1. Write a script to read any 2 floating values and find the sum, difference, quotient, and remainder.

```
echo "enter two numbers"
read a b
sum=`echo $a + $b | bc`
echo "sum: $sum"
diff=`echo $a - $b | bc`
echo "difference: $diff"
pro=`echo $a % $b | bc`
echo "remainder: $pro"
div=`echo $a / $b | bc`
echo "division: $div"
```

2. Write a script to read the length and breadth of a rectangle and radius of a circle and calculate the area and perimeter of the rectangle and area and circumference of the circle.

```
echo "Enter length of rectangle:"
read length
echo "Enter breadth of rectangle:"
read breadth
echo "Enter radius of circle:"
read radius
```

```
rect_area=`echo "$length * $breadth" | bc`
rect_perimeter=`echo "2 * ($length + $breadth)" | bc`
circle_area=`echo "3.14159 * $radius * $radius" | bc`
circle_circumference=`echo "2 * 3.14159 * $radius" | bc`
echo "Rectangle area: $rect_area"
echo "Rectangle perimeter: $rect_perimeter"
echo "Circle area: $circle_area"
echo "Circle circumference: $circle_circumference"
```

3. Write a shell program to find sum of digits of a number

```
echo "enter the number "
read n
sum=0

while [ $n -gt 0 ]
do
mod=$((n%10))
sum=$((sum+mod))
n=$((n/10))
done
```

```
echo "sum=$sum"
```

4. Write a shell program to find reverse of the number

```
echo Enter Number :

read n
num=0
while [ $n -gt 0 ]
do
num=$(expr $num \* 10)
k=$(expr $n % 10)
num=$(expr $num + $k)
n=$(expr $n / 10)
done

echo Reversed Number is $num
```

5. Write a shell program to determine whether the given number is a palindrome or not.

```
echo "Enter Number: "
read num
s=0
rev=0
temp=$num
```

```
while [ $num -gt 0 ]
do
s=$(( $num % 10 ))
num=$(( $num / 10 ))
rev=$( echo ${rev}${s} )
done
if [ $temp -eq $rev ];
then
echo "Number is palindrome"
else
echo "Number is NOT palindrome"
fi
```

6. Write a shell script to display the digits which are in odd positions in a given integer.

```
echo "enter digit number"
read num
c=1

echo "odd numbers"
while [ $c -le 5 ]
do
```

```
d= echo $num| cut -c $c
echo $d
c=$((c+2))
done
```

7. Write a script to read the basic salary of n employees and calculate the gross salary

If BP<15000, DA=30% of BP, HRA =Rs 500. TA = 10% of BP.

If BP >=5000, DA=50% of BP, HRA=15%, TA=1000.

```
echo "Enter the number of employees:"
read n
for (( i=1; i<=n; i++ ))
do
echo "Enter the basic salary of employee:"
read basic_salary
if [ $basic_salary -lt 15000 ]
then
DA=`echo $basic_salary*90/100 | bc`
HRA=500
TA=`echo $basic_salary*10/100 | bc`
else
```

```
DA=`echo $basic_salary*50/100 | bc`
HRA=`echo $basic_salary*15/100 | bc`
TA=1000
fi

gross_salary=`echo
$basic_salary+$DA+$HRA+$TA | bc`
echo "Gross salary of employee:
$gross_salary"
done
```

8.Write a script to read the cost and selling price of an item and to decide how much loss or profit has incurred by the seller.

```
echo "Enter the cost price of the item: "
read cost_price
echo "Enter the selling price of the item: "
read selling_price
if [ $cost_price -gt $selling_price ]
then
loss=$((cost_price - selling_price))
echo "You have incurred a loss of $loss"
elif [ $cost_price -lt $selling_price ]
then
```

```
profit=$((selling_price - cost_price))
echo " You have made a profit of $profit"
else
echo "You have neither made a profit nor incurred a loss."
fi
```

9. Write a script to read 5 marks of n students. Find the total and return distinction if the total percentage >= 80. [Distinction] if total % is >=60 and <80 [first class].if total % is >=50 and <60 [second class] else print failed [<50].

```
echo "enter the number of students"
read n
for i in `seq 1 1 $n`
do
echo "enter first subject mark"
read m1
echo "enter second subject mark"
read m2
echo "enter third subject mark"
read m3
echo "enter fourth subject mark"
```

```
read m4
echo "enter fifth subject mark"
read m5
total=$(($m1+$m2+$m3+$m4+$m5))
perc=$(($total/5))
if [ $perc -ge 80 ]
then
echo "Distinction"

elif [ $perc -ge 60 -a $perc -lt 80 ]
then

echo "First class"

elif [ $perc -ge 50 -a $perc -lt 60 ]
then
echo "Second Class"
else
echo "Failed"
fi
done
```

10. Write a script to prepare a multiplication table of a given number to any order.

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

```
read n
for i in `seq 1 1 10`
do
mul=$((n*i))

echo "$i*$n=$mul"
done
```

11. Write a script to find the value of one number raised to the power

```
echo "enter the number"
read num

echo "enter the power"
read pow
count=0
res=1
while [ $count -ne $pow ]
do
res=$(($res*$num))
count=$(($count+1))
done
echo "$num^$pow is:$res"
```

12. Write a script to print all prime numbers from 1 to n.

```
echo "Enter the limit: "
read n
echo "Prime numbers from 1 to $n:"
for ((i=2; i<=n; i++))
do

flag=0

for ((j=2; j<i; j++))
do
if [ $((i%j)) -eq 0 ]
then
flag=1
break
fi
done
if [ $flag -eq 0 ]
then
echo $i
fi
done
```

13. Write a shell script to sum up the following series

1/1! + 2/2! + 3/3! + ….

```
echo "enter the number"
read n
fact=1
sum=0
for((i=1;i<=$n;i++))
do
fact=`echo $fact \* $i|bc -l`
res=`echo $i / $fact|bc -l`
sum=`echo $sum +$res|bc -l`
done
echo "result=$sum
```

14. Write a script to read a year and to decide whether it is a leap year or not.

```
echo "enter the year"
read year
if [ $(($year%400)) -eq 0 -o $((year%4)) -eq 0 ]
then
echo "$year is a leap year"
else
echo "$year is not a leap year"
fi
```

15. Shell script to perform operations like display, list, make directory and copy, rename, delete, edit file.

```
echo enter the directory name to be created
read n
mkdir $n
a= pwd
echo $a
ls -l
echo enter the file to be renamed
read n
echo enter the name
read w
mv $n $w
echo enter the filename to be added
read d
cat >> $d
```

16. Write a menu driven program to display the following options. Contents of /etc/passwd List of output of ‘who’ Present working directory Exit

```
while true
do
echo “1. contents of/etc/passwd”
```

```
echo “2. List of users who have logged in”
echo “3. Present working director”
echo “4. Exit”

echo “Your choice?”
read ans
case $ans in

1) cat /etc/passwd;;

2) who;;

3) pwd;;

4) exit;;
esac
echo “ Press any key to continue...”
read key &gt;/dev/null
done
```

17. Write a script to accept a filename while running the script and check it has the write permission, if yes prompt the user to enter a

text and append the text to the given filename.

```
echo "Enter filename:"
read fn1
flag=0

if test -w $fn1
then
echo "$fn1 has write permission"
flag=1
else

echo "$fn1 has no write permission"
fi
if [ $flag -eq 1 ]
then
echo "Enter a few lines to $fn1:"
read newtext
fi

echo $newtext>>$fn1

echo "Content after appending is"
cat $fn1
```

18. Write a shell script which receives two file names as arguments. It should check whether the two file’s contents are the same or not .If they are the same, delete the second file.

```
if diff -u $1 $2
then
echo "they are same/removing the second\n"
rm -i $2
else

echo "they are different"
fi
```

19. Write a shell script, which will receive any number of filenames as arguments .The shell script should check whether such files already exist.

```
for file in "$@" do

if [ -e "$file" ]
then echo "$file
already exists"
else echo "$file
does not exist"
fi
```

done

20. Write a shell script to sort the given numbers in ascending order using Bubble sort.

```
arr=(10 8 20 100 12)

echo "Array in original order"
echo ${arr[*]}
for ((i = 0;i<5;i++))
do
for((j = 0; j<5-i-1;j++))
do
if [ ${arr[j]} -gt ${arr[$((j+1))]} ]
then
temp=${arr[j]}
arr[$j]=${arr[$((j+1))]}
arr[$((j+1))]=$temp
fi
done
done
echo "Array in sorted order:"
echo ${arr[*]}
```

21. Write a shell program to find the factorial of a number using function.

```
factorial()

{

if [ "$1" -gt "1" ];
then
a=`expr $1 - 1`
b=`factorial $a`
c=`expr $1 \* $b`
echo $c
else
echo 1
fi
}

echo "Enter a number:"
read x
factorial $x
```

22. Write a shell program to determine whether the given string is palindrome or not using function.

```
echo "Enter a string:"
```

```
read input
rev=" " len=${#input}
echo “length $len”
for ((i=$len-1;i>=0;i--))
do
rev=$rev${input:$i:1}
done
if [ $input == $rev ]
then
echo "String is Palindrome"
else
echo "string is not a palindrome"
fi
```

```
#!/bin/bash

# Function to check if a string is a palindrome
is_palindrome() {
    input=$1
    rev=""
    len=${#input}
    for ((i=$len-1;i>=0;i--)); do
```

```
        rev=$rev${input:$i:1}
    done

    if [ "$input" == "$rev" ]; then
        echo "String is Palindrome"
    else
        echo "String is not a palindrome"
    fi
}

# Main script
echo "Enter a string:"
read input
is_palindrome "$input"
```

23. Write a script to rename all c files to cpp files.

```
for file in *.c
do
mv ”$file” “${file%.c}.cpp”
done
```

24. Write a script to receive any number of filenames as arguments and to check whether the arguments supplied is a file or directory. If it is a directory, it should be appropriately

reported. if it is a filename then name of the file as well as the number of lines present in it should be reported.

```
for x in $*
do
if [ -f $x ]
then
echo " $x is a file "

echo " no of lines in the file are "
wc -l $x
elif [ -d $x ]
then
echo " $x is a directory "
else
echo " enter valid filename or directory name
"
fi
done
```

25. Write a script to read from a file which is supplied as a command line argument and count the number of lines and words. If there is no filename supplied, the script should accept text from the keyboard.

```
Lines=$(wc -l < “$1”)
Words=$(wc -w < “$1”)
Echo “Number of lines $lines”
Echo “Number of words: $words”
```

```
#!/bin/bash

# Check if a filename is supplied as a
command-line argument
if [ $# -gt 0 ]; then
   # If a filename is provided, count lines and
words in the file
   if [ -f "$1" ]; then
      # Count lines and words in the file
      lines=$(wc -l < "$1")
      words=$(wc -w < "$1")

      echo "Number of lines: $lines"
      echo "Number of words: $words"
   else
      echo "Error: File '$1' not found."
   fi
else
```

```
    # If no filename is provided, read from the
keyboard input
    echo "Enter text (Press Ctrl+D to end):"
    # Read input from the user until EOF
(Ctrl+D)
    file_content=$(cat)

    # Count lines and words in the keyboard
input
    lines=$(echo "$file_content" | wc -l)
    words=$(echo "$file_content" | wc -w)

    echo "Number of lines: $lines"
    echo "Number of words: $words"
fi
```

26. Write a script to wish the user “Good Morning, Good Afternoon and Good Evening” when he logs in to the system based on the time.

```
hour=$(date +%H)

if [ $hour -ge 0 -a $hour -le 12 ]
then
greet=”good morning, $USER”
```

```
elif [ $hour -ge 12 -a $hour -le 16 ]
then
greet=”good afternoon, $USER”
else
greet=”good evening, $USER”
fi
echo $greet
```

27. Write a script to read a character and to display if it is lowercase, uppercase, digit or special character or not a character.

```
echo "Enter a character: "
read char
if [[ $char == [a-z] ]]; then

echo "The character is a lowercase letter."
elif [[ $char == [A-Z] ]]; then
echo "The character is an uppercase letter."
elif [[ $char == [0-9] ]]; then
echo "The character is a digit."
else
echo "The character is a special character."
fi
```

28. The word “mca” is present in some of the files supplied as arguments. Write a script to

search each of these files, and to stop at the first file containing the word “mca” and report it.

```
for i in $*
do
w=`grep -c "mca" $1`
if [ $w -gt 0 ]
then

echo "first file containing the word mca is $1"
exit
fi
shift
done
```

29. Write a shell script which receives an even number of file names. Suppose four file names are supplied then the first file should get copied into the second file, the third file should get copied into the fourth file, and so on. If odd numbers of file names are supplied then no copying should take place and an error message should be displayed.

```
r=$#
```

```
if [ $((r%2)) -ne 0 ]
then
echo "odd number of files"
else
for((i=0;i<$r;i=i+2))
do
f1=$1
f2=$2
cp $f1 $f2
shift 2
done
echo "copied successfully"
```

30. Write a shell script which displays a list of all files in the current directory to which you have read, write & execute permissions.

```
echo "Files with read, write, and execute permissions:"
echo " "
for file in * do

if [ -r "$file" ] && [ -w "$file" ] && [ -x "$file" ]
then echo "$file"
fi
```

done