Repetition Practice Problems with for loop

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

```
#!/bin/bash -x
read -p "Power Number:" n;
for (( i=0;i<=$n;i++ ))
do
  power=$((2**$i));
  echo $power;
done
```

2) Write a program that takes a command-line argument n and prints the nth harmonic number. Harmonic Number is of the form
$Hn = \frac{1}{1} + \frac{1}{2} + \frac{1}{3} + ... + \frac{1}{n}$

```
#!/bin/bash -x
read -p "Enter a number:" number;
harmonic=0;
for(( i=1;i<=$number;i++ ))
do
  harm=`echo $i | awk '{div = 1/$i; printf "%f", div}'`;
  harmonic=`echo $harmonic $harm | awk '{add=$1+$2; printf "%f", add}'`;
done
echo $harmonic;
```

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

```
 #!/bin/bash -x
read -p "Enter Number:" number;
count=0;
for(( i=1; i<=$number; i++ ))
do
  n=$(($number%$i));
  if [ $n -eq 0 ];
  then
    count=$(($count+1));
  fi
done
if [ $count -eq 2 ];
then
  echo $number " is Prime Number";
```

```
else
  echo $number " is not Prime Number";
fi
```

4) Extend the program to take a range of numbers as input and output the Prime Numbers in that range.

```
#!/bin/bash -x
read -p "Enter First number:" firstNumber;
read -p "Enter Second number:" secondNumber;
for(( i=$firstNumber;i<=$secondNumber;i++))
do
  count=0;
  for((j=1;j<=$i;j++))
  do
    n=$(($i%$j));
    if [ $n -eq 0 ];
    then
      count=$(($count+1));
    fi
  done
  if [ $count -eq 2 ];
  then
    echo $i " is Prime Number";
  else
    echo $i " is not Prime Number";
  fi
done
```

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

```
#!/bin/bash -x
read -p "Enter a number:" number;
factorial=1;
if [ $number -gt 0 ];
then
  for(( i=$number;i>=1;i-- ))
  do
    factorial=$(( $factorial*$i ));
  done
  echo "Factorial of number is" $factorial;
elif [ $number -eq 0 ];
```

```
then
  echo "Factorial of number is 1";
fi
```

6) Write a program to compute Factors of a number N using the prime factorization method.

```
#!/bin/bash -x
read -p "Enter Number:" number;

for (( i=2;i*i<=$number;i++ ))
do
  if [ $(($number%$i)) -eq 0 ]
  then
    factor=$i
    count=0;
    for(( j=1;j<=$factor;j++ ))
    do
      n=$(($factor%$j));
      if [ $n -eq 0 ];
      then
        count=$(($count+1));
      fi
    done
    if [ $count -eq 2 ];
    then
      echo "Prime Factors are:" $factor;
    fi
  fi
done
```

Repetition Practice Problems with the 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.

```
#!/bin/bash -x
read -p "Power of Number:" number;
power=0;
count=0;
while [[ count -le $number && $power -lt 256 ]]
do
  power=$((2**$count));
  count=$(($count+1));
```

done

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.

```
#!/bin/bash -x

firstNumber=1;
lastNumber=100;
read -p "Think the number between 1 to 100:" number;
middle=$(( ($firstNumber+$lastNumber)/2 ));

while [[ $firstNumber -le $lastNumber ]]
do
  if [[ $middle -eq $number ]]
  then
    echo "Your magic Number is:" $middle;
    break;
  elif [[ $number -lt $middle ]]
  then
    lastNumber=$middle;
    middle=$(( ($firstNumber+$lastNumber)/2 ));
  else
    firstNumber=$middle;
    middle=$(( ($firstNumber+$lastNumber)/2 ));
  fi
done
```

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

```
#!/bin/bash -x
isHead=1;
count=0;
while [ $count -le 11 ]
do
  headTailCheck=$((RANDOM%2));
  if [ $isHead -eq $headTailCheck ]
  then
    echo "Heads";
  else
    echo "Tails";
```

```
  fi
  count=$(($count+1));
done
```

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 the number of times won and the number of bets made.

```
#!/bin/bash -x

money=100
target=200

read -p "How many time your play : " play;

while [ $play != 0 ]
do
  randomCheck=$(( RANDOM%10 ))
  if [ $randomCheck -ge 5 ]
  then
    echo "you win Re 1 ";
    ((money++));

    if [ $money -eq $target ]
    then
      echo "Your target is complited";
      $play=0;
    fi
  else
    echo "you loss Re 1";
    ((money--));
     if [ $money -eq 0 ]
     then
       echo "Sorry your balance is : '0'";
       $play=0;
     fi
  fi
  ((play--))
done

echo "your money is : $money";
```

Function Practice Problems

1) Help users find degF or degC based on their Conversion Selection. Use Case Statement and ensure that the inputs are within the Freezing Point (0 ℃ / 32 ℉)and the Boiling Point of Water (100 ℃ / 212 ℉)
    a) degF = (degC * 9/5) + 32
    b) degC = (degF - 32) * 5/9

```
#!/bin/bash -x

function degF() {
  read -p "Enter Temperature in Celsius:" degC;
  degF=$(((( $degC * 9/5 )) +32 ));
  echo $degF;
}
function degC() {
  read -p "Enter Temperature in Fahrenheit:" degF;
  degC=$((($degF - 32)) * 5/9 ));
  echo $degC;
}

echo "Temperature Conversion";
echo "1) degC to degF";
echo "2) degF to degC";
read -p "Enter Your Choice:" choice;
case $choice in
  1)
    degF
    ;;
  2)
    degC
    ;;
  *)
    echo "Invalid Inpute"
    ;;
esac
```

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

```
#!/bin/bash -x

function reverseOfNumber() {
  local firstNumber=$1;
  local reverseNumber=0;
```

```
  local lastDigit=0;
  while [[ $firstNumber -ne 0 ]]
  do
    lastDigit=$(( $firstNumber%10 ));
    reverseNumber=$(( $reverseNumber*10+$lastDigit ));
    firstNumber=$(( $firstNumber/10 ));
  done
  echo $reverseNumber;
}

function isPalindrome() {
  local reverseNumber=$1;
  local secondNumber=$1;
  if [[ $secondNumber -eq $reverseNumber ]]
  then
    echo "Both numbers are palindrome";
  else
    echo "Both numbers are not palindrome";
  fi
}

read -p "Enter First Number:" firstNumber;
read -p "Enter Second Numbe:" secondNumber;

reversNumber=$(reverseOfNumber $firstNumber);
isPalindrome $reverseNumber $secondNumber;
```

3) Take a number from user and check if the number is a Prime then show that its palindrome is also prime
    a) Write a function check if a number is Prime.
    b) Write a function to get the Palindrome.
    c) Check if the Palindrome number is also prime.

```
#!/bin/bash -x
read -p "Enter number:" number;

function isPrime() {
  local count=0;
  for(( i=1; i<=$number; i++ ))
  do
    n=$(($number%$i));
    if [ $n -eq 0 ];
    then
```

```
        count=$(($count+1));
      fi
    done
    if [ $count -eq 2 ];
    then
      echo "1";
    else
      echo "0";
    fi
}

function isPalindrome() {
    local rev=0;
    while [ $number -eq 0 ];
    do
      rev=$(( $rev*10+$number%10 ));
      number=$(($number/10));
      if [ $rev -eq $number ];
      then
        echo "1";
      else
        echo "0";
      fi
    done
}

prime=$(isPrime $number);
if [ $prime -eq 1 ]
then
    palindrome=$(isPalindrome $number);
    if [ $palindrme -eq 1 ]
    then
      echo "$number is prime number and also palindrome";
    else
      echo "$number is prime number but not palindrome";
    fi
else
    echo "$number is not a prime number";
fi
```