* **Variables in Linux**

Sometimes to process our data/information, it must be kept in computers RAM memory. RAM memory is divided into small locations, and each location had unique number called memory location/address, which is used to hold our data. Programmer can give a unique name to this memory location/address called memory variable or variable (Its a named storage location that may take different values, but only one at a time).

In Linux, there are two types of variable

**1) System variables -** Created and maintained by Linux itself. This type of variable defined in CAPITAL LETTERS.

**2) User defined variables (UDV) -** Created and maintained by user. This type of variable defined in lower LETTERS.

**Rules for Naming variable name (Both UDV and System Variable:**

* Variable name must *begin with Alphanumeric character or underscore character (\_)*, followed by one or more Alphanumeric character.

**For e.g.,** Valid shell variable are as follows : **HOME, SYSTEM\_VERSION, vech, no**

* *Don't put spaces on either side of the equal sign* when assigning value to variable.

**For e.g.** In following variable declaration there will be no error

**$ no=10**

But here there will be problem for following

$ no =10 **//ERROR**

$ no= 10 **//ERROR**

$ no = 10 **//ERROR**

* *Variables are case-sensitive*, just like filename in Linux. **For e.g**.

$ no=10

$ No=11

$ NO=20

$ nO=2

Above all are different variable name, so to print value 20 we have to use $ echo $NO and Not any of the following:

echo $no **# will print 10** but not 20

echo $No **# will print 11** but not 20

echo $nO **# will print 2** but not 20

* You can define NULL variable as follows (NULL variable is variable which has no value at the time of definition) **For e.g**.

$ vech=

$ vech=""

Try to print it's value **$ echo $vech** , Here nothing will be shown because variable has no value i.e. NULL variable.

* Do not use ?,\* etc, to name your variable names.

**25 Oct, Monday**

* **Shell Programming:**

**UNIX Shells:**

UNIX shell is a command line like DOS in Windows. It’s a user interface for UNIX operating system. Mainly shells are used for inputting user OS commands. It is called “Shell” because it hides all the information behind the shell interface.

* **Types of Shells:**

**1)Bourne Shell (sh):** It’s the default UNIX shell. Most of the scripts to configure OS is written using this shell. It was developed by Stephen Bourne.

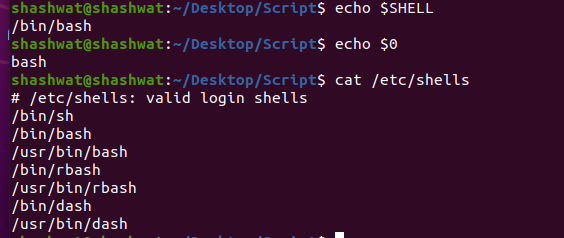
**2)C Shell (csh):** It is called C shell because the syntax used here is similar to c language. It adds many features compare to bourne shell. This shell is not widely used now. It was developed by Bill Joy.

**3)Korn Shell (ksh):** This shell is backward compatible with bourne shell & inherits many features of C shell. This was developed by David Korn.

**4)Bash Shell (bash):** It stands for Bourne again Shell i.e. It is superset of bourne shell. It was built by Stephen Bourne.

* **Finding Shell:**

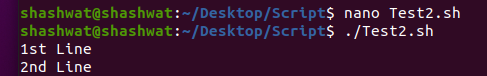
1. **echo $SHELL / echo $0** 🡺 To find which shell we are working at
2. **cat /etc/shells** 🡺 To check Shells available in UNIX



* **Command Separator (Semicolon):**

We can write two or more commands in single line. To do this we need to use semicolon (;)

Eg: echo "1st Line " ; echo "2nd Line" **;**



* **Eg:**

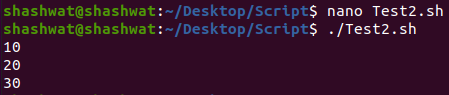
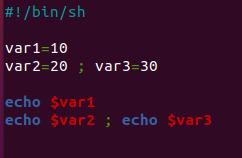
var1=10 OUTPUT:

var2=20 ; var3=30 10

20

echo $var1 30

echo $var2 ; echo $var3



* **Writing and executing in shell**

**Write Script:**

Create a shell script using a text editor (Nano). Save the script file as **.sh**.

**Syntax:** nano filename.sh

**Eg:** nano Test.sh

**Setting up Execute Permission:**

Before executing the script, you need to set permission to read, write and execute. To set file permission use command**,**

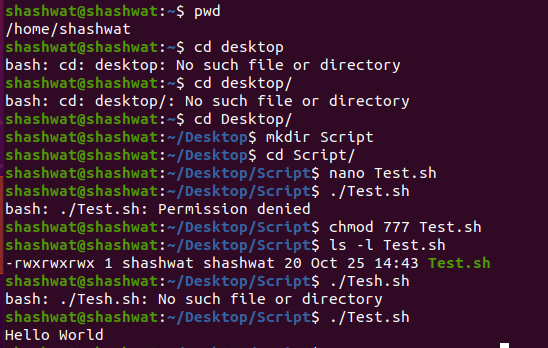
**Syntax:** chmod 777 <script\_name>

**Eg:** chmod 777 Test.sh

**Nano Editor:**

* nano <filename.sh> 🡪 open nano editor
* Ctrl+X 🡪 Exit from editor

Select yes to save changes and Press Enter



pwd

cd Desktop/ 🡺 go to Desktop

mkdir Scripts 🡺 make a directory with name “Scripts”

cd Scripts/ 🡺 change directory to “Scripts”

nano Test.sh 🡺 open nano editor and create a shell file with name “Test.sh”

./Test.sh 🡺 Run Test.sh

chmod 777 Test.sh 🡺 set file permission to read, write and execute

ls -l Tesh.sh 🡺

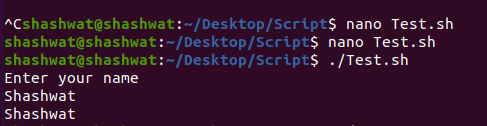
* **Read Name from user and display it**

#!/bin/sh

echo "Enter your name"

read name

echo $name



* **Enter the arguments and print it**

#!/bin/sh

#echo "Enter no of arguments: "

#echo $#

echo $0

echo $1

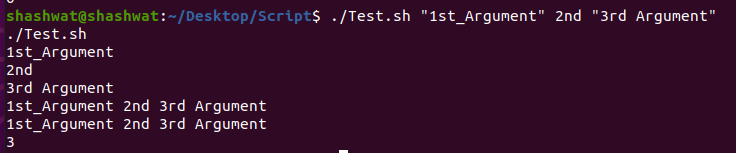
echo $2

echo $3

echo $\*

echo $@

echo $#



**#** This is comment 🡺 Comment

echo $0 🡺 displays the name of file

echo $1 🡺 displays 1st argument

echo $2 🡺 displays 2nd argument

echo $ 🡺 displays 3rd argument

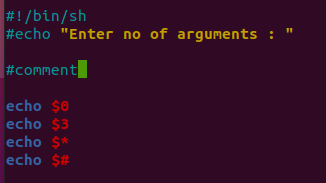
echo $\* 🡺 displays all the arguments entered by user in one line

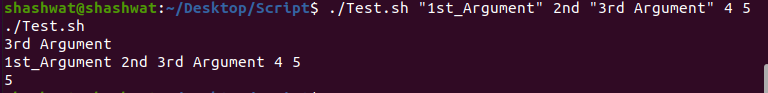
echo $@ 🡺 same as $\*

echo $# 🡺 displays the no of arguments (int value)

**NOTE:**

* If we enter more arguments then the rest of the arguments are neglected
* If we enter 4-5 arguments and print echo $3 then only 3rd argument will be printed only

****

****

***# Arithmetic operations***

* **Increment Value of a**

#!/bin/sh

a=3

a=`expr $a + 5`

echo "Value is :$a"



* **Multiply Value by a number**

#!/bin/sh

a=10

a=`expr $a \\* 3`

echo $a



* **Add two numbers**

#!/bin/sh

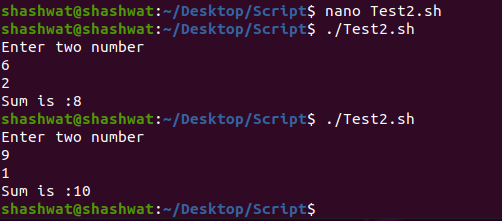
echo "Enter two number"

read a

read b

sum=`expr $a + $b`

echo "Sum is :$sum"



***#Relational Operator***

#-eq ==> Equal to

#-ne ==> Not equal to

#-gt ==> Greater than (>)

#-lt ==> Less than (<)

#-ge ==> Greater than equal to (>=)

#-le ==> Less than equal to (<=)

* **Less than equal to**

#!/bin/sh

b=10

c=20

if [ $b -le $c ]

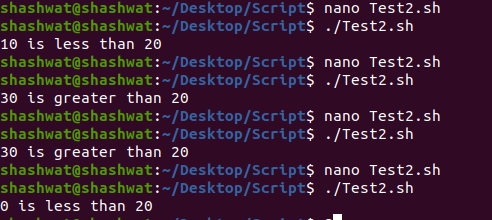
then

echo "$b is less than $c"

else

echo "$b is greater than $c"

fi



* **Enter a no and check for ODD/EVEN :**

#!/bin/sh

echo "Enter the number:"

read n

a=`expr $n %2`

if [ $a -eq 0 ]

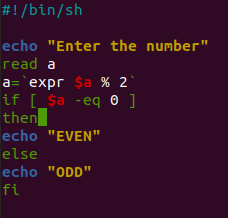
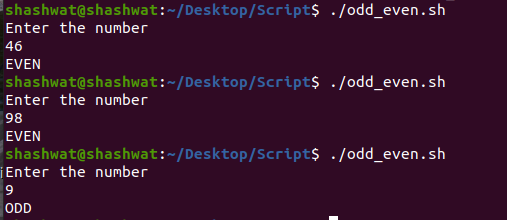
then

echo “EVEN”

else

echo “ODD”

fi

***#Logical operators:* !, -o, -a**

***#File operators****:* -b, -c, -d, -f, -g ......

***#String Operators:*** =, !=, -z, -n, str

* **Check for String Equality ( = , != ):**

str1="Sahu"

str2="Kumar"

echo "String 1 : $str1"

echo "String 2 : $str2"

if [ $str1 = $str2 ]

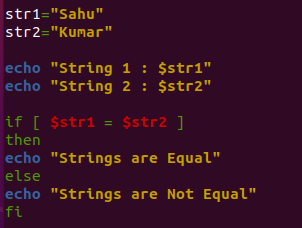
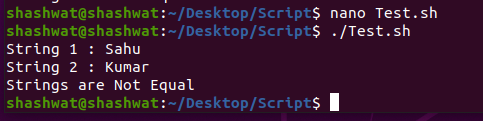
then

echo "Strings are Equal"

else

echo "Strings are Not Equal"

fi

* **Check for Empty string:**

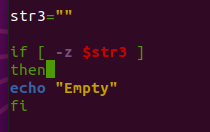
str3=""

if [ -z $str1 ]

then

echo "Empty"

fi

***#Looping Statement***

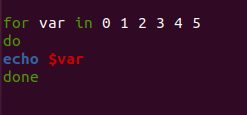
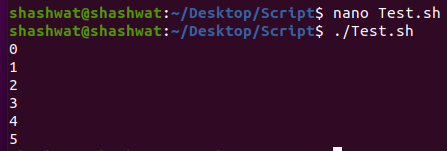
* **FOR LOOP :**
* **Simple for loop print**

for var in 0 1 2 3 4 5

do

echo $var

done

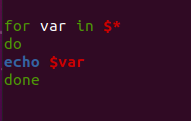
* **Take Argument input from user and print using for loop**

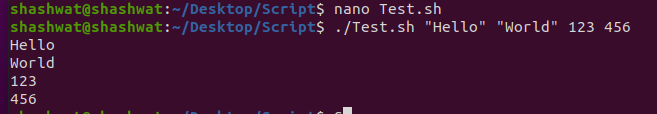
for var in $\*

do

echo $var

done





* **For loop with two Parameters:**

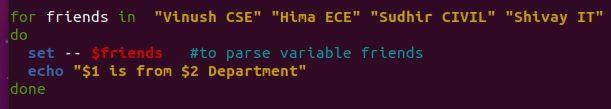
for friends in "Vinush CSE" "Hima ECE" "Sudhir CIVIL" "Shivay IT"

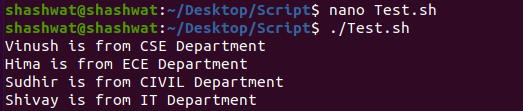
do

set -- $friends **#to parse variable friends**

echo "$1 is from $2 Department"

done





* **Prime No or Not :**

echo "Enter a number "

read num

for((i=2; i<=$num/2; i++))

do

if [[ $((n%i)) -eq 0 ]]

then

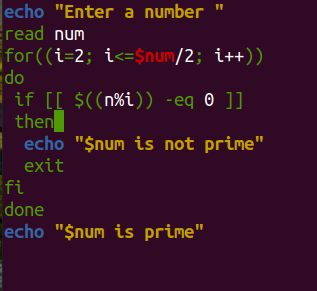
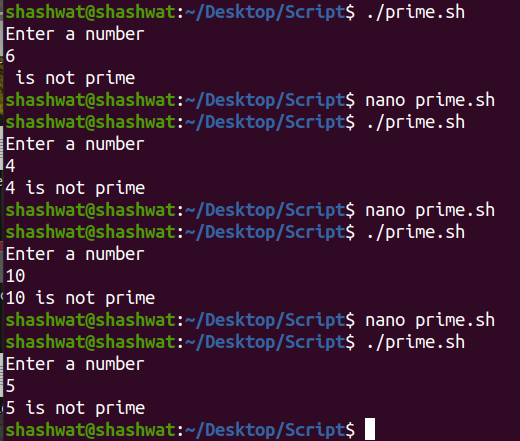
echo "$num is not prime"

exit

fi

done

echo "$num is prime"

* **WHILE LOOP**
* **Simple while loop print**

a=1

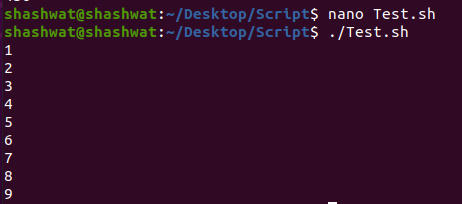
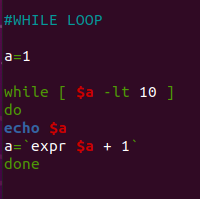
while [ $a -lt 10 ]

do

echo $a

a=`expr $a + 1`

done



* **UNTIL LOOP**

a=12

until [ !$a -lt 10 ]

do

echo $a

a=`expr $a +1`

done

**8 NOV, Monday**

* **Multiple of 5 or not**

#!/bin/sh

echo "Enter a number "

read a

n=`expr $a % 5`

if [ $n -eq 0 ]

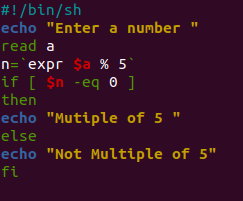
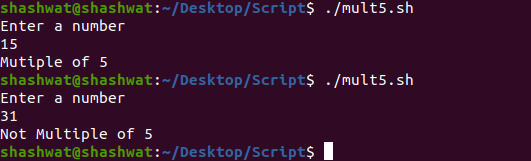
then

echo "Multiple of 5 "

else

echo "Not Multiple of 5"

fi

* **Table of 3**

#!/bin/sh

echo "Enter the number:"

read num

i=1

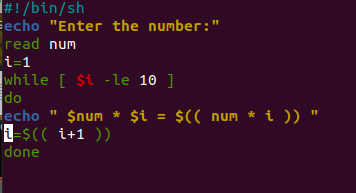
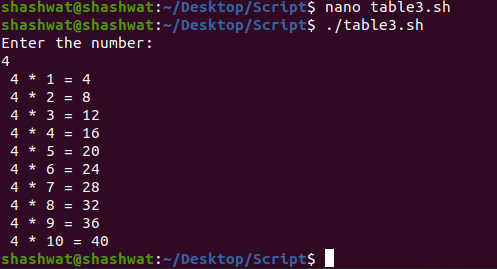
while [ $i -le 10 ]

do

echo " $num \* $i = $(( num \* i )) "

i=$(( i+1 ))

done

* **Count no of digit in an integer**

echo "Enter a number"

read num

count=0

temp=$num

while [ $num -gt 0 ]

do

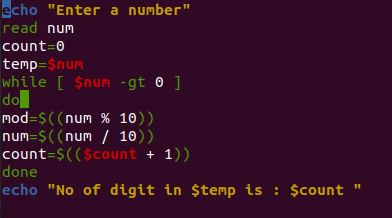
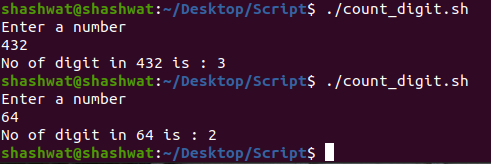
mod=$((num % 10))

num=$((num / 10))

count=$(($count + 1))

done

echo "No of digit in $temp is : $count "

* **Check if number is positive or negative**

echo " Enter a number : "

read num

if [ $num -gt 0 ]

then

echo "Positive Number"

elif [ $num -lt 0 ]

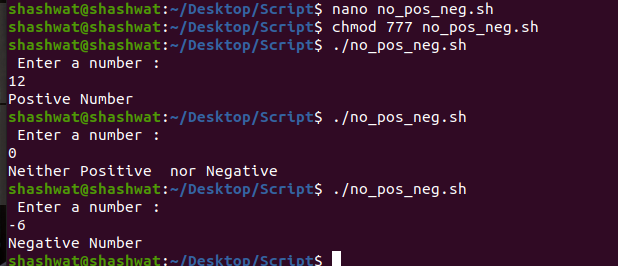
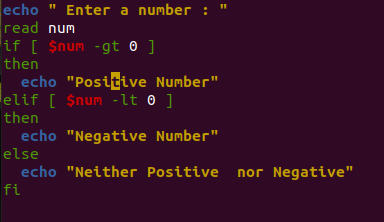
then

echo "Negative Number"

else

echo "Neither Positive nor Negative"

fi



* **Find Sum of digits of a number**

echo "Enter a number"

read num

sum=0

while [ $num -gt 0 ]

do

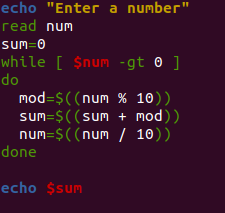
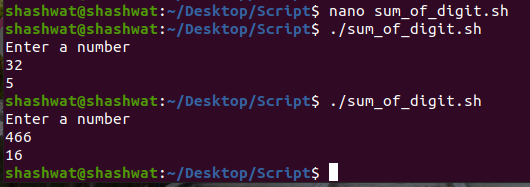
mod=$((num % 10))

sum=$((sum + mod))

num=$((num / 10))

done

echo $sum

* **Find greater of 2 numbers**

echo "Enter first num : "

read a

echo "Enter second num : "

read b

if [ $a -gt $b ]

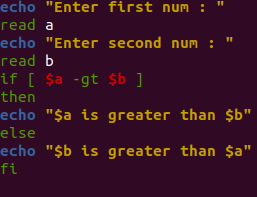
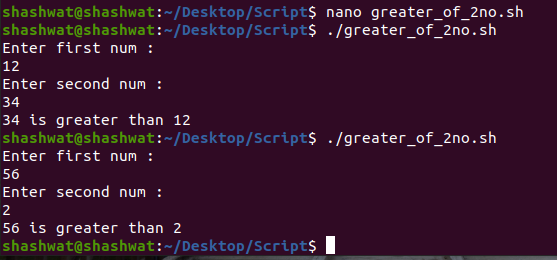
then

echo "$a is greater than $b"

else

echo "$b is greater than $a"

fi

* **Find power of a number**

echo "Enter the number : "

read num

echo "Enter power:"

read pow

a=1

c=1

while [ $a -le $pow ]

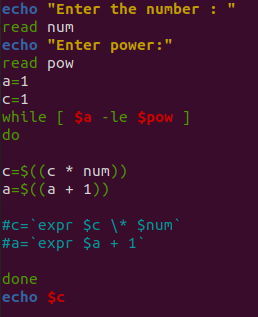
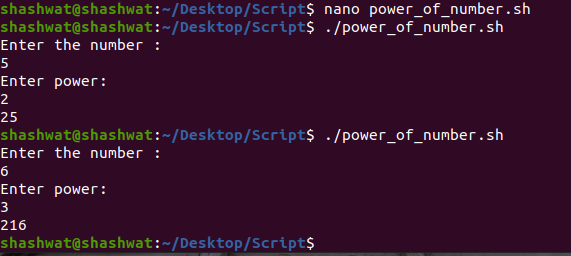
do

c=`expr $c \\* $num` // c=$((c \* num))

a=`expr $a + 1` // a=$((a + 1))

done

echo $c

* **Check if the character is vowel or not**

**METHOD 🡪 1**

echo "Enter a character :"

read ch

case $ch in

a)echo "Vowel";;

e)echo "Vowel";;

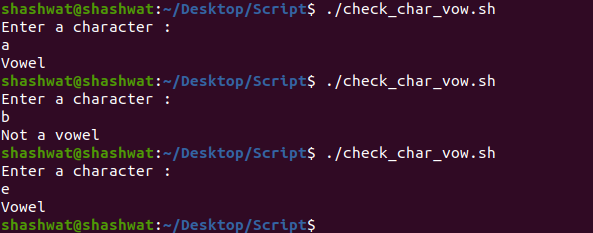
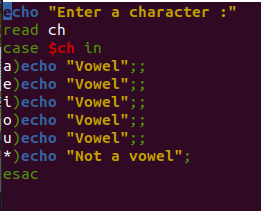
i)echo "Vowel";;

o)echo "Vowel";;

u)echo "Vowel";;

\*)echo "Not a vowel";

esac



**METHOD 🡪 2**

#!/bin/bash

echo "Enter a character: "

read c

if [[ $c == [AEIOUaeiou] ]]

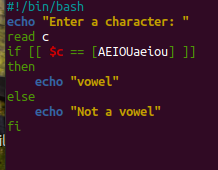
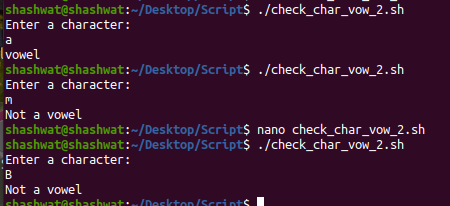
then

echo "vowel"

else

echo "Not a vowel"

fi

* **Print first even number in a list**

echo "Enter staring number"

read n1

echo "Enter ending number"

read n2

while [ $n1 -le $n2 ]

do

if [ $(($n1 % 2)) -eq 0 ]

then

echo "$n1 is first even no"

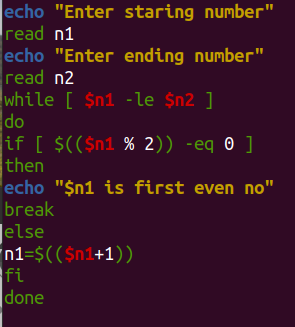
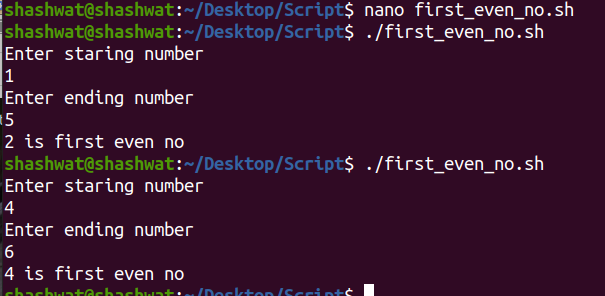
break

else

n1=$(($n1+1))

fi

done

* **Print second vowel in a string**

echo "Enter String : "

read str

x=0

echo "$str" | grep -o . | while read letter

do

if [[ $letter == [AEIOUaeiou] ]]

then

x=$((x+1))

fi

if [ $x -eq 2 ]

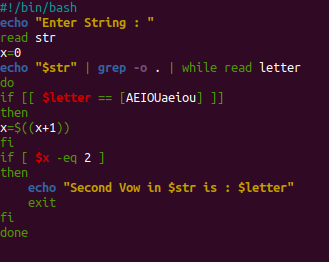
then

echo "Second Vow in $str is : $letter"

exit

fi

done



* **Print Sum of digit in a number**

echo "Enter a number"

read num

count=0

temp=$num

sum=0

while [ $num -gt 0 ]

do

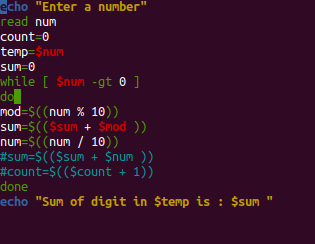
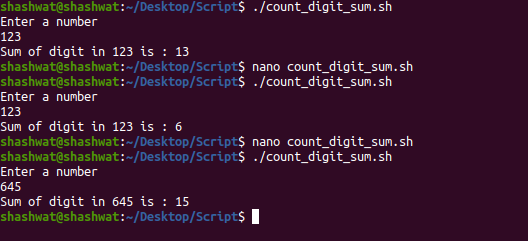
mod=$((num % 10))

sum=$(($sum + $mod ))

num=$((num / 10))

done

echo "Sum of digit in $temp is : $sum "

* **Greater of 3 numbers**

echo "Enter Num1"

read num1

echo "Enter Num2"

read num2

echo "Enter Num3"

read num3

if [ $num1 -gt $num2 ] && [ $num1 -gt $num3 ]

then

echo $num1

elif [ $num2 -gt $num1 ] && [ $num2 -gt $num3 ]

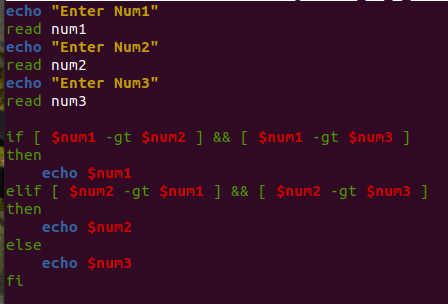
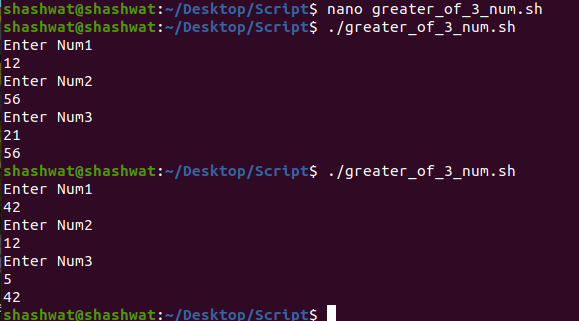
then

echo $num2

else

echo $num3

fi

* **Print All Vowels in a String**

read str

length=${#str}

for((i=0;i<$length;i++))

do

c=${str:i:1}

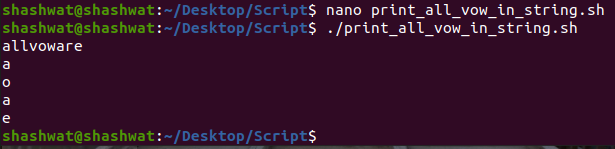
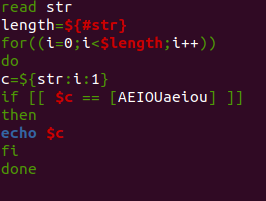
if [[ $c == [AEIOUaeiou] ]]

then

echo $c

fi

done



* **Print Second Even no of a given List**

echo "Enter the numbers in the list"

read list

count=0

for n in $list

do

if [ $(( $n%2 )) -eq 0 ]

then

if [ $count -eq 1 ]

then

echo "Second even no. of the list is $n"

count=2

break

fi

count=1

fi

done

if [ $count -eq 1 ]

then

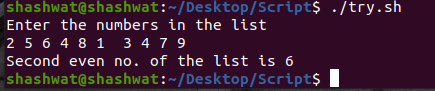
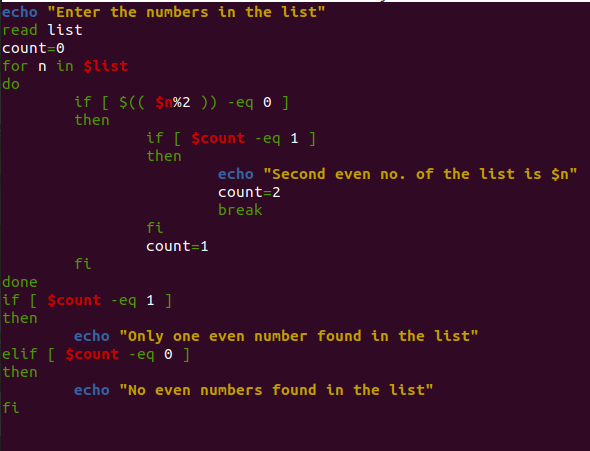
echo "Only one even number found in the list"

elif [ $count -eq 0 ]

then

echo "No even numbers found in the list"

fi



* **Calculate Simple Interest**

echo "Enter the principle value: "

read p

echo "Enter the rate of interest:"

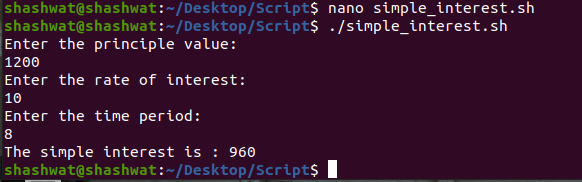
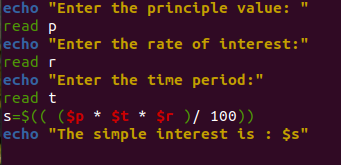
read r

echo "Enter the time period:"

read t

s=$(( ($p \* $t \* $r )/ 100))

echo "The simple interest is : $s"



* **Check for Palindrome No**

echo "Enter a number"

read num

sum=0

temp=$num

while [ $num -gt 0 ]

do

mod=$((num % 10))

sum=$(( ($sum\*10) + $mod))

num=$((num / 10))

done

if [ $sum -eq $temp ]

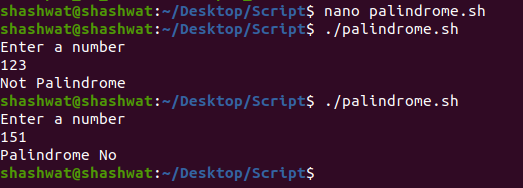
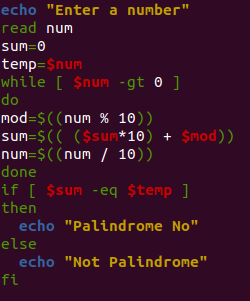
then

echo "Palindrome No"

else

echo "Not Palindrome"

fi



* **Check for Multiple of 9**

#!/bin/sh

echo "Enter a number "

read a

n=`expr $a % 9`

if [ $n -eq 0 ]

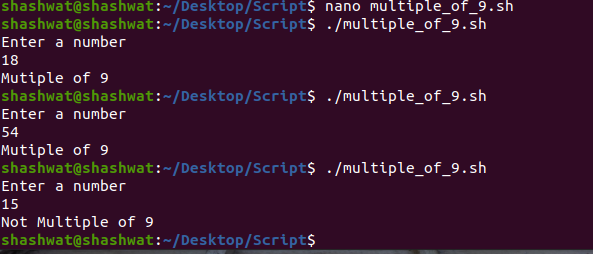
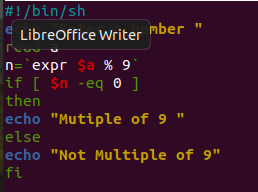
then

echo "Mutiple of 9 "

else

echo "Not Multiple of 9"

fi



* **Check for Compound Interest**

#!/bin/sh

* **Fibonacci Series**

echo "Enter number : "

read n

a=-1

b=1

while [ $n -gt 0 ]

do

c=$(($a+$b))

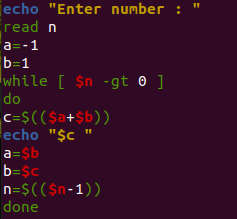
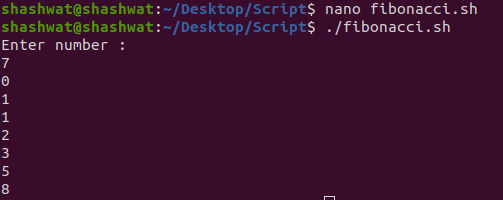
echo "$c "

a=$b

b=$c

n=$(($n-1))

done

* **Write 5 cmnd line argument and print it**

#!/bin/sh

echo "First Argument : $1"

echo "Second Argument : $2"

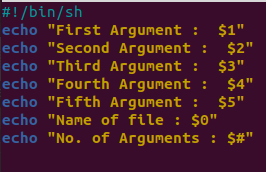
echo "Third Argument : $3"

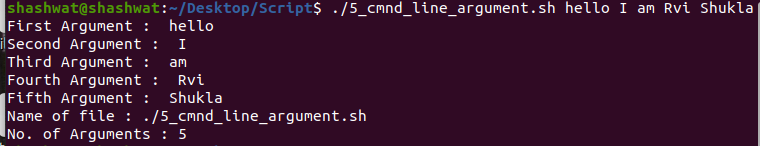
echo "Fourth Argument : $4"

echo "Fifth Argument : $5"

echo "Name of file : $0"

echo "No. of Arguments : $#"

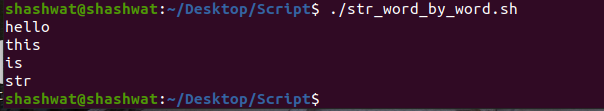




* **Print String Word by Word**

str="hello this is str "

echo $str | xargs -n1

* **First 2 ODD Digit in an integer**

echo "Input a number"

read n

a=$n

c=0

d=0

while [ $a -gt 0 ]

do

r=$(( $a%10 ))

if [ $(( $r%2 )) -eq 1 ]

then

c=$d

d=$r

fi

a=$(( $a/10 ))

done

if [ $d -gt 0 ] && [ $c -gt 0 ]

then

echo "$d $c"

elif [ $d -gt 0 ]

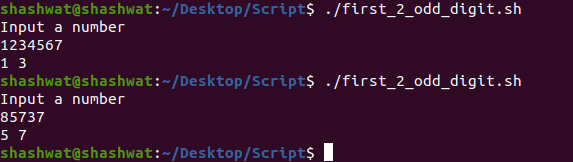
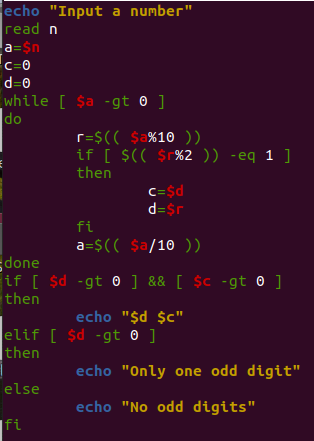
then

echo "Only one odd digit"

else

echo "No odd digits"

fi



* **Factorial of a No**

echo "Enter a number : "

read n

res=1

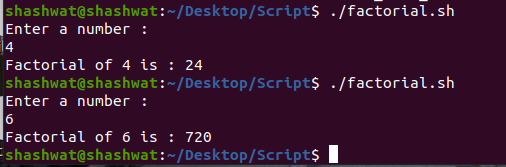
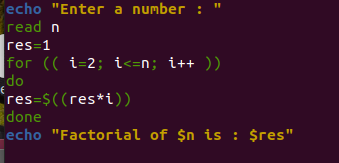
for (( i=2; i<=n; i++ ))

do

res=$((res\*i))

done

echo "Factorial of $n is : $res"



* **Print Reverse of a String**

read -p "Enter a string : " str

length=${#str}

i=$((length-1))

while [ $i -ge 0 ]

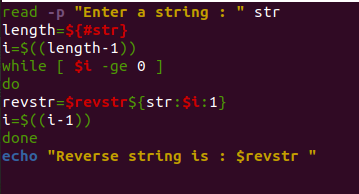
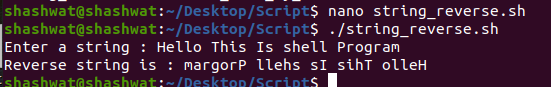
do

revstr=$revstr${str:$i:1}

i=$((i-1))

done

echo "Reverse string is : $revstr "

* **Print Reverse of an Integer**

echo "Enter a number"

read num

rev=0

while [ $num -gt 0 ]

do

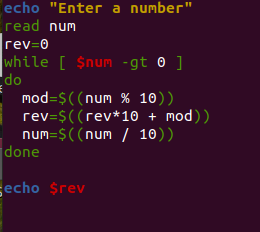
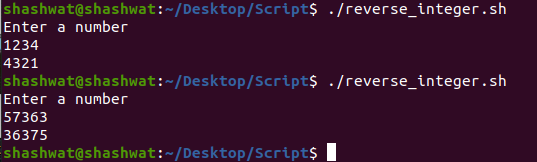
mod=$((num % 10))

rev=$((rev\*10 + mod))

num=$((num / 10))

done

echo $rev

* **Enter 2 no and check if they are equal or not**

echo "Enter a"

read a

echo "Enter b"

read b

if [ $a -eq $b ]

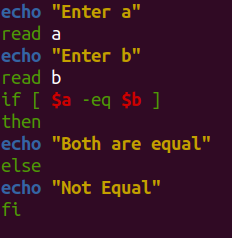
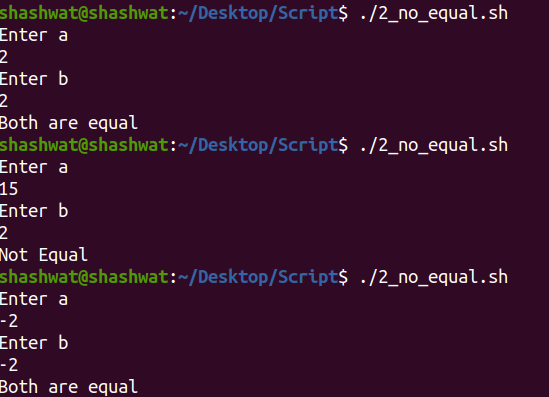
then

echo "Both are equal"

else

echo "Not Equal"

fi

* **Enter 2 String and check if they are equal or not**

echo "Enter a"

read a

echo "Enter b"

read b

if [ $a = $b ]

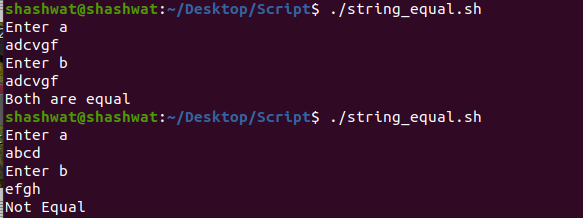
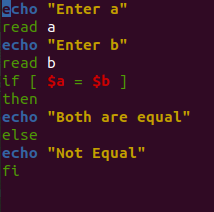
then

echo "Both are equal"

else

echo "Not Equal"

fi



* **Take a String and print its Substring**

echo "Enter String"

read str

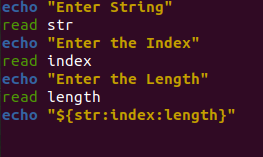
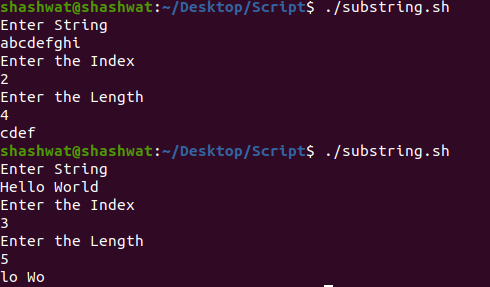
echo "Enter the Index"

read index

echo "Enter the Length"

read length

echo "${str:index:length}"

* **Enter number using loop and the program Exits when user Enters an Even Number**

while ((1))

do

echo "Enter a number : "

read a

if [ $(($a % 2)) -eq 0 ]

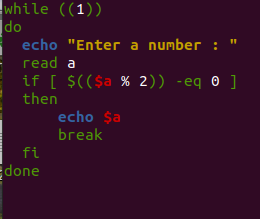
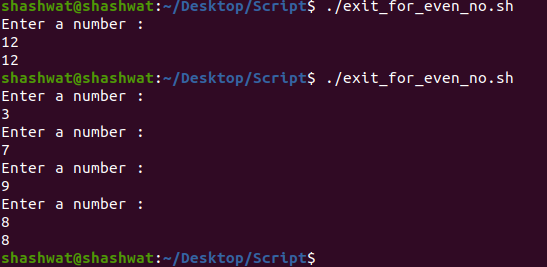
then

echo $a

break

fi

done

* **Count the no of Vowels in a given String**

#Count of vowel

#...........................

echo "Enter string "

read s

n=${#s}

cnt=0

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

do

ch=${s:i:1}

if [ $ch == 'a' ] || [ $ch == 'e' ] || [ $ch == 'i' ] || [ $ch == 'o' ] || [ $ch == 'u' ]

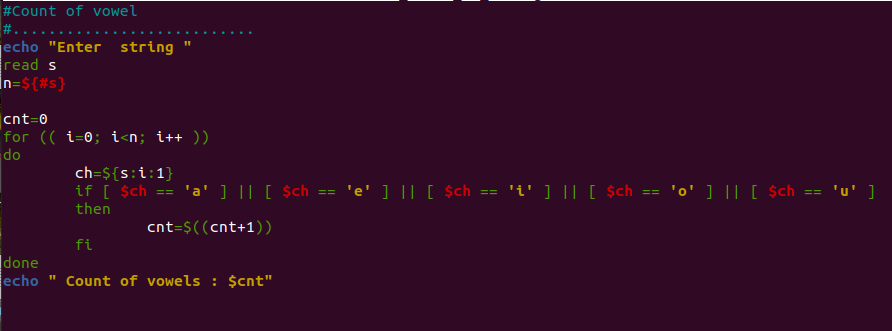
then

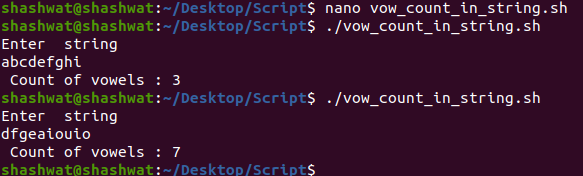
cnt=$((cnt+1))

fi

done

echo " Count of vowels : $cnt"





* **Enter a Number and Print 1 for Positive No, -1 for Negative No and 0 for Zero**

read a

if [ $a -gt 0 ]

then

echo "1"

elif [ $a -lt 0 ]

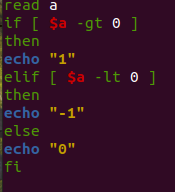
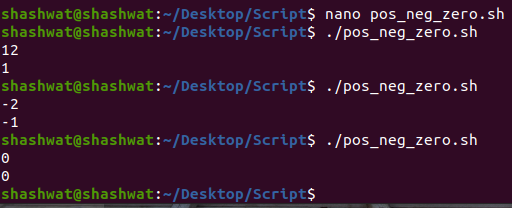
then

echo "-1"

else

echo "0"

fi

* **Program for vote eligibility**

echo "Enter Age"

read age

if [ $age -ge 18 ]

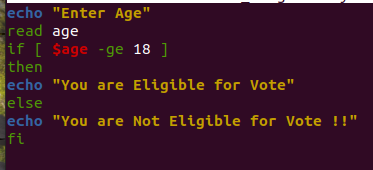
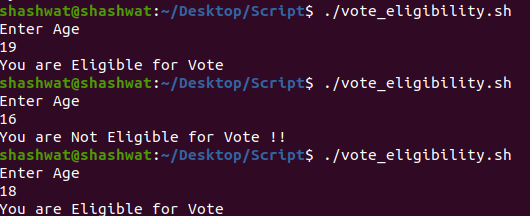
then

echo "You are Eligible for Vote"

else

echo "You are Not Eligible for Vote !!"

fi

* **Print Last Vowel in a String**

read -p "Enter a string : " str

length=${#str}

for((i=0;i<$length;i++))

do

c=${str:i:1}

if [[ $c == [AEIOUaeiou] ]]

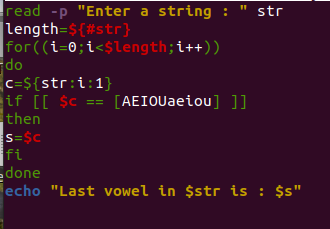
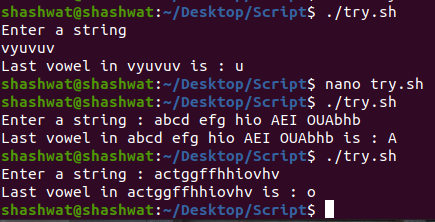
then

s=$c

fi

done

echo "Last vowel in $str is : $s"

* **Program for Coordinate Axis and tell the Quadrant**

echo "Enter X-Cordinate"

read x

echo "Enter Y-Cordinate"

read y

if [ $x -gt 0 ] && [ $y -gt 0 ]

then

echo "First Quadrant"

elif [ $x -lt 0 ] && [ $y -gt 0 ]

then

echo "Second Quadrant"

elif [ $x -lt 0 ] && [ $y -lt 0 ]

then

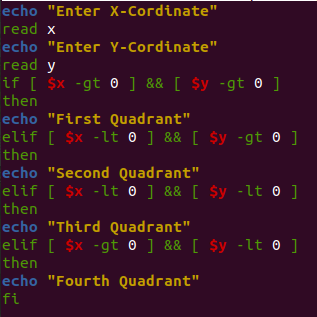
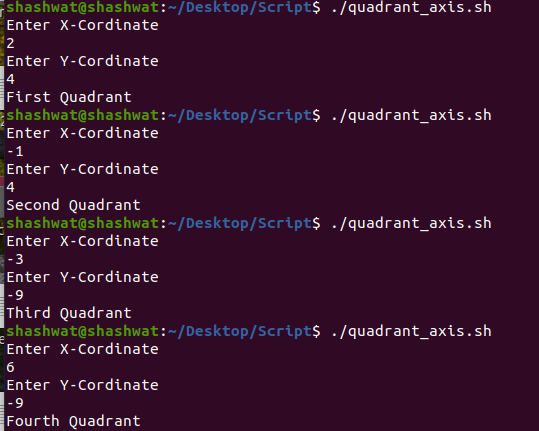
echo "Third Quadrant"

elif [ $x -gt 0 ] && [ $y -lt 0 ]

then

echo "Fourth Quadrant"

fi

* **Print 2nd Last Even Number of a given Integer**

echo "Enter a number"

read num

a=$num

count=0

while [ $a -gt 0 ]

do

c=$(( $a%10 ))

if [ $(( $c%2 )) -eq 0 ]

then

count=$(( count+1 ))

if [ $count -eq 2 ]

then

echo "Second last even digit of $num=$c"

break

fi

fi

a=$(( $a/10 ))

done

if [ $count -eq 0 ]

then

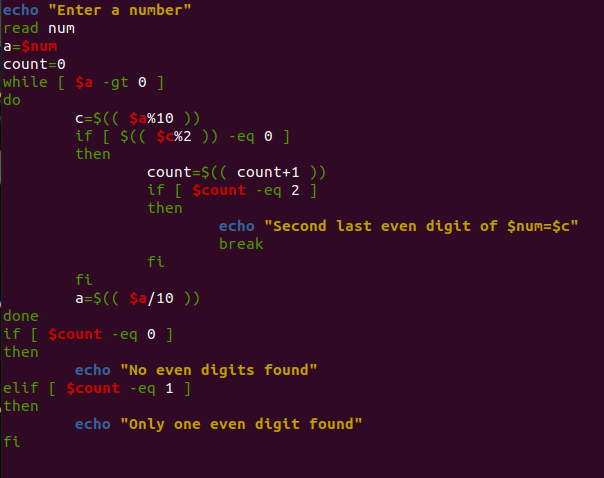
echo "No even digits found"

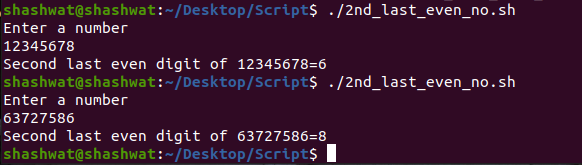
elif [ $count -eq 1 ]

then

echo "Only one even digit found"

fi





* **Print nth Fibonacci Sequence**
* **Multiple of 3and 5 or not**

#!/bin/sh

echo "Enter a number "

read a

m=$a

n=$(($a % 3))

m=$(($m % 5))

if [ $n -eq 0 ] && [ $m -eq 0 ]

then

echo "Mutiple of 3 and 5 "

elif [ $n -eq 0 ]

then

echo "Multiple of 3 only"

elif [ $m -eq 0 ]

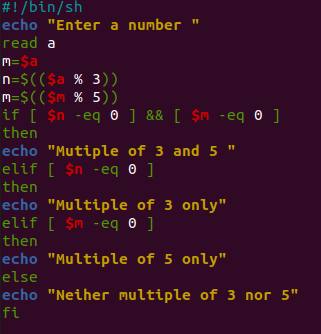
then

echo "Multiple of 5 only"

else

echo "Neiher multiple of 3 nor 5"

fi





* **Print Success Message on Odd No else say try again(say for 3-4 times)**

echo "Enter an odd number"

ans=0

for ((i=1;i<=3;i++))

do

echo "Chance $i"

read n

if [ $(( n%2 )) -eq 1 ]

then

ans=1

echo "Congratulations odd number found $n"

break

else

echo "Try again"

fi

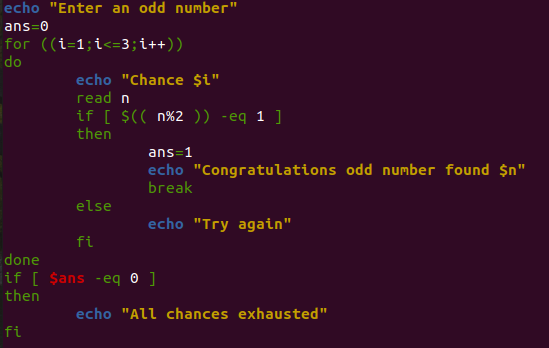
done

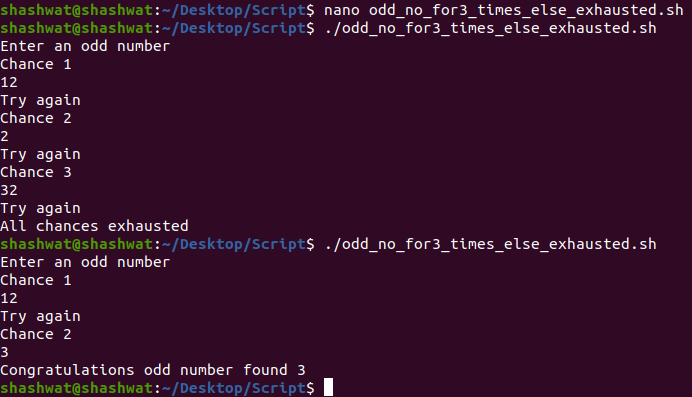
if [ $ans -eq 0 ]

then

echo "All chances exhausted"

fi





* **Sum of Alternate Odd no:**

echo "Enter the value of n"

read n

c=1

sum=0

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

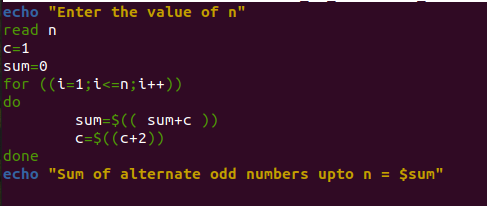
do

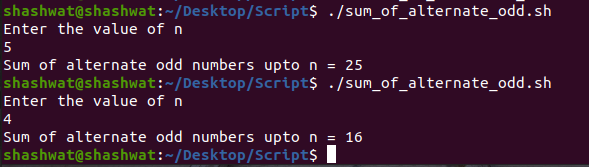
sum=$(( sum+c ))

c=$((c+2))

done

echo "Sum of alternate odd numbers upto n = $sum"





* **Sum of Alternate Odd no from 1 to N:**

echo "Enter the value of n"

read n

sum=0

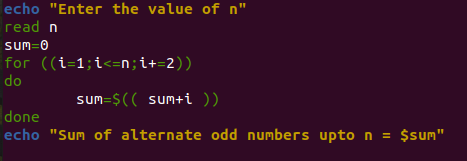
for ((i=1;i<=n;i+=2))

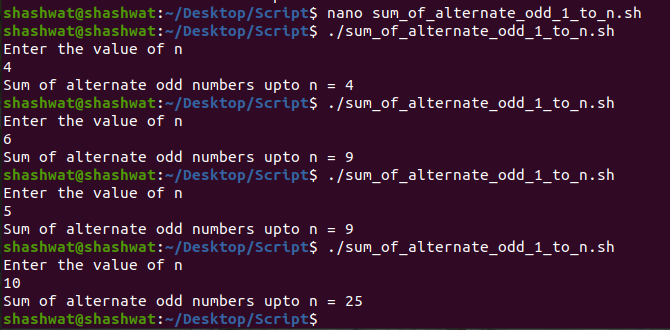
do

sum=$(( sum+i ))

done

echo "Sum of alternate odd numbers upto n = $sum"





* **Sum of Alternate Odd no from given range (A to B):**

echo "Enter start value: "

read n1

if [ $(($n1 % 2)) -eq 0 ]; then n1=$((n1+1))

fi

echo "Enter end value: "

read n2

sum=0

while [ $n1 -le $n2 ]

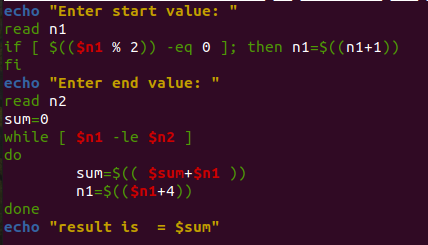
do

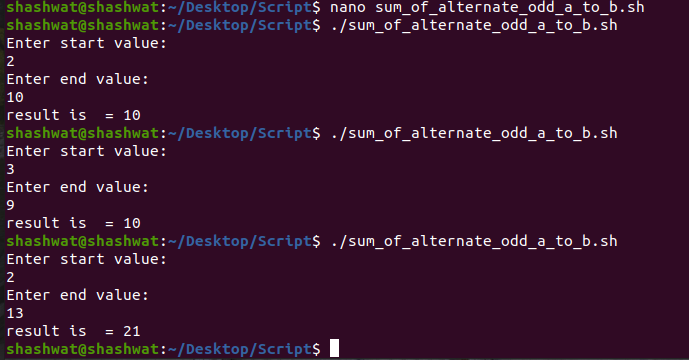
sum=$(( $sum+$n1 ))

n1=$(($n1+4))

done

echo "result is = $sum"





* **Count the number of Consonants in a given String:**
* **Check if a Year is Leap Year or Not:**

#!/bin/sh

echo "Enter the year"

read year

x=$(( $year % 400))

y=$(( $year % 100))

z=$(( $year % 4))

if [ $x -eq 0 ] || [ $y -ne 0 ] && [ $z -eq 0 ]

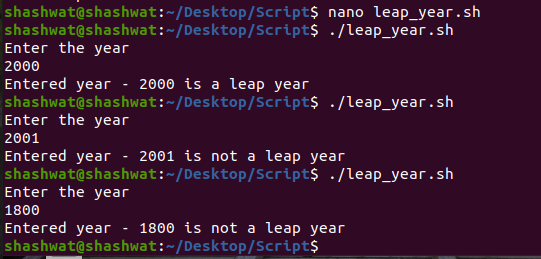
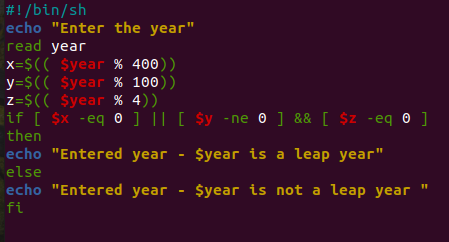
then

echo "Entered year - $year is a leap year"

else

echo "Entered year - $year is not a leap year "

fi



* **Enter the next immediate Leap Year:**

#!/bin/sh

echo "Enter the year"

read year

a=$(( $year % 4 ))

b=$(( $year % 100 ))

c=$(( $year % 400 ))

e=$year

if [ $a -eq 0 -a $b -ne 0 -o $c -eq 0 ]

then

echo " Entered year - $year is a leap year"

d=$(( $e+4 ))

echo "Next Leap Year is : $d"

else

echo "Entered year - $year is not a leap year "

while [ $(($year %4)) -ne 0 ]

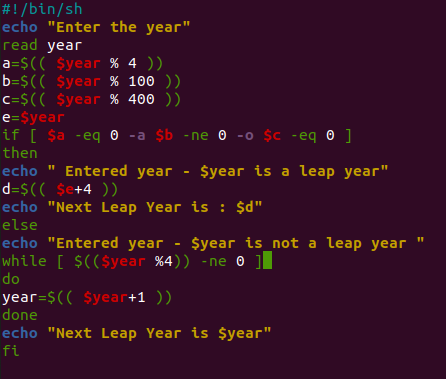
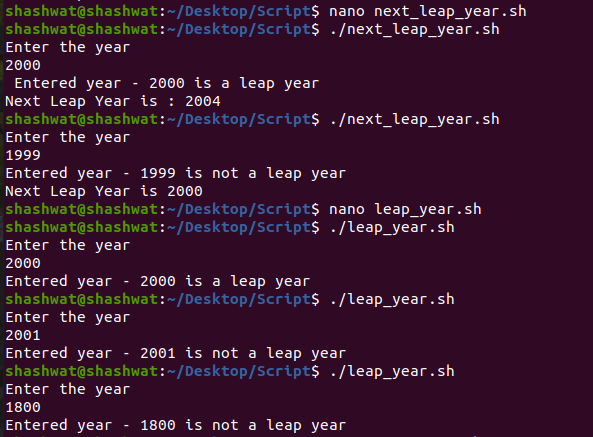
do

year=$(( $year+1 ))

done

echo "Next Leap Year is $year"

fi

* **Enter 3 Sides of triangle and check if Triangle is possible or not:**
* **Program for X power of Y:**
* **H**
* **J**