**SHELL SCRIPT**

**EXPT NO: 3 DATE: 13/9/2022**

**AIM**

[A] Write a shell script program to find maximum from 3 numbers.

[B] Write a shell script program to find if entered year is leap year or not.

[C] Write a shell script program to find factorial of a number.

[D] Write a shell script program to find whether entered number is even or odd.

[E] Write a shell script program to find reverse of a number.

[F] Write a shell script program to print fibonacci series.

[G] Write a shell script program to implement a menu driven program.

[H] Write a shell script program to find whether entered number is prime or not.

[I] Write a shell script program to find maximum digit of a number.

[J] Write a shell script program to find minimum digit of a number.

**THEORY**

Operators:

The different operators supported by shell script are:

1. Arithmetic Operators (+, - , \*, /, %, = )

2. Relational Operators

(a) -eq : Checks if the value of two operands are equal or not; if yes, then the condition becomes true.

(b) – ne: Checks if the value of two operands are equal or not; if values are not equal, then the condition becomes true.

(c) -gt: Checks if the value of left operand is greater than the value of right operand; if yes, then the condition becomes true.

(d) -lt: Checks if the value of left operand is less than the value of right operand; if yes, then the condition becomes true

(e) -ge: Checks if the value of left operand is greater than or equal to the value of right operand; if yes, then the condition becomes true.

(f) – le: Checks if the value of left operand is less than or equal to the value of right operand; if yes, then the condition becomes true.

Loops:

Loops allow us to take a series of commands and keep re-running them until a particular situation is reached. They are useful for automating repetitive tasks. The different types of loops supported by shell script are as follows:

1. While loop

The while loop enables you to execute a set of commands repeatedly until some condition occurs. It is usually used when you need to manipulate the value of a variable repeatedly.

Syntax:

while [condition]

do

// Code to be executed if condition is true

done

2. For loop

The for loop operates on lists of items. It repeats a set of commands for every item in a list

Syntax:

for var in list

do

// Code to be executed

done

Conditional Statements:

The following conditional statements are supported by shell script:

1. if statement

This block will process if specified condition is true.

Syntax:

if condition

then

statement

fi

2. if else statement

If specified condition is not true in if part then else part will be executed.

Syntax:

if condition

then

statement1

else

statement2

fi

3. case statement

Case statement works as a switch statement if specified value match with the pattern then it will execute a block of that particular pattern. When a match is found all of the associated statements until the double semicolon (;;) is executed. A case will be terminated when the last command is executed.

Syntax:

case var in

pattern1) statement 1;;

pattern2) statement 2;;

.

.

.

patternn) statement n;;

esac

**CODE**

[A] Write a shell script program to find maximum from 3 numbers.

echo "Enter num1:"

read num1

echo "Enter num2:"

read num2

echo "Enter num3:"

read num3

if [ $num1 -gt $num2 ]

then

if [ $num1 -gt $num2 ]

then

echo "Max=$num1"

else

echo "Max=$num2"

fi

else

if [ $num2 -gt $num3 ]

then

echo "Max=$num2"

else

echo "Max=$num3"

fi

OUTPUT:

[B] Write a shell script program to find if entered year is leap year or not.

echo "Enter the year"

read year

if [ $(($year%4)) -eq 0 ]

then

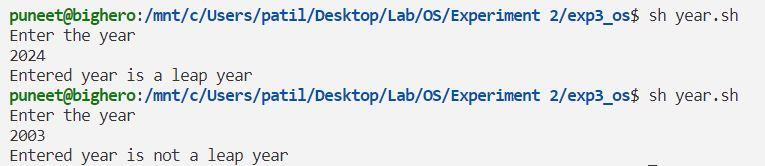
echo "Entered year is a leap year"

else

echo "Entered year is not a leap year"

fi

OUTPUT**:**



[C] Write a shell script program to find factorial of a number.

echo "Enter the number:"

read num

fact=1

while [ $num -gt 1 ]

do

fact=$(($fact\*$num))

num=$(($num-1))

done

echo "The factorial is $fact"

OUTPUT:



[D] Write a shell script program to find whether entered number is even or odd.

echo "Enter a number:"

read num

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

then

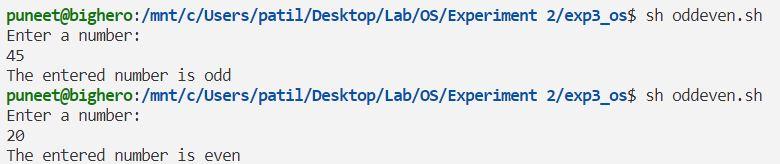
echo "The entered number is even"

else

echo "The entered number is odd"

fi

OUTPUT:



[E] Write a shell script program to find reverse of a number.

echo "Enter a number:"

read num

rev=0

while [ $num -ne 0 ]

do

  rev=$(($rev\*10))

  digit=$(($num%10))

  rev=$(($rev+$digit))

  num=$(($num/10))

done

echo "The reverse is $rev"

OUTPUT:



[F] Write a shell script program to print fibonacci series.

echo "Enter a number:"

read num

i=1

a=0

b=1

while [ $i -le $num ]

do

if [ $i -eq 1 ]

then

echo "$a "

fi

if [ $i -eq 2 ]

then

echo "$b "

fi

if [ $i -gt 2 ]

then

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

echo "$c "

a=$b

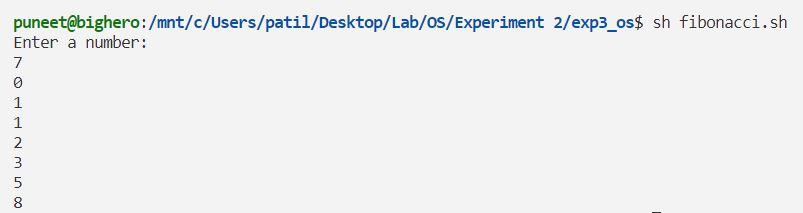
b=$c

fi

i=$(($i+1))

done

OUTPUT:



[G] Write a shell script program to implement a menu driven program.

ch=1

while [ $ch -eq 1 ]

do

echo "1. Display Calendar"

echo "2. Display Users"

echo "3. Display current directory"

read choice

case $choice in

1) cal

;;

2) who

;;

3) pwd

;;

\*) echo "Invalid choice"

;;

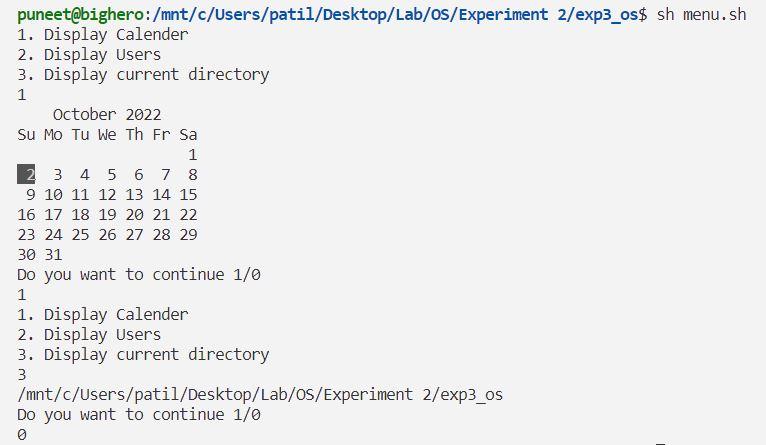
esac

echo "Do you want to continue 1/0"

read ch

done

OUTPUT:



[H] Write a shell script program to find whether entered number is prime or not.

echo "Enter a number:"

read num

i=2

flag=0

while [ $i -lt $num ]

do

x=$(($num%$i))

if [ $x -eq 0 ]

then

flag=1

fi

i=$(($i+1))

done

if [ $flag -eq 0 ]

then

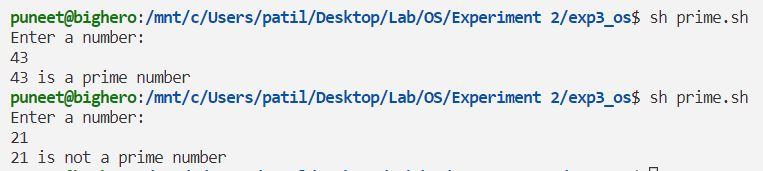
echo "$num is a prime number"

else

echo "$num is not a prime number"

fi

OUTPUT:



[I] Write a shell script program to find maximum digit of a number.

echo "Enter a number:"

read num

max=0

while [ $num -ne 0 ]

do

digit=$(($num%10))

if [ $digit -gt $max ]

then

max=$digit

fi

num=$(($num/10))

done

echo "The max digit is $max"

OUTPUT:



[J] Write a shell script program to find minimum digit of a number.

echo "Enter a number:"

read num

min=9

while [ $num -ne 0 ]

do

digit=$(($num%10))

if [ $digit -lt $min ]

then

min=$digit

fi

num=$(($num/10))

done

echo "The min digit is $min"

OUTPUT:



**CONCLUSION**

The given problem statements were successfully executed