**Aim**: To create shell scripts for the following questions.

1. **To find Largest of Three Numbers.**

#!/bin/bash

# Script to find the largest of three numbers

# Check if three arguments are provided

if [ $# -ne 3 ]; then

echo "Usage: $0 number1 number2 number3"

echo "Please provide exactly three numbers as arguments."

exit 1

fi

# Store the three numbers in variables

num1=$1

num2=$2

num3=$3

# Find the largest number

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

echo "$num1 is the largest number"

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

echo "$num2 is the largest number"

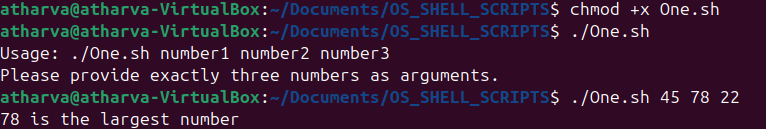
elif [ $num3 -gt $num1 ] && [ $num3 -gt $num2 ]; then

echo "$num3 is the largest number"

else

echo "All numbers are equal"

fi



1. **To find a year is leap year or not.**

#!/bin/bash

# Script to check if a year is a leap year

# Check if one argument is provided

if [ $# -ne 1 ]; then

echo "Usage: $0 year"

echo "Please provide a year as an argument."

exit 1

fi

year=$1

# Check if year is a leap year

# A year is a leap year if:

# 1. It is divisible by 400, OR

# 2. It is divisible by 4 but not by 100

if ([ $((year % 400)) -eq 0 ] || [ $((year % 4)) -eq 0 -a $((year % 100)) -ne 0 ]); then

echo "$year is a leap year"

else

echo "$year is not a leap year"

fi



1. **To input angles of a triangle and find out whether it is valid triangle or not.**

#!/bin/bash

# Script to check if a triangle is valid based on angles

# Get three angles from user

echo "Enter three angles of the triangle:"

read -p "Angle 1: " angle1

read -p "Angle 2: " angle2

read -p "Angle 3: " angle3

# Calculate the sum of angles

sum=$((angle1 + angle2 + angle3))

# Check if the sum is 180 degrees

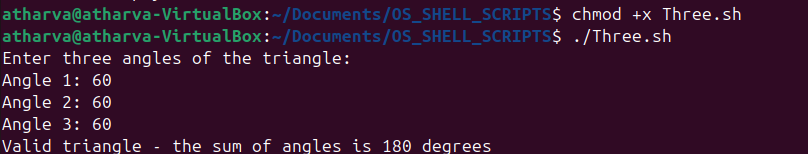
if [ $sum -eq 180 ]; then

echo "Valid triangle - the sum of angles is 180 degrees"

else

echo "Invalid triangle - the sum of angles is not 180 degrees"

fi



1. **To check whether a character is alphabet, digit or special character.**

#!/bin/bash

# Script to check if a character is alphabet, digit, or special character

# Get a character from user

read -p "Enter a character: " char

# Check if the input is a single character

if [ ${#char} -ne 1 ]; then

echo "Please enter only a single character."

exit 1

fi

# Check character type using regex

if [[ $char =~ [a-zA-Z] ]]; then

echo "$char is an alphabet"

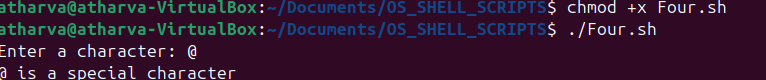
elif [[ $char =~ [0-9] ]]; then

echo "$char is a digit"

else

echo "$char is a special character"

fi



1. **To calculate profit or loss.**

#!/bin/bash

# Script to calculate profit or loss

# Get cost price and selling price from user

read -p "Enter cost price: " cp

read -p "Enter selling price: " sp

# Calculate profit or loss

if [ $sp -gt $cp ]; then

profit=$((sp - cp))

percentage=$(echo "scale=2; ($profit \* 100) / $cp" | bc)

echo "Profit: $profit"

echo "Profit percentage: $percentage%"

elif [ $cp -gt $sp ]; then

loss=$((cp - sp))

percentage=$(echo "scale=2; ($loss \* 100) / $cp" | bc)

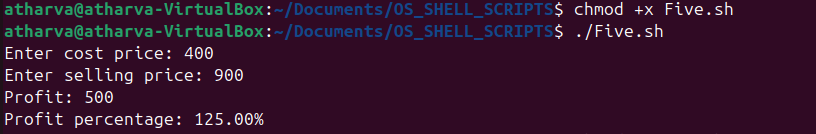
echo "Loss: $loss"

echo "Loss percentage: $percentage%"

else

echo "No profit, no loss"

fi



1. **To print all even and odd number from 1 to 10.**

#!/bin/bash

# Script to print all even and odd numbers from 1 to 10

echo "Even numbers from 1 to 10:"

for ((i=2; i<=10; i+=2)); do

echo $i

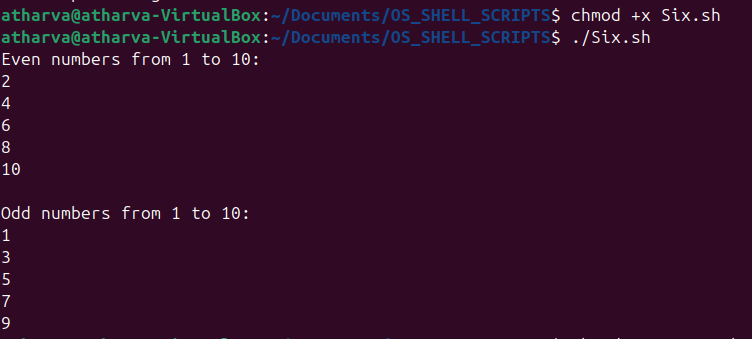
done

echo -e "\nOdd numbers from 1 to 10:"

for ((i=1; i<=10; i+=2)); do

echo $i

done



1. **To print table of a given number.**

#!/bin/bash

# Script to print the table of a given number

# Get a number from user

read -p "Enter a number to display its multiplication table: " num

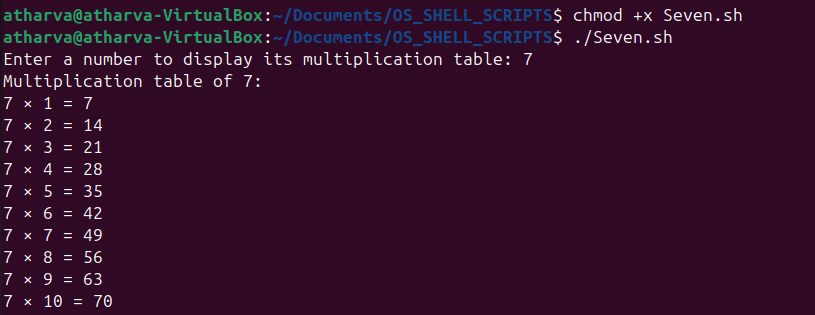
echo "Multiplication table of $num:"

for ((i=1; i<=10; i++)); do

result=$((num \* i))

echo "$num × $i = $result"

done



1. **To find factorial of a given integer.**

#!/bin/bash

# Script to find factorial of a given integer

# Get a number from user

read -p "Enter a positive integer: " num

# Validate input

if ! [[ $num =~ ^[0-9]+$ ]]; then

echo "Please enter a valid positive integer."

exit 1

fi

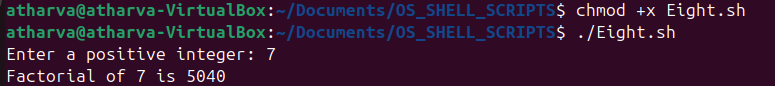
factorial=1

for ((i=1; i<=num; i++)); do

factorial=$((factorial \* i))

done

echo "Factorial of $num is $factorial"



1. **To print sum of all even numbers from 1 to 10.**

#!/bin/bash

# Script to print sum of all even numbers from 1 to 10

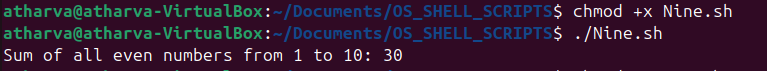
sum=0

for ((i=2; i<=10; i+=2)); do

sum=$((sum + i))

done

echo "Sum of all even numbers from 1 to 10: $sum"



1. **To print sum of digit of any number.**

#!/bin/bash

# Script to print sum of digits of any number

# Get a number from user

read -p "Enter a number: " num

# Remove any negative sign if present

num=${num#-}

sum=0

while [ $num -gt 0 ]; do

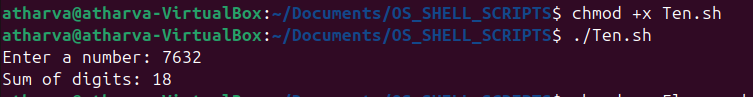
digit=$((num % 10))

sum=$((sum + digit))

num=$((num / 10))

done

echo "Sum of digits: $sum"



1. **To make a basic calculator which performs addition, subtraction, Multiplication, division.**

#!/bin/bash

# Script to create a basic calculator for arithmetic operations

echo "Basic Calculator"

echo "1. Addition"

echo "2. Subtraction"

echo "3. Multiplication"

echo "4. Division"

read -p "Enter your choice (1-4): " choice

read -p "Enter first number: " num1

read -p "Enter second number: " num2

case $choice in

1)

result=$((num1 + num2))

echo "$num1 + $num2 = $result"

;;

2)

result=$((num1 - num2))

echo "$num1 - $num2 = $result"

;;

3)

result=$((num1 \* num2))

echo "$num1 \* $num2 = $result"

;;

4)

if [ $num2 -eq 0 ]; then

echo "Error: Division by zero"

exit 1

fi

# Using bc for floating-point division

result=$(echo "scale=2; $num1 / $num2" | bc)

echo "$num1 / $num2 = $result"

;;

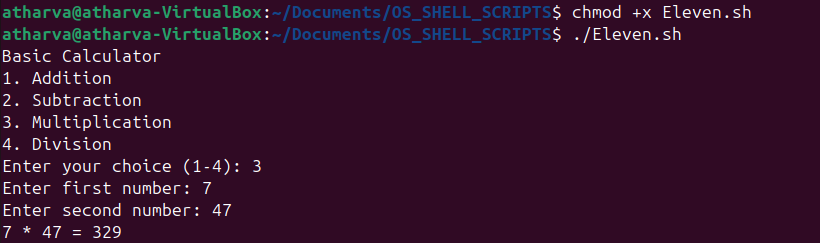
\*)

echo "Invalid choice"

exit 1

;;

Esac



1. **To print days of a week.**

#!/bin/bash

# Script to print days of a week

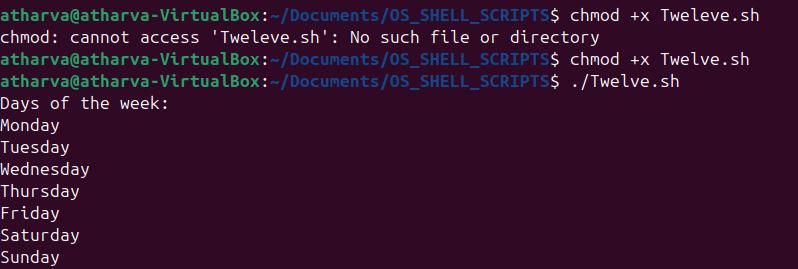
echo "Days of the week:"

days=("Monday" "Tuesday" "Wednesday" "Thursday" "Friday" "Saturday" "Sunday")

for day in "${days[@]}"; do

echo $day

done



1. **To print starting 4 months having 31 days.**

#!/bin/bash

# Script to print starting 4 months having 31 days

echo "First 4 months with 31 days:"

months\_with\_31\_days=("January" "March" "May" "July" "August" "October" "December")

# Print the first 4 months with 31 days

count=0

for month in "${months\_with\_31\_days[@]}"; do

echo $month

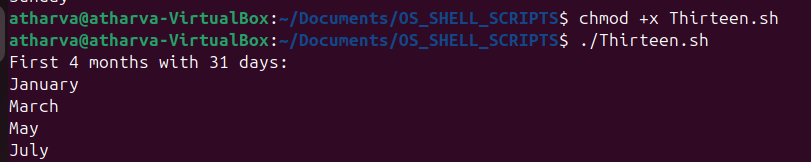
count=$((count + 1))

if [ $count -eq 4 ]; then

break

fi

done



1. **Using functions:**

* **To find given number is Armstrong number or not.**

#!/bin/bash

# Script to check if a number is Armstrong number

# Function to check if a number is Armstrong number

is\_armstrong() {

num=$1

original\_num=$num

num\_digits=${#num}

sum=0

while [ $num -gt 0 ]; do

digit=$((num % 10))

# Calculate digit raised to power of number of digits

power=1

for ((i=1; i<=num\_digits; i++)); do

power=$((power \* digit))

done

sum=$((sum + power))

num=$((num / 10))

done

if [ $sum -eq $original\_num ]; then

echo "$original\_num is an Armstrong number"

else

echo "$original\_num is not an Armstrong number"

fi

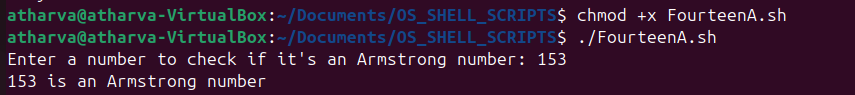
}

# Get a number from user

read -p "Enter a number to check if it's an Armstrong number: " number

# Call the function

is\_armstrong $number



* **To find whether a number is palindrome or not.**

#!/bin/bash

# Script to check if a number is palindrome

# Function to check if a number is palindrome

is\_palindrome() {

num=$1

original\_num=$num

reverse=0

while [ $num -gt 0 ]; do

digit=$((num % 10))

reverse=$((reverse \* 10 + digit))

num=$((num / 10))

done

if [ $original\_num -eq $reverse ]; then

echo "$original\_num is a palindrome"

else

echo "$original\_num is not a palindrome"

fi

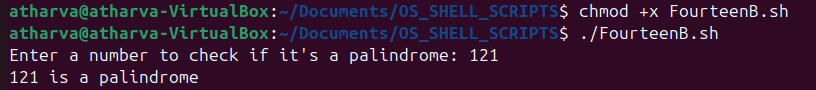
}

# Get a number from user

read -p "Enter a number to check if it's a palindrome: " number

# Call the function

is\_palindrome $number



* **To print Fibonacci series up to n terms.**

#!/bin/bash

# Script to print Fibonacci series up to n terms

# Function to print Fibonacci series

fibonacci() {

n=$1

a=0

b=1

echo "Fibonacci series up to $n terms:"

if [ $n -ge 1 ]; then

echo $a

fi

if [ $n -ge 2 ]; then

echo $b

fi

for ((i=3; i<=n; i++)); do

c=$((a + b))

echo $c

a=$b

b=$c

done

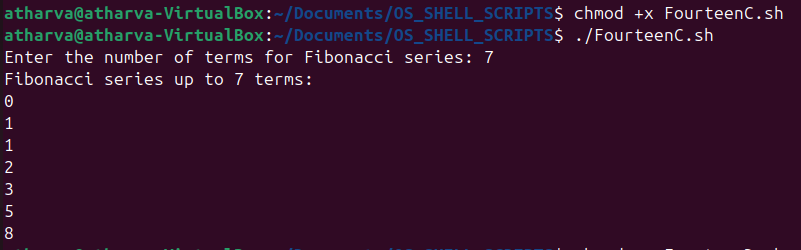
}

# Get number of terms from user

read -p "Enter the number of terms for Fibonacci series: " terms

# Call the function

fibonacci $terms



* **To find given number is prime or composite.**

#!/bin/bash

# Script to check if a number is prime or composite

# Function to check if a number is prime

is\_prime() {

num=$1

# 1 is neither prime nor composite

if [ $num -eq 1 ]; then

echo "1 is neither prime nor composite"

return

fi

# 2 is prime

if [ $num -eq 2 ]; then

echo "$num is a prime number"

return

fi

# Even numbers other than 2 are composite

if [ $((num % 2)) -eq 0 ]; then

echo "$num is a composite number"

return

fi

# Check odd divisors up to square root of num

for ((i=3; i\*i<=num; i+=2)); do

if [ $((num % i)) -eq 0 ]; then

echo "$num is a composite number"

return

fi

done

echo "$num is a prime number"

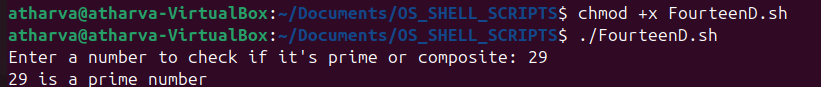
}

# Get a number from user

read -p "Enter a number to check if it's prime or composite: " number

# Call the function

is\_prime $number



* **To convert a given decimal number to binary equivalent.**

#!/bin/bash

# Script to convert decimal to binary

# Function to convert decimal to binary

decimal\_to\_binary() {

decimal=$1

if [ $decimal -eq 0 ]; then

echo "Binary equivalent of $decimal is 0"

return

fi

binary=""

while [ $decimal -gt 0 ]; do

remainder=$((decimal % 2))

binary="$remainder$binary"

decimal=$((decimal / 2))

done

echo "Binary equivalent of $1 is $binary"

}

# Get a decimal number from user

read -p "Enter a decimal number to convert to binary: " decimal\_num

# Call the function

decimal\_to\_binary $decimal\_num

