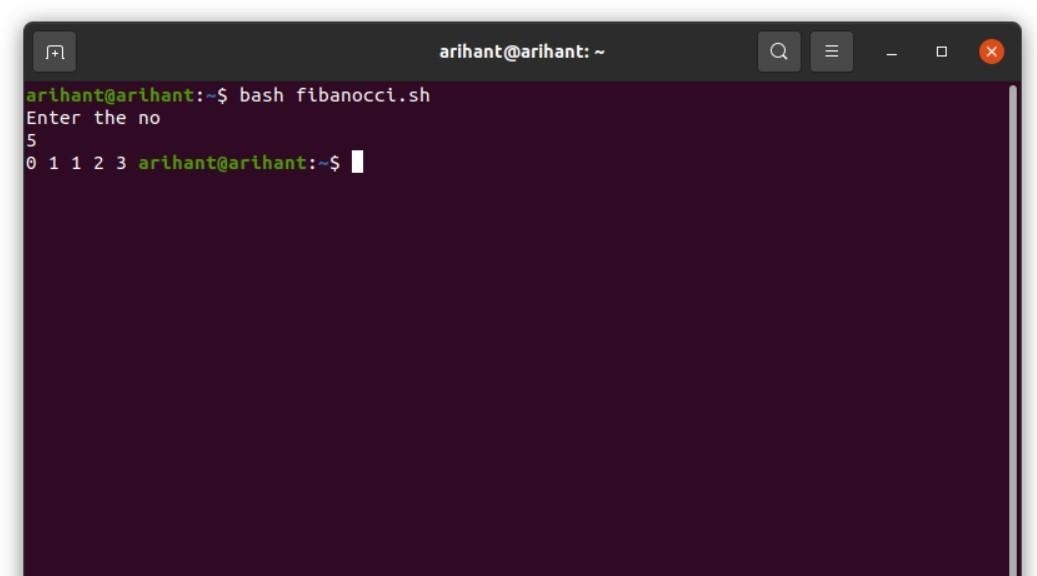
m=0 n=1

while [ $no -gt 0 ] do

echo -e "$m \c" temp=$m m=$((m+$n)) n=$temp no=$((no-1))

Done

**OUTPUT:**



**LAB PROGRAM 15**

**Q. Shell script to count the number of vowels of a string**

**PROGRAM:**

CODE:

#!/bin/sh

echo "Enter the string " read str

count=0

len=`expr length $str` while [ $len -gt 0 ]

do

ch=`expr $str | cut -c $len` case $ch in

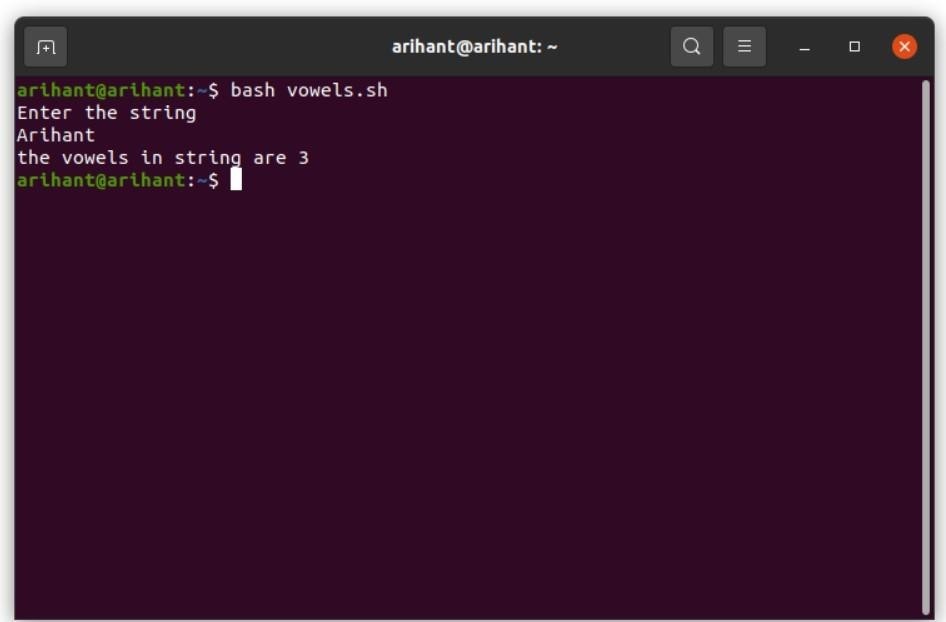
[aeiouAEIOU]) count=$((count+1)) ;; esac

len=$((len-1))

done

echo "the vowels in string are $count "

**OUTPUT:**



**LAB PROGRAM 16**

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

**PROGRAM**

CODE:

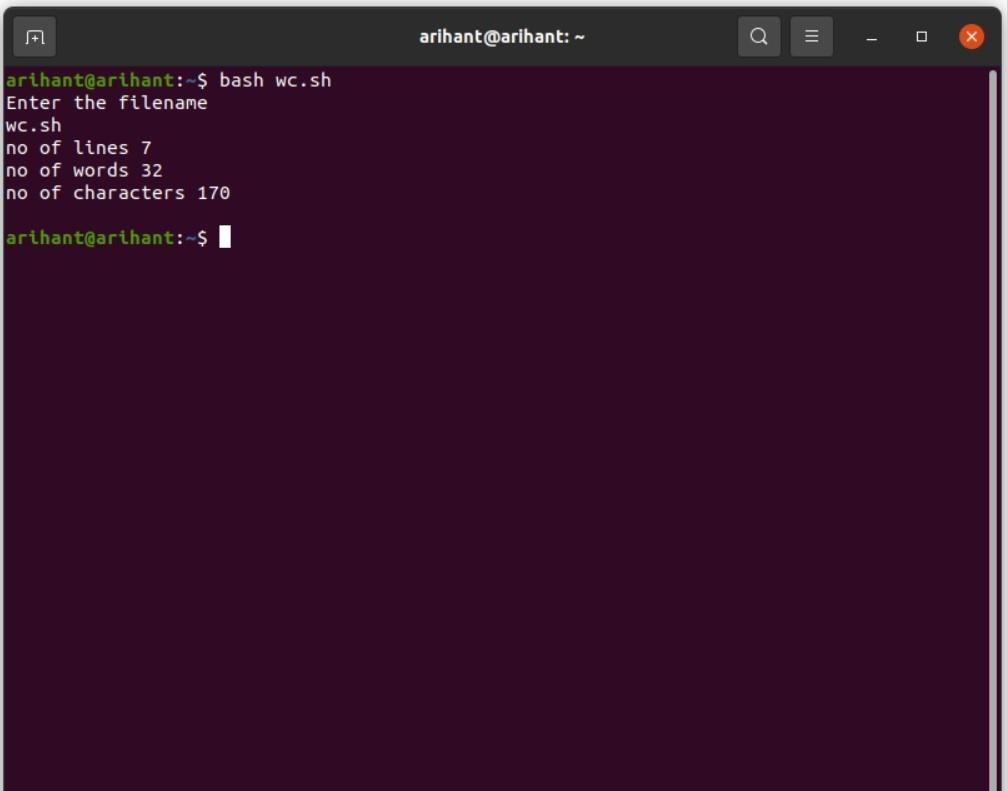
#!/bin/sh

echo "Enter the filename " read fname

l=`wc -l < $fname` w=`wc -w < $fname` c=`wc -m < $fname`

echo -e "no of lines $l\nno of words $w\nno of characters $c\n"

**OUTPUT**



**LAB PROGRAM 17**

**Q.** Write a C/C++ program to that outputs the contents of its Environment list

**PROGRAM**

CODE:

#include <stdio.h>

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

{

int i;

char \*\*ptr;

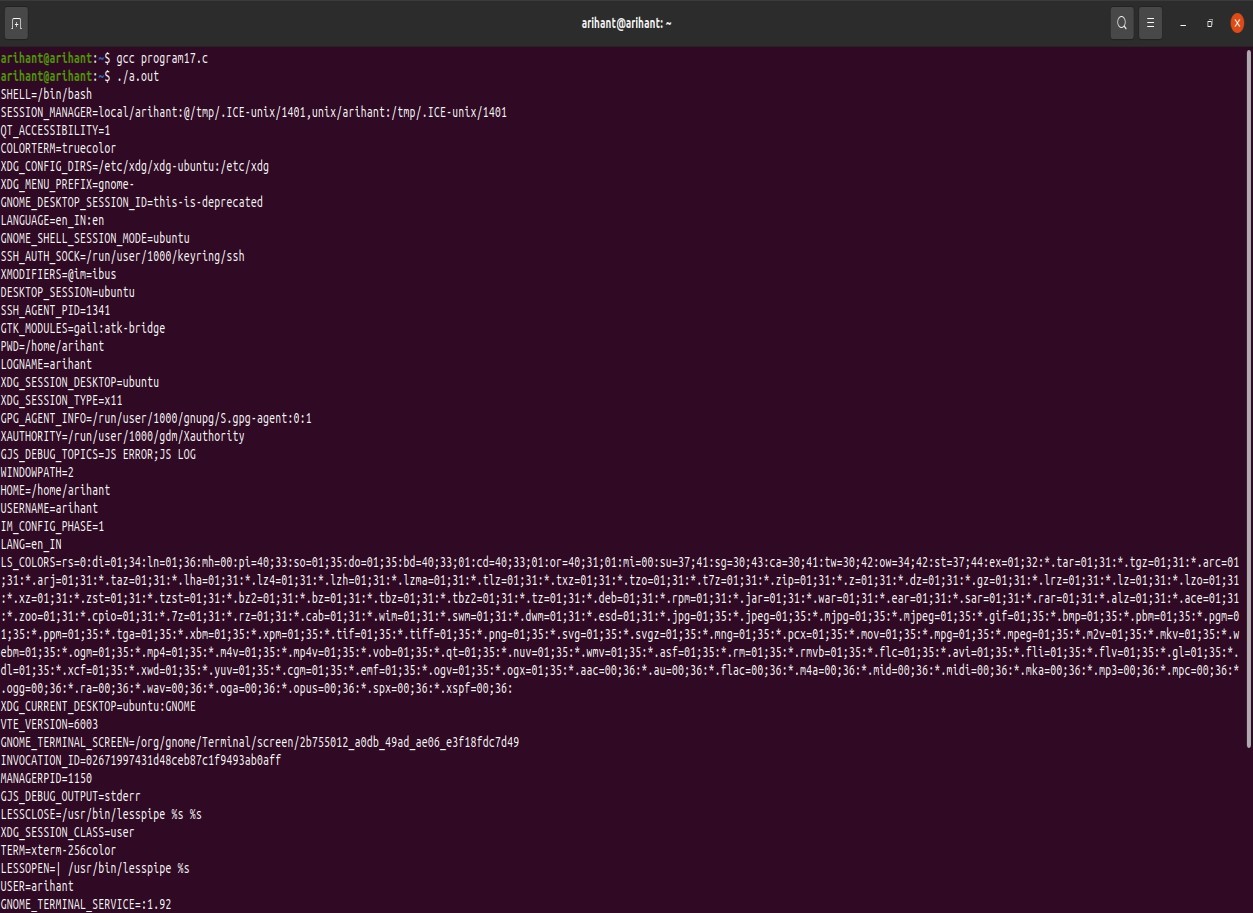
extern char \*\*environ;

for( ptr = environ; \*ptr != 0; ptr++ ) /\*echo all env strings\*/ printf("%s\n", \*ptr);

return 0;

}

**OUTPUT:**



**LAB PROGRAM 18**

**Q.** 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

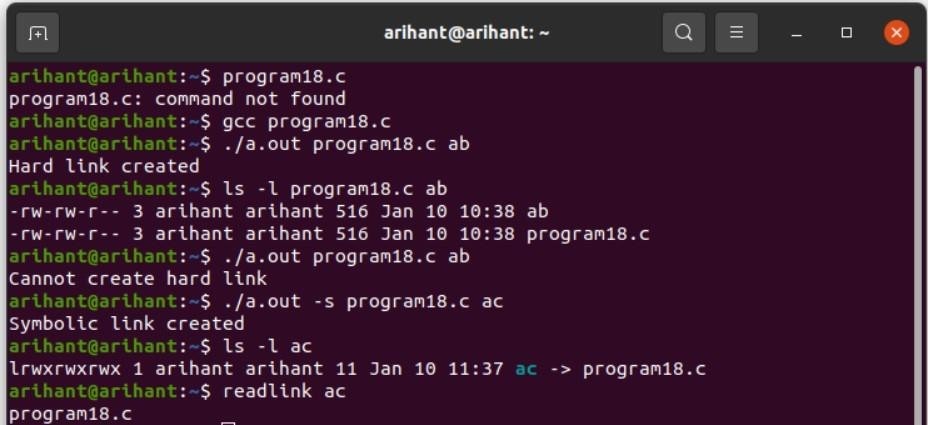
printf("Hard link created\n");

}

return 0;

}

**OUTPUT:**



**LAB PROGRAM 19**

**Q.** 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()

{

#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

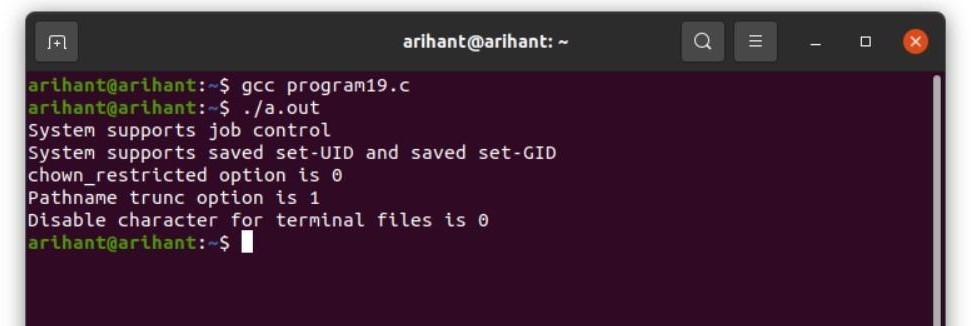
printf("Disable character for terminal files is %d\n",\_POSIX\_VDISABLE); #else

printf("System does not support \_POSIX\_VDISABLE \n"); #endif

return 0;

}

**OUTPUT:**



**LAB PROGRAM 20**

# **Q**. 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

# **PROGRAM**

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

#include<fcntl.h> #include<sys/stat.h> #include<string.h> #include<errno.h> #include<stdio.h>

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 0;

}

mkfifo(argv[1],S\_IFIFO | S\_IRWXU | S\_IRWXG | S\_IRWXO ); if(argc == 2) //reader process

{

fd = open(argv[1], O\_RDONLY|O\_NONBLOCK); while(read(fd, buf, sizeof(buf)) > 0) printf("%s",buf);

}

else

{

fd = open(argv[1], O\_WRONLY); write(fd,argv[2],strlen(argv[2]));

}

close(fd);

}

**OUTPUT:**

