* Repetition Practice Problems with for loop

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.

solution::

#!/bin/bash

# prints the table of the power of 2 . [ 2 ^ n ]

#base value of power=1

powerOftwo=1

read -p "enter any number " n

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

do

powerOftwo=$((2\*$powerOftwo))

echo " 2^$i = " $powerOftwo;

done

output ::

$ ./powerOftwo.sh

enter any number 16

2^1 = 2

2^2 = 4

2^3 = 8

2^4 = 16

2^5 = 32

2^6 = 64

2^7 = 128

2^8 = 256

2^9 = 512

2^10 = 1024

2^11 = 2048

2^12 = 4096

2^13 = 8192

2^14 = 16384

2^15 = 32768

2^16 = 65536

#######################################################################

2. Write a program that takes a command-line argument n and prints the nth harmonic

number. Harmonic Number is of the form

solution::

#!/bin/bash

harmonic=1

al=1

read -p "any number" n

#range=$(($n+1))

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

do

harmonic=$(( $((1/$i)) + $harmonic ))

#done

echo $harmonic

done

######################################################################

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

Solution::

#! /bin/bash -x

#prime num is divisible by 1 and itself only so count willbe 2.

count=0;

read -p "enter any number to check weather it is prime number or not " number

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

do

if [[ $number%$i -eq 0 ]]

then

count=$(($count+1))

fi

done

if [ $count -le 2 ]

then

echo " number $number is Prime "

else

echo "number $number is not a prime number"

fi

output::

$ ./primenum.sh

+ count=0

+ read -p 'enter any number to check weather it is prime number or not ' number

enter any number to check weather it is prime number or not 7

+ (( i=1 ))

+ (( i<=7 ))

+ [[ 7%1 -eq 0 ]]

+ count=1

+ (( i++ ))

+ (( i<=7 ))

+ [[ 7%2 -eq 0 ]]

+ (( i++ ))

+ (( i<=7 ))

+ [[ 7%3 -eq 0 ]]

+ (( i++ ))

+ (( i<=7 ))

+ [[ 7%4 -eq 0 ]]

+ (( i++ ))

+ (( i<=7 ))

+ [[ 7%5 -eq 0 ]]

+ (( i++ ))

+ (( i<=7 ))

+ [[ 7%6 -eq 0 ]]

+ (( i++ ))

+ (( i<=7 ))

+ [[ 7%7 -eq 0 ]]

+ count=2

+ (( i++ ))

+ (( i<=7 ))

+ '[' 2 -le 2 ']'

+ echo ' number 7 is Prime '

number 7 is Prime

#######################################################################

4. Extend the program to take a range of number as input and output the Prime

Numbers in that range.

Solution ::

#! /bin/bash

#prime num is divisible by 1 and itself only so count willbe 2.

#number=1

count=0;

read -p "enter any range to check all prime numbers " range

for ((number=1; number<=$range; number++ ))

do

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

do

if [[ $number%$i -eq 0 ]]

then

count=$(($count+1))

fi

done

if [ $count -lt 3 ]

then

echo " $number is Prime "

count=0

else

count=0

fi

done

Output::

$ ./primeinRange.sh

enter any range to check all prime numbers 20

1 is Prime

2 is Prime

3 is Prime

5 is Prime

7 is Prime

11 is Prime

13 is Prime

17 is Prime

19 is Prime

###########################################################################

5. Write a program that computes a factorial of a number taken as input.

5 Factorial – 5! = 1 \* 2 \* 3 \* 4 \* 5

Solution::

#!/bin/bash -x

# factorial num {n!=n(n-1)(n-2)........1}

fact=1;

read -p " enter any number to find factorial " number

if [ $number -gt 0 ]

then

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

do

fact=$(($fact\*$i))

done

else

echo "enter any positive number "

fi

echo "Factorial of $number is " $fact

output::

$ ./factorial.sh

+ fact=1

+ read -p ' enter any number to find factorial ' number

enter any number to find factorial 5

+ '[' 5 -gt 0 ']'

+ (( i=1 ))

+ (( i<=5 ))

+ fact=1

+ (( i++ ))

+ (( i<=5 ))

+ fact=2

+ (( i++ ))

+ (( i<=5 ))

+ fact=6

+ (( i++ ))

+ (( i<=5 ))

+ fact=24

+ (( i++ ))

+ (( i<=5 ))

+ fact=120

+ (( i++ ))

+ (( i<=5 ))

+ echo 'Factorial of 5 is ' 120

Factorial of 5 is 120

6. 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.

**Repetition Practice Problems with while loop**

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 till 256 is

reached..

solution::

#!/bin/bash

# prints the table of the power of 2 . [ 2 ^ n ] till 256

#base value of power=1

powerOftwo=1

i=0;

read -p "enter any number " n

while [ $powerOftwo -ne 256 ]

do

if [[ $i -lt $n ]]

then

powerOftwo=$((2\*$powerOftwo))

i=$(($i+1))

echo " 2^$i = " $powerOftwo;

else

exit

fi

done

output

$ ./powerOftwoWhile.sh

enter any number 20

2^1 = 2

2^2 = 4

2^3 = 8

2^4 = 16

2^5 = 32

2^6 = 64

2^7 = 128

2^8 = 256

############################################################

2. 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..

solution::

#!/bin/bash

count=o

low=0

high=100

mid=50

echo " Think Any Number Between [ 1 -100 ] "

while [ $low -le $high ]

do

echo "is number is greater than"$mid "then press 1"

echo "is number is lower then "$mid "then press 2"

echo "is number is equals to " $mid "then press 3"

read key

case $key in

1)

low=$mid

mid=$(( ($high+$mid) / 2))

;;

2)

high=$mid

mid=$(( ($low+$mid) / 2 ))

;;

3)

echo " the magic no is " $mid

exit

;;

\*)

echo "please try again "

exit

;;

esac

mid=$((($low+$high)/2))

done

output

$ ./guessThenum2.sh

Think Any Number Between [ 1 -100 ]

is number is greater than50 then press 1

is number is lower then 50 then press 2

is number is equals to 50 then press 3

1

is number is greater than75 then press 1

is number is lower then 75 then press 2

is number is equals to 75 then press 3

2

is number is greater than62 then press 1

is number is lower then 62 then press 2

is number is equals to 62 then press 3

2

is number is greater than56 then press 1

is number is lower then 56 then press 2

is number is equals to 56 then press 3

1

is number is greater than59 then press 1

is number is lower then 59 then press 2

is number is equals to 59 then press 3

1

is number is greater than60 then press 1

is number is lower then 60 then press 2

is number is equals to 60 then press 3

3

the magic no is 60

##################################################

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

Solution::

#!/bin/bash

#flip

heads=0

tails=1

count\_Head=0

count\_Tail=0

count=0

#flip=$((RANDOM%2))

#echo $flip

while [ $count -lt 12 ]

do

flip=$((RANDOM%2))

if [ $flip -eq $heads ]

then

echo " HEADS WINS"

count\_Head=$(( $count\_Head + 1 ))

count=$((1+$count\_Head))

elif [ $flip -eq $tails ]

then

echo "TAILS WINS"

count\_Tail=$(( $count\_Tail + 1 ))

count=$((1+$count\_Tail))

else

echo Wexit

fi

#count=$(($count+1))

done

echo "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

echo "HEADS " $count\_Head

echo "TAILS " $count\_Tail

output::

$ ./coingame.sh

TAILS WINS

TAILS WINS

HEADS WINS

HEADS WINS

HEADS WINS

HEADS WINS

TAILS WINS

HEADS WINS

HEADS WINS

HEADS WINS

TAILS WINS

HEADS WINS

HEADS WINS

HEADS WINS

HEADS WINS

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

HEADS 11

TAILS 4

############################################################

4. 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.

Solution::

#!/bin/bash

#flip

win=1

loss=0

win\_count=0

loss\_count=0

money=100

goal1=200

goal2=0

while [ $goal1 -ne $money ] || [ $goal2 -ne $money ]

do

bet=$((RANDOM%2))

if [ $bet -eq $win ]

then

echo "BET WINS"

win\_count=$(( $win\_count + 1 ))

money=$(($money+1))

echo "available money " $money

if [ $money -eq 200 ]

then

echo "win count:: " $win\_count

exit

fi

elif [ $bet -eq $loss ]

then

echo "BET LOSS"

loss\_count=$(( $loss\_count + 1 ))

money=$(($money-1))

echo "available money " $money

if [ $money -eq 0 ]

then

echo " loss count:: " $loss\_count

exit

fi

else

echo exit

fi

done

echo $win\_count

echo $loss\_count

output::…. Last 5 outputs

BET WINS

available money 197

BET WINS

available money 198

BET WINS

available money 199

BET WINS

available money 200

win count::  3668

**Functions Practice Problems**

1. 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

solution::

#!/bin/bash

function ferToCel()

{

read -p "enter degree in celsius" degC

if [[ $degC -ge 0 && $degC -le 100 ]]

then

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

echo "$degC c celsius in ferenhite" $degF "f"

else

echo " degree not in range [0C - 100C] "

fi

}

function celToFer(){

read -p "enter degree in ferenhite " degF

if [[ $degF -ge 32 && $degF -le 212 ]]

then

degC=$(( ($degF - 32) \* 5/9 ))

echo "$degF f ferenhite in celsius "$degC "c"

else

echo "degree not in range [32F -212F]"

fi

}

echo "please make a choice.. "

echo " 1. ferhanite to celsius "

echo " 2. celsius to ferenhite "

read -p "choice" choice

case $choice in

1)

ferToCel

echo $ferToCel

;;

2)

celToFer

echo $celToFer

;;

\*)

echo "wrong choice"

;;

esac

output::

$ ./func\_Conversion.sh

please make a choice..

1. ferhanite to celsius

2. celsius to ferenhite

choice2

enter degree in ferenhite 32

32 f ferenhite in celsius 0 c

##########################################################

2. Write a function to check if the two numbers are Palindromes

Solution::

#!/bin/bash

function palindromcheck() {

number=$num

while [[ $number -gt 0 ]]

do

rmdr=$(($number%10));

number=$(($number/10))

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

done

echo "revsre is " $rev

if [[ $tmp -eq $rev ]]

then

echo "palindrom"

else

echo "not a palindrom"

fi

}

rev=0

rmdr=0

read -p "enter any num" num

tmp=$num

palindromcheck

output::

$ sh -x ./palindrom.sh

+ rev=0

+ rmdr=0

+ read -p 'enter any num' num

enter any num4114

+ tmp=4114

+ palindromcheck

+ number=4114

+ [[ 4114 -gt 0 ]]

+ rmdr=4

+ number=411

+ rev=4

+ [[ 411 -gt 0 ]]

+ rmdr=1

+ number=41

+ rev=41

+ [[ 41 -gt 0 ]]

+ rmdr=1

+ number=4

+ rev=411

+ [[ 4 -gt 0 ]]

+ rmdr=4

+ number=0

+ rev=4114

+ [[ 0 -gt 0 ]]

+ echo 'revsre is ' 4114

revsre is 4114

+ [[ 4114 -eq 4114 ]]

+ echo palindrom

palindrom

####################################################

3. 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

solution::

#!/bin/bash

function palindromcheck() {

number=$num

while [[ $number -gt 0 ]]

do

rmdr=$(($number%10))

number=$(($number/10))

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

done

echo "revsre is " $rev

if [[ $tmp -eq $rev ]]

then

echo "number $rev is a Palindrom"

else

echo "not a Palindrom"

fi

}

function primechk()

{

count=0;

number=$num

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

do

if [[ $number%$i -eq 0 ]]

then

count=$(($count+1))

fi

done

if [ $count -le 2 ]

then

echo " number $number is Prime "

else

echo "number $number is not a prime number"

fi

}

rev=0

rmdr=0

read -p "enter any num" num

tmp=$num

palindromcheck $num

primechk $num

output::

$ ./palindrom\_prime.sh

enter any num191

revsre is 191

number 191 is a Palindrom

number 191 is Prime