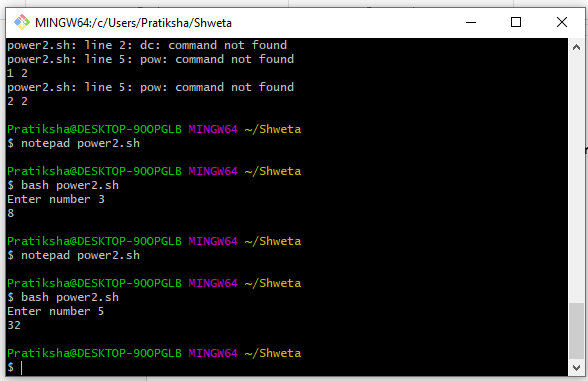
**Day 6**

**1.Write a program that takes a command-line argument n and prints a table of the powers of 2 that are less than or equal to 2^n.**

read -p "Enter number" num

echo $(( 2 \*\* num ))



**Write a program that takes a command-line argument n and prints the nth harmonic number. Harmonic Number is of the form**

read -p "Enter number: " num

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

do

$output += 1 / $i

done

echo " sum is $output"

**Write a program that takes a input and determines if the number is a prime.**

echo -e "Enter Number: "

read n

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

do

ans=$((n%i))

if [ $ans -eq 0 ]

then

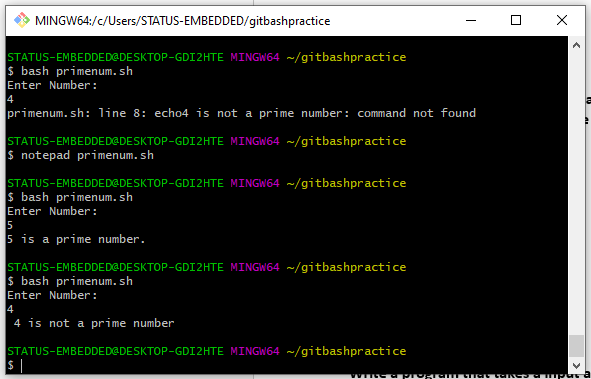
echo " $n is not a prime number "

exit 0

fi

done

echo "$n is a prime number."

****

**Extend the program to take a range of number as input and output the Prime Numbers in that range.**

echo -e "enter the range m and n:"

read m n

echo "The prime number between $m and $n are:"

for i in $(seq $m $n)

do

flag=0

for j in $(seq 2 $(($i-1)))

do

ans=$((i%j))

if [ $ans -eq 0 ]

then

flag=1

fi

done

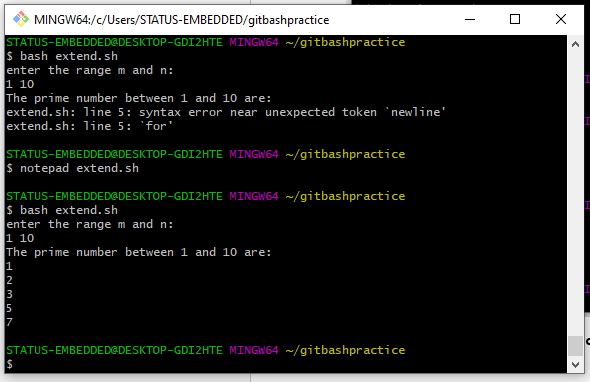
if [ $flag -eq 0 ]

then

echo $i

fi

done



**Write a program that computes a factorial of a number taken as input. 5 Factorial – 5! = 1 \* 2 \* 3 \* 4 \* 5**

read -p "Enter number " num

fact=1

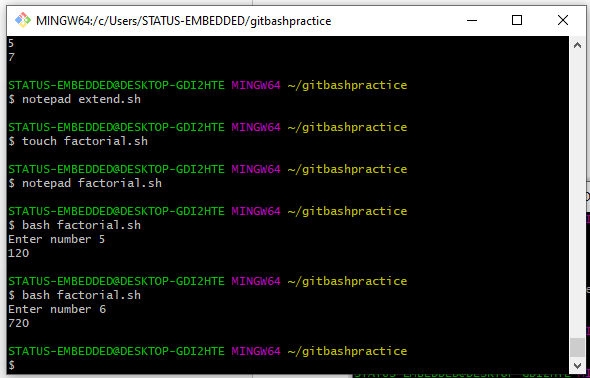
for i in $(seq $num)

do

fact=$((fact\*i))

done

echo $fact

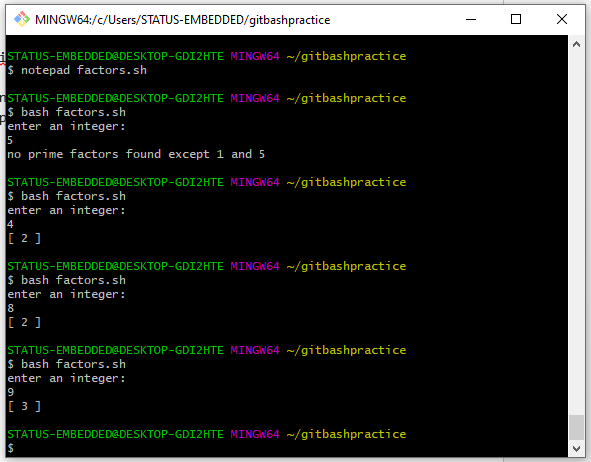


**Write a program to compute Factors of a number N using prime factorization method.**

**Logic -> Traverse till i\*i <= N instead of i <= N for efficiency.**

**O/P -> Print the prime factors of number N.**

|  |
| --- |
| echo "enter an integer:" |
|  | read input |
|  | if [ $input -lt 1 ];then |
|  | echo "not allowed!" |
|  | exit 1 |
|  | fi |
|  | # find factors and prime |
|  |  |
|  | i=2 |
|  | count=0 |
|  | flag=0 |
|  | for ((i;i<$input;));do |
|  |  |
|  | if [ `expr $input % $i` -eq 0 ];then |
|  | factor=$i |
|  |  |
|  | for ((j=2;j<=`expr $factor / 2`;));do |
|  | flag=0 |
|  | if [ `expr $factor % $j` -eq 0 ];then |
|  | flag=1 |
|  | break |
|  | fi |
|  | j=`expr $j + 1` |
|  | done |
|  | if [ $flag -eq 0 ];then |
|  | echo "[ $factor ]" |
|  | count=1 |
|  | fi |
|  | fi |
|  | i=`expr $i + 1` |
|  | done |
|  | if [ $count -eq 0 ];then |
|  | echo "no prime factors found except 1 and $input" |
|  | fi |

****

**Repetition Practice Problems with while loop**

**Write a program that takes a command-line argument n and prints a**

**table of the powers of 2 that are less than or equal to 2^n till 256 is**

**reached..**

**Find the Magic Number**

**a. Ask the user to think of a number n between 1 to 100**

**b. Then check with the user if the number is less then n/2 or greater**

**c. Repeat till the Magic Number is reached..**

echo "Think of a no. in between 1 to 100"

number=$((RANDOM%100+1))

echo $number

m=$((number/2))

if [ $number -le $m ]

then

echo "Number is less than N/2"

else

echo "Number is greater than N/2"

fi

no=$(((number-1)%9))

if [ $no -eq 0 ]

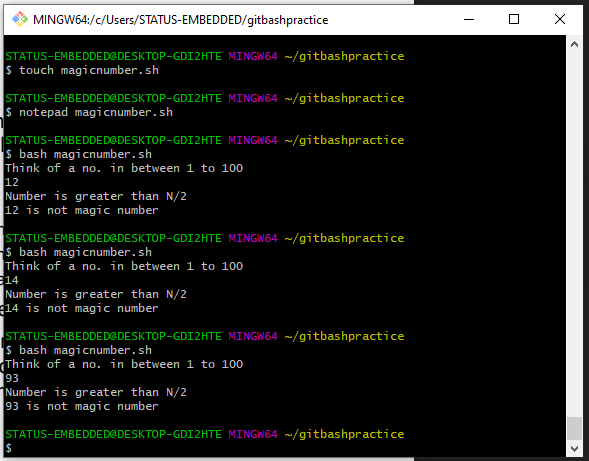
then

echo "$number is a magic number"

else

echo "$number is not magic number"

fi



**Extend the Flip Coin problem till either Heads or Tails wins 11 times.**

flips=1

heads=0

tails=0

while [ $flips -le 11 ]

do

Result=$((RANDOM%2))

flips=$((flips+1))

if [ ${Result} -eq 0 ]

then

echo "HEADS"

heads=$((heads+1))

elif [ ${Result} -eq 1 ]

then

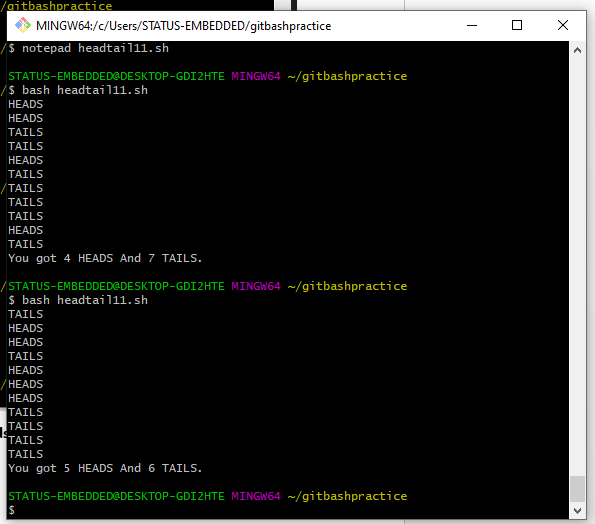
echo "TAILS"

tails=$((tails+1))

fi

done

echo "You got $heads HEADS And $tails TAILS."



**Write a Program where a gambler starts with Rs 100 and places Re 1 bet**

**until he/she goes broke i.e. no more money to gamble or reaches the**

**goal of Rs 200. Keeps track of number of times won and number of bets**

**made.**

**Help user find degF or degC based on their Conversion Selection. Use**

**Case Statement and ensure that the inputs are within the Freezing Point (**

**0 °C / 32 °F ) and the Boiling Point of Water ( 100 °C / 212 °F )**

**a. degF = (degC \* 9/5) + 32**

**b. degC = (degF – 32) \* 5/9**

echo "1. Convert Celsius temperature into Fahrenheit"

echo "2.Convert Fahrenheit temperature into Celsius"

echo -n "Select your choice(1-2):"

read choice

case $choice in

1 ) echo -n "Enter Temperature(C):"

read tc

tf=$(( (tc\*9/5)+32 ))

echo "$tc C= $tf F"

;;

2 ) echo -n "Enter Temperature(F):"

read tf

tc=$(( (tf-32)\*5/9 ))

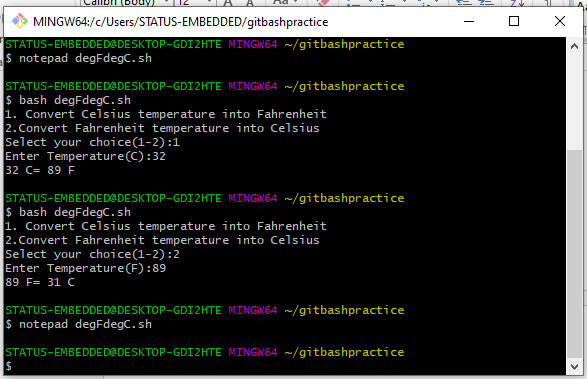
echo "$tf F= $tc C"

;;

\* ) echo -n "Wrong choice entered"

;;

esac

****

**Write a function to check if the two numbers are Palindromes**

Answer1:

echo " Enter the two number"

read mn

read n1

function pal()

{

number=$n

number1=$n1

reverse=0

local s=$1

for i in $s

do

while [ $i -gt 0 ]

do

a= ` expr $i % 10 `

n=` expr $i / 10 `

reverse=`expr $reverse \\* 10 + $a`

done

done

if [ $number -eq $reverse&&$number1 -eq $reverse ]

then

echo "Number is palindrome"

else

echo " Number is not Palindrome"

fi

}

r=`pal $n $n1`

echo "$r"

Answer2:

echo "Enter the number"

read n

echo "Enter second number"

read m

function pal

{

number=$n

num=$m

reverse=0

while [ $n -gt 0 ]

do

a=`expr $n % 10 `

n=`expr $n / 10 `

reverse=`expr $reverse \\* 10 + $a`

done

echo $reverse

if [ $number -eq $reverse ]

then

echo "$number is palindrome"

else

echo "$number is not palindrome"

fi

while [ $m -gt 0 ]

do

b=`expr $m % 10 `

m=`expr $m / 10 `

reverse=`expr $reverse \\* 10 + $b`

done

echo $reverse

if [ $num -eq $reverse ]

then

echo "$num is palindrome"

else

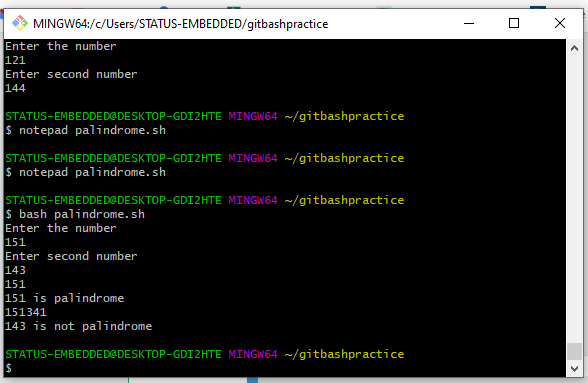
echo "$num is not palindrome"

fi

}

r=`pal $n $m`

echo "$r"

****

**Take a number from user and check if the number is a Prime then show that its palindrome is also prime**

**a. Write function check if number is Prime**

**b. Write function to get the Palindrome.**

**c. Check if the Palindrome number is also prime**

function palindromeNo()

{

number=$1

while [ $1 -gt 0 ]

do

a=$(($number%10))

rev=$(echo ${rev}${a})

r=$(($1 /10))

done

echo $reverse

if [ $number -eq $reverse ]

then

echo "Number is palindrome"

else

echo "Number is not Palindrome"

fi

}

function primeNo()

{

while [ $temp -le $1/2 ]

do

ans=$(($1%i))

if [ $ans -eq 0 ]

then

echo "$1 is not a prime number"

else

echo "$1 is prime number"

fi

done

palindromeNo $1

}

echo -e "Enter Number:\c"

read n

primeNo $n