



B M S. COLLEGE OF ENGINEERING (Autonomous Institution)

RECORD OF PRACTICAL WORK

NAMEBhavana.C.K.....
SUBJECTU.S.P.-lab.....
SEMESTER5 th
ROLL NOIBM20CS403USN :.....
BRANCHCSE.....

Particulars of the Experiments Performed

CONTENTS

Expt No.	Date	Experiment	Marks Obtained	Page No.
1	25/10/21	Shell script to find if the given year is leap or not		1
2	25/10/21	Shell script to find the area of a circle		2
3	25/10/21	Shell script to check whether the number is zero/positive/negative		3
4	25/10/21	Shell script to find the biggest of three numbers		4
5	8/11/21	Shell script to find the factorial of a number		5
6	8/11/21	Shell script to compute the gross salary of an employee		6
7	8/11/21	Shell script to convert the temperature Fahrenheit to Celsius		7
8	8/11/21	Shell script to perform arithmetic operations on given two numbers		8
9	15/11/21	Shell script to find sum of even numbers upto n		9
10	15/11/21	Shell script to print combinations of numbers 123		10
11	15/11/21	Shell script to find power of a number		11
12	15/11/21	Shell script to find the sum of n natural numbers		12

Particulars of the Experiments Performed

CONTENTS

Expt No.	Date	Experiment	Marks Obtained	Page No
13		8 shell script to display the pass class of a student		13-14
14		shell script to find the Fibonacci series upto n		15
15		shell script to count the number of vowels of a string		16
16		shell script to check number of lines, words, characters in a file		17
17		Write a C/C++ pgm to list o/p's the contents of its environment list		18
18		Write a C/C++ pgm to emulate the Unix In command		19-20
19		Write a C/C++ POSIX compliant pgm that prints the POSIX defined configuration options supported on any given system using feature test macros		21-22
20		Write a C/C++ pgm which demonstrates Interprocess communication b/w a reader process and a writer process. Use mififo, open, read, write and close api's in your pgm		23-24

Program - 01

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

echo enter the year

read leap

if [\$(((leap%400)) -eq 0)] ; then

echo "entered year is leap year"

elif [\$(((leap%4)) -eq 0)] ; then

echo "entered year is leap year"

elif [\$(((leap%100)) -ne 0)] ; then

echo "entered year is not a leap year"

else

echo "entered year is leap year"

fi

Output

Enter the year

2024

This year is a leap year

Enter the year

2001

This year is not a leap year

Program - 02

Shell Script to find the area of a circle

echo Radius:

read r

pi=3.142

area=`echo \$pi*\$r*\$r | bc`

echo Area of the circle:

echo \$area

Output

Radius : 2.5
Area of the circle is 12.56

Program - 03

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

```
echo Enter a Number:  
read no  
if [ $no -eq 0 ]  
then  
echo "The Number is 0"  
elif [ $no -gt 0 ]  
then  
echo "The Number is Positive"  
else  
echo "The Number is Negative"  
fi
```

Output:

Enter a Number

-50

The Number is Negative

Enter a Number

10

The Number is positive

Enter a Number

0

The Number is zero

Program - 04

Shell script to find the biggest of three numbers

```
echo "enter the first number"
```

```
read a
```

```
echo "enter the second number"
```

```
read b
```

```
echo "enter the third number"
```

```
read c
```

```
if [ $a -gt $b -a $a -gt $c ]
```

```
then
```

```
echo "$a is biggest number"
```

```
elif [ $b -gt $a -a $b -gt $c ]
```

```
then
```

```
echo "$b is biggest number"
```

```
else
```

```
echo "$c is biggest number"
```

```
fi
```

Output:

Enter first number

2

Enter second number

3

Enter third number

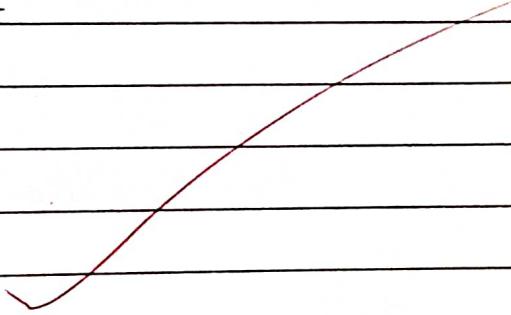
5

5 is biggest number

Program - 05

Shell script to find the factorial of a number

```
echo "enter a number"
read num
fact = 1
while [ $num -gt 1 ]
do
    fact=$((fact * num))      # fact = fact * num
    num=$((num - 1))          # num = num - 1
done
echo $fact
```



Output:

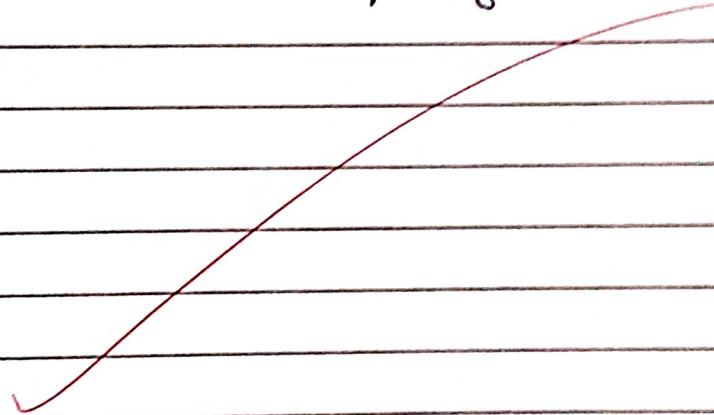
Enter a number

3
→ 6

Program - OG

Shell script to compute the gross salary of an employee

```
echo "Enter the basic salary of the employee"
read basic
da=`echo 0.1 * $basic | bc`
hra=`echo 0.2 * $basic | bc`
gross=`echo $basic + $da + $hra | bc`
echo "Gross salary : $gross."
```



Output:

Enter the basic salary of employee

1000

Gross salary : 1300.0

Program - 07

Shell script to convert the temperature Fahrenheit to Celsius

```
echo "enter the temperature in fahrenheit"
read temp
t = `echo "scale=4; 5/9;" | bc`
cel = `echo $((temp-32)) | * $t | bc`
echo "Temperature in celsius : $cel"
```

Output:

Enter the temperature in fahrenheit

32

Temperature in celsius : 0

Program - 08

Shell script to perform arithmetic operations on given two numbers

```
echo "enter two number:"
```

```
read a
```

```
read b
```

```
echo "enter choice:"
```

```
echo "1. Addition"
```

```
echo "2. Subtraction"
```

```
echo "3. Multiplication"
```

```
echo "4. Division"
```

```
read ch
```

Shell script to perform arithmetic operations on 2 no.

```
case $ch in
```

```
1) res = `echo $a + $b | bc`
```

```
;;
```

```
2) res = `echo $a - $b | bc`
```

```
;;
```

~~```
3) res = `echo $a * $b | bc`
```~~~~```
;;
```~~~~```
4) res = `echo "scale=2; $a / $b" | bc`
```~~~~```
;;
```~~

```
esac
```

```
echo "Result : $res"
```

Output:

Enter two numbers

6

enter choice

1. Addition
2. Subtraction
3. Multiplication
4. Division

1

Result : 9

Program - 09

Shell script to find the sum of even numbers upto n

```
#!/bin/bash
echo enter the number
read n
i=2
while [ $i -lt $n ]
do
sum=$((sum+i))
i=$((i+2))
done
echo $sum
```

Output:

Enter the number

7

12

Program - 10

Shell script to print the combinations of numbers
123

```
for i in 1 2 3
do
for j in 1 2 3
do
for k in 1 2 3
do
if [ $i -ne $j -a $j -ne $k -a $i -ne $k ]
then echo "$i$j$k"
fi
echo $i $j $k
done
done
done
```

Output:

1 3 3 output

2 1 1

2 1 2 output at begin

2 1 3

2 1 3

2 2 1 output at begin

2 2 2

2 2 3

2 3 1

2 3 1 output

2 3 2

2 3 3 output

3 1 1 output

1

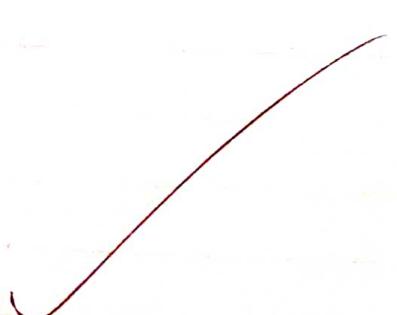
1

1

1

3 3 2

3 3 3



Program - 11

Shell script to find the power of a number

```
echo "enter a number and power"
```

```
read a
```

```
read b
```

```
p=$b
```

```
res=1
```

```
while [ $b -gt 0 ]
```

```
do
```

```
res=`echo "$res*$a" | bc`
```

```
b=`echo "$b-1" | bc`
```

```
done
```

```
echo "result is $res"
```

Output:

Enter a number and power

2

6

result is 64

Program - 12

shell script to find the sum of n natural numbers

```
echo "enter size(N)"
read N
i=1
sum=0
echo "enter Numbers"
while [ $i -le $N ]
do
    read num      #get number
    sum=$((sum+num)) #sum+=num
    i=$((i+1))
done
echo $sum
```

Output:

Enter size (N)

5

Enter numbers

3

8

55

2

1

69

Program - 13

Shell script to display the pass class of a student

```
pass = 0
```

```
fail = 0
```

```
for ((i=0; i<6; i++))
```

```
do
```

```
echo "Enter your cie marks out of 50"
```

```
read cie
```

```
echo "Enter your see marks out of 50"
```

```
read see
```

```
total = $((cie+see))
```

```
if [ $total -gt 90 ]
```

```
then echo "S grade"
```

```
pass = $((pass+1))
```

```
elif [ $total -gt 80 ]
```

```
then echo "A grade"
```

```
pass = $((pass+1))
```

```
elif [ $total -gt 70 ]
```

```
then echo "B grade"
```

```
pass = $((pass+1))
```

```
elif [ $total -gt 60 ]
```

```
then echo "C grade"
```

```
pass = $((pass+1))
```

```
elif [ $total -gt 50 ]
```

```
then echo "D grade"
```

```
pass = $((pass+1))
```

```
elif [ $total -gt 40 ]
```

```
then echo "E grade"
```

```
pass = $((pass+1))
```

```
else
```

```
echo "Fail"
```

```
fail = $((fail+1))
```

```
fi
```

```
done
```

```
echo "Number of subjects passed : $pass"
```

```
echo "Number of subjects failed : $fail"
```

Output :

Enter your cie marks out of 50

23

Enter your see marks out of 50

23

E grade

Enter your cie marks out of 50

12

Enter your see marks out of 50

12

Fail

Enter your cie marks out of 50

45

Enter your see marks out of 50

35

B Grade

Program-14

Shell script to find the Fibonacci series up to n

```
echo "Enter the value of n"
```

```
read n
```

```
a=0
```

```
b=1
```

```
for ((i=0; i<n; i++))
```

```
do
```

```
echo "$a"
```

```
fn=$((a+b))
```

```
a=$b
```

```
b=$fn
```

```
done
```

Output:

Enter the value of n

3

0

1

1

Enter the value of n

6

0

1

1

2

3

5

Program - 15

Shell script to count the number of vowels of a string

```
#!/bin/bash
echo "Enter a string"
read string
count = 0
l = `expr "$string": '.*'`
for (( i=0; i<${#l}; i++ ))
do
    c = `expr "$string": '\(.*)'`"
    if [ "$c" = 'a' -o "$c" = 'e' -o "$c" = 'i' -o "$c" = 'o' -o "$c" = 'u' ]
    then
        count = $((count + 1))
    fi
    string = `expr "$string": '\.*\)'`"
done
echo "The number of vowels: $count"
```

Output:

Enter a string

asdtfghjkl

1

Enter a string

aoive

5

Enter a string

Bharava

3

Program - 16

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

```
echo "Number of lines:"  
cat program.sh | wc -l  
echo "Number of words:"  
cat program.sh | wc -w  
echo "Number of characters:"  
cat program.sh | wc -c
```

Output:

Number of lines:
15

Number of words:
55

Number of characters:
312

Program 14

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

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

Output

SHELL = /bin/bash

SESSION-MANAGER = local /bmscecs - HP-Pro-3330-MT:@/tmp/

QT-ACCESSIBILITY = 1

COLORTERM = truecolor

GDMSESSION = ubuntu

DBUS-SESSION-BUS-ADDRESS = unix: path=/run/user/wool

-= ./a.out

Program - 18

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

```
#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
    }
}
```

Teacher's Signature : _____

Expt. No. _____

Date _____

Page No. 20

```
printf ("Hard link created\n");
}
return 0;
}
```

Teacher's Signature : _____

Output:

./a.out prog18.c z
Hard link created

ls -l

total 212

-rwx-rw-r-- 1 bmscece bmscece 1649 Oct 25 19156 a1
-rwxrwxr-x 1 bmscece bmscece 15808 Jan 3 15:22 a2

:

:

-rwx-rw-r-- 1 bmscece bmscece 1514 Jan 3 15:22 a2

./a.out prog18.c z
Cannot create hard link

./a.out prog18.c zz

Hard link created

./a.out -s prog18.c zz

Cannot create symbolic link

Program-19

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

```
#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 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_REST
           -RICTED);
    #else
    printf("system does not support chown-restricted
           option\n");
    #endif
```

Teacher's Signature : _____

```
#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
#ifndef _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:

\$gcc prog19.c -o prog19

\$./prog19

System supports job control

System supports saved set-UID and saved set-GID

chown-restricted option is 1

Pathname trunc option is 1

Disable character for terminal files is 0

Program-20

Write a C/C++ program which demonstrates Interprocess Communication between a reader process and a writer process. Use mknod, open, read, write and close apis in your 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;
    }
    mknod(argv[1], S_IFIFO | S_IRWXU | S_IRWXG | S_IRWXO);
    if(argc == 2)
    {
        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:

Terminal -1 : Write Process

\$ gcc prog20.c -o prog20

\$./prog20

USAGE ./prog20 <File> [<args>]

\$./prog20 pipeNam "Good Morning"

[waits for reorder process]

\$ bmsce@bmsce:\$

[ends after reader]

Terminal - 2 : Reorder Process

\$./prog20 pipeNam

Good Morning bmsce@bmsce:\$