Output

```
Enter a number
2
Even number
```


PROGRAM-1

AIM: Write a shell program to find whether the given number is odd or even

```
echo "enter a number"
read n
if test `expr $n % 2` -eq 0
then
echo "Even number"
else
echo "Odd number"
fi
```

Output

Enter the radius
2.
Area : 12.56
Circumference : 12.56

PROGRAM - 2

AIM: Write a shell program to find the area and circumference of a circle.

```
echo "Enter the radius"
read r
area=$(echo 3.14 \* $r \* $r | bc)
cir=$(echo 2 \* 3.14 \* $r | bc)
echo "Area : $area"
echo "circumference : $cir"
```

Output

Enter a number
151

The given number and its reverse are same

PROGRAM-3

AIM: Write a shell program to check whether the given number and its reverse are same.

```
echo "Enter a number"
read n
t=$n
s=0
while test $n -gt 0
do
r='expr $n % 10'
s='expr $r + $s \* 10'
n='expr $n / 10'
done
if test $s -eq $t
then
echo "The given number and its reverse are same"
else
echo "The given number and its reverse are not same"
fi
```

Output

```
Enter a string:
Malayalam
String is palindrome
```


PROGRAM - 4

AIM: Write a shell program to check the given string is palindrome or not.

```
len=0
i=1
echo -n "enter a string:"
read s
len='echo $s | wc -c'
len='expr $len - 1'
halflen='expr $len / 2'
while test $i -le $halflen
do
c1='echo $s | cut -c $i'
c2='echo $s | cut -c $len'
if test $c1 -ne $c2
then
echo "string is not palindrome"
exit
fi
i='expr $i + 1'
len='expr $len - 1'
done
echo "string is palindrome"
```

Output

```
Enter a number
407
407 is an amstrong number
```


PROGRAM-5

AIM: Write a shell program to check the given integer is amstrong or not.

```
echo "Enter a number"
read n
t=$n
s=0
while test $n -gt 0
do
r='expr $n % 10'
s='expr $s + $r \* $r \* $r'
n='expr $n / 10'
done
if test $s -eq $t
then
echo "$t is an amstrong number"
else
echo "$t is not an amstrong number"
fi
```

Output

Enter a number
2
2 is a prime number

PROGRAM-6

AIM: Write a shell program to check the given integer is prime or not.

```
echo "Enter a number"
read a
i=2
z=0
while test $i -lt $a
do
s='expr $a % $i'
if test $s -eq $z
then
echo "not a prime number"
exit
else
i='expr $i + 1'
fi
done
echo "$a is a prime number"
```

Output

Enter a number
123
the sum of square of individual digits of 123
is 14

PROGRAM - 7

AIM: Write a shell program to find the sum of squares of individual digits of a number.

```
echo "Enter a number"
read n
t = $n
s = 0
while test $n -gt 0
do
r = 'expr $n % 10'
s = 'expr $s + $r \* $r'
n = 'expr $n / 10'
done
echo "the sum of square of individual digits of
      $t is $s"
```

Output

1 - who am I?
2 - who is logged on?
3 - date
4 - Calender
enter your choice:
3
Thu Mar 14 11:47:23 1st 2024

PROGRAM - 8

Aim: Write a shell program to execute various linux commands using case statement.

```
echo "1-who am I?"
echo "2-who is logged on?"
echo "3-date"
echo "4-calender"
echo "enter your choices"
read n
case $n in
1)
  whoami
  ;;
2)
  who
  ;;
3)
  date
  ;;
4)
  cal
  ;;
esac
```

Output

Enter a string to find the number of vowels
hello
Number of vowels in the string is 2

PROGRAM - 9

AIM: Write a shell program to count the number of vowels in a line of text

```
echo "Enter a string to find the number of vowels"
read st
len='expr $st | wc -c'
len='expr $len - 1'
count=0
while test $len -gt 0
do
ch='expr $st | cut -c $len'
case $ch in
a|A)
   count='expr $count + 1'
   ;;
e|E)
   count='expr $count + 1'
   ;;
i|I)
   count='expr $count + 1'
   ;;
o|O)
   count='expr $count + 1'
   ;;
u|U)
   count='expr $count + 1'
   ;;
```

```
esac
len='expr $len - 1'
done
echo "Number of vowels in the given string is $count"
```

Output

```
Enter student name
Hari
Enter Marks
English
85
Maths
90
Science
90
A grade
```


PROGRAM-10

AIM: Write a shell program to display student grade.

```
echo "Enter student name"
read name
echo "Enter Marks"
echo "English"
read m1
echo "Maths"
read m2
echo "Science"
read m3
total='expr $m1 + $m2 + $m3'
avg='expr $total / 3'
if test $avg -ge 90
then
echo "A grade"
elif test $avg -ge 80
then
echo "B grade"
elif test $avg -ge 70
then
echo "C grade"
else
echo "Better luck next time"
fi
```

Output

```
Enter limit
5
Enter numbers
10
80
30
20
50
Largest number is : 80
```


PROGRAM - 11

AIM: Write a shell program to find the largest number from N numbers.

```
echo "Enter limit"
read N
i=1
max=0
echo "Enter numbers"
while test $i -le $N
do
read num
if test $i -eq 1
then
max=$num
else if test $num -gt $max
then
max=$num
fi
fi
i=$((i+1))
done
echo "Largest number is : $max"
```

Output

```
Enter a number
173
the smallest digit is 1
```

PROGRAM-12

AIM: Write a shell program to find the smallest digit from a number.

```
echo "enter a number"
read n
s=9
while test $n -gt 0
do
r='expr $n % 10'
if test $r -lt $s
then
s=$r
fi
n='expr $n / 10'
done
echo "the smallest digit is $s"
```

Output

```
Enter a number
1258
the single digit is 7
```


PROGRAM-13

AIM: Write a shell program to find the sum of digits of a number until a single digit is obtained.

```
echo "Enter a number"
read n
s=0
while test $n -gt 0
do
r='expr $n % 10'
s='expr $s + $r'
n='expr $n / 10'
if test $n=0 -a $s -gt 9
then
n=$s
s=0
fi
done
echo "the single digit is $s"
```

Output

```
Enter a number
689
the second largest digit is : 8
```


PROGRAM - 14

AIM: Write a shell program to find the second largest digit from a number

```
echo "enter a number"
read n
a=0
b=0
while test $n -gt 0
do
r='expr $n % 10'
if test $r -gt $a
then
b=$a
a=$r
elif test $r -gt $b
then
b=$r
fi
n='expr $n / 10'
done
echo "the second largest digit is : $b"
```

Output

```
Enter 2 numbers
30
50
largest number is 50
```


PROGRAM-15

AIM: Write a shell program to find the largest of 2 numbers using command line arguments

```
echo "Enter 2 numbers"
read n1
read n2
if test $n1 -gt $n2
then
echo "largest number is $n1"
else
echo "largest number is $n2"
fi
```

Output

```
Enter a number
3456
Sum of odd digits : 8
Sum of even digits : 10
```


PROGRAM-16

AIM: Write a shell program to find the sum of odd and even digits of an integer using command line

```
echo "Enter a number"
read n
os=0
es=0
while test $n -gt 0
do
r='expr $n % 10'
if test 'expr $r % 2' -eq 0
then
es='expr $es + $r'
else
os='expr $os + $r'
fi
n='expr $n / 10'
done
echo "sum of odd digits : $os"
echo "sum of even digits : $es"
```

Output

```
Enter a number
345
the reverse of the number 345 is 543
```


PROGRAM - 17

AIM : Write a shell program to print the reverse of a number.

```
echo "Enter a number"
read n
t=$n
s=0
while test $n -gt 0
do
r='expr $n % 10'
s='expr $r + $s \* 10
n='expr $n / 10'
done
echo "the reverse of the number $t is $s"
```

Output

```
Enter a number
5
120
```


PROGRAM-18

Aim: Write a shell program to find the factorial of a number

```
echo "Enter a number"
read num
fact=1
while test $num -gt 1
do
fact=$((fact \* num))
num=$((num - 1))
done
echo $fact
```

Output

```
How many number of terms to be generated
8
Fibonacci series up to 8 terms
0
1
1
2
3
5
8
13
```


PROGRAM-19

AIM: Write a shell program to find fibonacci series up to a given number using command line.

```
echo "How many number of terms to be generated"
read n
x=0
y=1
i=2
echo "fibonacci series up to $n terms"
echo "$x"
echo "$y"
while test $i -lt $n
do
i='expr $i + 1'
z='expr $x + $y'
echo "$z"
x=$y
y=$z
done
```

Output

```
Enter filename
Sample
Sample contains : 7  39  47  56  3
3  7  39  47  56
```


PROGRAM - 20

AIM : Write a shell program to sort numbers in a file in ascending order.

```
file=" "
echo -n "Enter filename"
read file
if [ ! -f $file ]
then
echo "$file is not a file"
exit 1
fi
sort -n $file
```

Output

```
example.sh linux file0
file contains : Linux is an open source os
                linux support multitasking
                Unix is not a free OS
Here are the strings with the pattern 'linux'
linux is an open source os
linux support multitasking
The word linux is found 2 times
```


PROGRAM-21

AIM: Write a shell program to find a specific pattern of string in a file using command line arguments.

```
PATTERN = $1

FILE = $2
if grep -i $PATTERN $FILE
then
echo "Here are the strings with the pattern '$PATTERN'"
echo "$(grep -i $PATTERN $FILE)
echo "The word $PATTERN is found $(grep -io
      $PATTERN $FILE | wc -l) times"
else
echo "Error : The pattern '$PATTERN' was not found
      in '$FILE'"
exit 0
fi
```

Output

```
example.sh 10 20 30
Sum is : 60
Average is : 20
```


Aim: Write a shell program to find the sum and average of numbers using command line arguments.

```
sum=0
for i in $*
do
sum='expr $sum + $i'
done
echo "Sum is : $sum"
avg='expr $sum / $#'
echo "Average is : $avg"
```

Output

```
Enter 2 numbers
10
20
Choose an operation:
1. Addition
2. Subtraction
3. Multiplication
4. Division
enter your choice
3
Mul = 200
If you want to continue, enter "y"
```


PROGRAM-23

AIM: Write a shell program to make a menu driven calculator using case

```
sum=0
i="y"
while test $i -eq "y"
do
echo "enter 2 numbers:"
read n1
read n2
echo "choose an operation:"
echo "1. Addition"
     "2. Subtraction"
     "3. Multiplication"
     "4. Division"
echo "enter your choice"
read op
case $op in
1)
  result='expr $n1 + $n2'
  echo "sum = $result"
  ;;
2)
  result='expr $n1 - $n2'
  echo "Sub = $result"
  ;;
```

```
3)
result='expr $n1 \* $n2'
echo "Mul = $result"
;;
4)
result='expr $n1 / $n2`
echo "Div = $result"
;;
*)
echo "Invalid choice"
;;
esac
echo "If you want to continue, enter "y""
read i
if test $i ! -eq "y"
then
exit
fi
done
```

Output

```
Enter the filename
sample
File sample exists
read = yes
write = yes
execute = yes
```


PROGRAM-24

AIM: Write a shell program to print the file access permission of a given file.

```
echo -n "Enter the filename"
read file
if [ -e $file ]
then
echo "File $file exists"
if [ -r $file ]
then
  echo "read = yes"
else
  echo "read = no"
fi
if [ -w $file ]
then
  echo "write = yes"
else
  echo "write = no"
fi
if [ -x $file ]
then
  echo "execute = yes"
else
  echo "execute = no"
fi
else
echo "$file does not exists"
fi
```

Output

```
Enter the size of array
5
Enter the elements:
23
12
34
10
56
Given array before sorting:
23  12  34  10  56
Sorted array:
10  12  23  34  56
```


PROGRAM-25

AIM: Write a shell program for sorting without using sort command.

```
echo "Enter the size of array"
read n
echo "Enter the elements:"
for((i=0;i<n;i++))
do
read a[i]
done
echo "Given array before sorting:"
echo ${a[@]}
for((i=0;i<n;i++))
do
for((j=1;j<n;j++))
do
if [ ${a[i]} -gt ${a[j]} ]
then
t=${a[i]}
a[i]=${a[j]}
a[j]=$t
fi
done
done
echo "Sorted array : ${a[@]}
```

Output

```
Enter the number
2
Enter its power
3
2^3 : 8
```


AIM: Write a shell script to find the power of a number

```
echo "Enter the number"
read num
echo "Enter its power"
read power
power_of_number =1
for((i=1;i<$power;i++))
do
power_of_number=$((power_of_number \* num))
done
echo "$number ^ $power : $power_of_number"
```

PROGRAM - 27

AIM: Write the basic linux commands to

a. Check the present working directory
   pwd [option]

b. List the contents of a directory using all options
   ls [option] [directory]
   -a, -d, -l

c. To create a file using cat command.
   cat > [name_of_new_file]

d. To display the contents of a file
   cat <filename>

e. To append the data to an existing file
   cat filename1 >> filename2

f. To add the data of two files to a third file.
   cat filename1 filename2 >> filename3

g. To create a directory
   mkdir [option] <directory_name>

h. To create multiple directories.
   mkdir dir1 dir2 dir3

i. To create nested directories
   mkdir -p parent_dir | child_dir | nested_child_dir

PROGRAM-28

AIM: Write the basic linux commands to

a. Copy a file

cp [options] <source_file> <destination_file>

b. Copy a directory

cp -R source_directory destination_directory

c. Change the directory

cd [options] [directory]

d. Move/Rename a file.

mv <filename> <destination_folder>

e. Copy a file to a different directory

cp filename /new_directory

f. Move a directory

mv <current_directory> /* <new_directory>

g. Delete a directory

rmdir [options] <directory_name>

h. Delete a file

rm [options] <filename>

i. Implement more and less commands

more <filename>

less <filename>

PROGRAM - 29

AIM: Write the linux commands to

a. clear the screen
   clear -x

b. To get the help of any command
   man [command]

c. To find the User/Users who have logged in
   who [options]

d. To get a binary calculator.
   bc

e. Change to different terminals.
   alt + cntrl + F_1 - F_7

f. To redirect the output of ls command to a file.
   ls command > ~/ filename

g. Change to different terminals
   alt + cntrl + F_1 - F_7

h. find the difference between files
   File comparison commands,
   cmp [options] <filename1> <filename2>
   diff [options] <filename2> <filename2>
   comm [options] <filename3> <filename2>

PROGRAM - 30

AIM: Write down advanced linux commands to

a. Find the processes run by different users and terminals
   ps -u [username]

b. To redirect the output of ls command to a file
   ls > filename

c. To terminate a process using different options.
   Kill [-signal number] <PID>

d. To start a background process
   To start a background process add ampresand (&) at the end of the command.

e. To issue a nohup command
   nohup <commands> &

f. Find the processes using different options.
   ps [options]
   - A, -e // view all running processes
   - T // select all processes on this terminal
   - a // view processes except ones associated with terminal and session leaders
   - u // list processes running on specified user
   - t // list processes running on specified terminal
   - x // list all system processes