Output :

1) enter year : 1800
1800 is not a leap year.

2) Enter year : 1300
1300 is not a leap year

3) enter year : 1600
1600 is a leap year

4) enter year : 2016
2016 is a leap year

Date 28/10/21

Expt. No. 1 Page No. 1

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

```
#!/bin/sh
echo " enter year : \c "
read year
if [ `expr $year %100` -eq 0 ]
then
        if [ `expr $year %400` -eq 0 ]
        then
            echo " $year is a leap year"
        else
            echo " $year is not a leap year"
        fi
elif [ `expr $year %4` -eq 0 ]
then
    echo " $year is a leap year"
else
    echo " $year is not a leap year"
fi
```

Teacher's Signature :

Output:

```
enter the radius of the circle : 4
area of the circle is : 50.24
```

Date 28/10/21

Expt. No. 2 Page No. 2

Shell script to find the area of the circle.

```
#!/bin/sh
echo " enter the radius of the circle : \c"
read radius.
pi = 3.14
echo " area of the circle is : \c"
echo " $pi * $radius * $radius " | bc
```

Teacher's Signature :

Output:

1) enter a number
-4
-4 is a negative number.

2) enter a number
5
5 is a positive number.

3) enter a number
0
0 is zero

Date 25/10/21

Expt. No. 3 Page No. 3

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

```
#!/bin/sh
echo "enter a number"
read n
if [ $n -lt 0 ]
then
    echo "$n is a negative number"
elif [ $n -eq 0 ]
then
    echo "$n is zero"
else
    echo "$n is a positive number"
fi
```

Teacher's Signature :

Output:

enter three numbers.
3 8 6
8 is the largest

Date 25/10/21

Expt. No. 4 Page No. 4

Shell script to find the biggest of three numbers.

```
echo " enter three numbers "
read n1 n2 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 the largest"
fi
```

Teacher's Signature :

Output :

enter a number = 4
factorial = 24

Date 8/11/21

Expt. No. 5 Page No. 5

Shell script to find the factorial of a number

```
#!/bin/sh
echo "enter a number : \c"
read n
fact = 1
while [ $n -gt 1 ]
do
    fact = `echo "$fact * $n" |bc`
    n = `echo "$n-1" |bc`
done
echo "factorial = $fact"
```

Teacher's Signature :

Output:

```
enter the basic salary
1000
hra = 200
da = 100
gross = 1300
```

Date 8/11/21

Expt. No. 6 Page No. 6

Shell script to compute the gross salary of an employee.

```
#!/bin/sh
echo "enter the basic salary"
read sal
hra = $((sal * 20/100))
echo "hra = $hra"
da = $((sal * 10/100))
echo "da = $da"
gross = $((sal + hra + da))
echo "gross = $gross"
```

Teacher's Signature :

Output :

enter the fahrenheit temperature
100
100 degree fahrenheit = 37.7777 degree celsius

Date 8/11/21

Expt. No. 7 Page No. 7

Shell script to convert the temperature Fahrenheit to Celsius.

```
#!/bin/sh
echo "enter fahrenheit temperature"
read f
c= `echo "scale=4 ; ($f -32)/1.8" | bc`
echo "$f degree fahrenheit = $c degree celsius"
```

Teacher's Signature :

Output:

```
enter two numbers
read num1 num2 3 2
echo "Please enter your choice:
 1.Addition
 2.Subtraction
 3. Multiplication
 4. Division
 5. Modulus
4
division is 1.50
```

Date 8/11/21

Expt. No. 8 Page No. 8

Shell script to perform arithmetic operations on given two numbers.

```
echo "enter two numbers"
read num1 num2
echo " Please enter your choice :\n 1. Addition \n 2.Subtraction \n
    3. Multiplication \n 4. Division \n 5. Modulus \n"
read option
case $option in
        1) echo "Sum is $((num1+ num2))" ;;
        2) echo "Difference is $((num1- num2)) ;;
        3) echo " Product is $(( num1 *num2))" ;;
        4) ans = `echo "scale=2; $num1/$num2" | bc`
            echo "division is $ans" ;;
        5) echo " Modulus is $(( num1 % num2))" ;;
        *) echo " Enter a valid number" ;;
esac
```

Teacher's Signature :

Output :

```
enter n
10
The sum of even numbers upto $n is 10 is 30
```

Date 15/11/21

Expt. No. 9 Page No. 9

Shell script to find sum of even numbers upto n

```
#!/bin/sh.

echo "enter n"
read n
i=1
sum=0
while [ $i -le $n ]
do
    if [ $((i%2)) -eq 0 ]
    then
        sum= $((sum+i))
    fi
    i= $((i+1))
done
echo " The sum of even numbers upto $n is $ sum"
```

Teacher's Signature :

Output:

```
111
112
113
121
122
123
131
132
133
211
212
213
221
222
223
231
232
233
311
312
313
321
322
323
331
332
333
```

Date 15/11/21

Expt. No. 10 Page No. 10

Shell script to find/print the combinations of numbers 123

```
#!/bin/sh
echo " Combinations of 123
for i in 1 2 3
do
    for j in 123
    do
        for k in 123
        do
            echo $i $j $k
        done
    done
done
```

Teacher's Signature :

Output:

enter the number
4
enter power
2
The result is 16

Date 15/11/21

Expt. No. 11 Page No. 11

Shell script to find the power of a number

```
#!/bin/sh
echo " enter the number "
read num
echo " enter power"
read pow
res=1
while [ $pow -ne 0 ]
do
        res=$((res*num))
        pow=$((pow-1))
done
echo " The result is $res"
```

Teacher's Signature :

Output:

```
enter n
4
The sum of n natural numbers upto n is 10
```

Date 15/11/21

Expt. No. 12

Page No. 12

Shell script to find sum of n natural numbers.

```
#!/bin/sh
echo "enter n"
read n
i=1
sum=0
while [ $i -le $n ]
    do
        sum=$((sum+i))
        i=$((i+1))
    done
echo "The sum of n natural numbers upto n is $sum"
```

Teacher's Signature :

Output:

enter the marks of student in subject1
70
enter the marks obtained in subject2
80
enter the marks obtained in subject3
90
Grade B

Date 29/11/21

Expt. No. 13 Page No. 13

Shell script to display the pass class of a student

```
#!/bin/sh
echo " enter the marks obtained in student1"
read m1
echo " enter the marks obtained in subject 2"
read m2
echo "enter the marks obtained in subject 3"
read m3
sum = `expr $m1+$m2+$m3`
avg = `expr $sum/3`
if [$avg -gt 90]
then
        echo "Grade S"
elif [$avg -gt 80 -a $avg -le 90]
then
        echo " Grade A"
elif [ $avg -gt 65 -a $avg -le 80]
then
        echo "Grade B"
elif [$avg -gt 40 -a $avg -le 65]
then
        echo " Grade C"
else
        echo "Fail!"
fi
```

Teacher's Signature :

Output:

```
Enter positive value
5
Fibonacci Series
0
1
1
2
3
```

Date 29/11/21

Expt. No. 14 Page No. 14

Shell script to find the Fibonacci series upto n

```
#!/bin/sh
a1=0
a2=1
echo " enter a positive value"
read n
echo " \n fibonacci series"
if [$n -eq 1]
then
        echo "$a1"
elif [$n -eq 2]
then
        echo "$a1 \n $a2"
else
        echo " $a1 \n $a2"
        while [$n -gt 2]
        do
                a3 = `expr $a1 + $a2`
                echo "$a3"
                a1 = $a2
                a2 = $a3
                n = `expr $n -1`
        done
fi
```

Teacher's Signature :

Output:

Enter the string
great
length of string = 5
No. of vowels = 2

Date 29/11/21

Expt. No. 15 Page No. 15

Shell script to count the number of vowels in a string

```
#!/bin/sh
echo "enter the string"
read str
len = $(expr len $str)
echo " length of the string = " $len
vowel=0
while [ $len -gt 0]
  do
    ch = $ (echo $str | cut -c $len)
      case $ch in
          ([aeiouAEIOU]) vowel = $((vowel+1));;
      esac
      len= $((len-1))
    done
  echo " No. of vowels = $vowels"
```

Teacher's Signature :

Output:

Enter filename
vowel.sh
Lines = 15
Characters = 301
Words = 49

Date 29/11/21

Expt. No. 16

Page No. 16

Shell script to check no. of lines, words, characters in a file.

```
#!/bin/sh
echo "enter the no. of lines, characters & words in a file"
echo "enter the filename"
read fname
lines=`wc -l < $fname`
characters=`wc -c < $fname`
words=`wc -w < $fname`
echo "Lines= $lines"
echo "Characters = $characters"
echo "Words = $words"
```

Teacher's Signature :

Expt. No. 17 Date 3/1/22

Page No. 17

Write a c/c++ program to that outputs the content 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;
}
```

Teacher's Signature :

Write a program to emulate Unix ln command in C/C++.

```
#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 file>\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;
}
```

Date 3/1/22

Expt. No. 19

Page No. 19

Q. Write a program in C/C++ to print posix defined configuration options.

```
#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 & set-GID\n");
  #else
      printf("System does not support saved set of UID & 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
```

Teacher's Signature :

Date 3/1/22

Expt. No. 19

Page No. 20

```
#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 file is
                                %d\n", _POSIX_VDISABLE);
#else
    printf(" System does not support _POSIX_VDISABLE \n");
#endif
return 0;
}
```

Teacher's Signature :

Date 3/1/22

Expt. No. 20 Page No. 21

Write a C/C++ program to demonstrate Interprocess communication between a reader process & a writer process.

```
#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 buff[254];
    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)
    {
        fd = open(argv[1], O_RDONLY | O_NONBLOCK);
        while (read (fd, buff, sizeof (buff)) > 0)
            print("%s", buff);
    }
```

Teacher's Signature :

```
else
{
    fd = open(argv[1], O_WRONLY);
    write(fd, argv[2], strlen(argv[2]));
}
    close(fd);
}
```