Assignment 2

Satyam Kumar
ID: 201552062

August 31, 2018

1 Print the decremented numerical value of a given number N.

```
#!/bin/bash

echo "Enter Integer Number"
read num

echo "Number entered is $num"

for (( i=num; i!=0;i-- ))
do
   if [ $i -ge 0 ]
   then
       echo -ne "$i\n";
   else
       echo -e "Positive integer (greater or equal to 0) is accepted.\n
           Terminal will close after 5seconds"  && sleep 5 && exit 1;
   fi
done
```

2 Print the sum of all digit of a given number N.

```
#!/bin/sh

echo "Enter number :"
read num

rem=0
sum=0

while [ $num -gt 0 ]
do
```

```
   rem=$(( $num % 10 ))
   num=$(( $num / 10 ))
   sum=$(( $sum + $rem ))
done
echo "Sum of digits of number $num is $sum"
```

3 Find whether entered year is leap year or not.

```
#!/bin/bash

echo "Enter the year"
read year

echo "Year entered is $year"

if [ $((year % 4)) -eq 0 ]
then
  if [ $((year % 100)) -eq 0 ]
    then
    if [ $((year % 400)) -eq 0 ]
          then
        echo "$year is a Leap Year"
    else
           echo "$year is not a Leap Year"
    fi
  else
  echo "$year is not a Leap Year."
  fi
else
   echo "$year is not a leap year"
fi
```

4 Convert file names from uppercase to lowercase and vice versa.

4.1 Uppercase to Lowercase

```
#!/bin/bash
echo "Converting filename from upper case to lower case."
for f in *;
        do mv -- "$f" "$(tr [:upper:] [:lower:] <<< "$f")";
done
```

4.2 Lowercase to Uppercase

```
#!bin/bash
```

```
#find . -type f -name '*.*'
#Lower to Upper Filename Conversion
echo "Converting filename from lower case to upper case."
for f in *;
do mv -- "$f" "$(tr [:lower:] [:upper:] <<< "$f")";
done
```

5 Calculate the biggest number from the three numbers supplied as command-line arguments.

```
#!/bin/bash

#3 numbers are a,b,c

a=$1
b=$2
c=$2

if [ $# -ne 3 ]
then
   echo "Some arguments are missing."&& exit 1
fi

#Comparing each argument with other.
#-a is used for as an extension for argument
 if [ $1 -eq $2 -a  $1 -eq $2 ]
 then
   echo "All arguments have equal values"
 elif [[  $1 -eq $2 && $1 -ge $3  ||  $2 -eq $3 && $2 -ge $1  ||
$3 -eq $1 && $1 -ge $2  ]]
 then
   echo "Greatest Number cannot be figured out."
 elif [ $1 -gt $2 -a $1 -gt $3 ]
 then
   echo "$1 is the Greatest Number"
  elif [ $2 -gt $3 -a $2 -gt $1 ]
  then
        echo "$2 is the Greatest number"
  else
        echo "$3 is the Greatest number"
fi
```

6 Generate basic math calculator (with case and select statements).

```
#!/bin/bash
```

```
echo -ne "Enter two number\n"
read a b

echo -n "Select your choice"
options=("Addition" "Subtraction" "Multiplication" "Division" "Quit")

select opt in "${options[@]}"
do
   case $opt in
                "Addition")
                        sum=$(($a + $b))
                echo "$sum"
                    break
                    ;;
                "Subtraction")
                        sub=$(($a - $b))
                    echo "$sub"
                    break
                    ;;
                "Multiplication")
                        mult=$(($a * $b))
                    echo "$mult"
                    break
                    ;;
                "Division")
                        div=$(($a / $b))
                    echo "$div"
                    if [ $b -eq 0 ]
                    then
                       echo -e "Division by Zero is not possible.\n
                             Terminal closes in 5s" && sleep 5 && exit 1;
                    fi
                    break
                    ;;
                "Quit")
                    break
                    ;;
   esac
done
```

7 Generate math calculator for real numbers (with select statements).

```
#!/bin/bash
```

```
echo -ne "Enter two real number\n"
read a b

echo -n "Select your choice"
options=("Addition" "Subtraction" "Multiplication" "Division" "Quit")

select opt in "${options[@]}"
do
        case $opt in
                "Addition")
                                echo "scale=4;$a + $b" | bc
                                break
                                ;;
                "Subtraction")
                                echo "scale=4;$a - $b" | bc
                                break
                                ;;
                "Multiplication")
                                echo "scale=4;$a * $b" | bc
                                break
                                ;;
                "Division")
                    if [ $b = 0 ];
                    then
                            echo -e "Division by Zero is not possible.\n
                            Terminal closes in 5s" && sleep 5 && exit 1;
                        else
                                echo "scale=4; $a / $b" | bc
                     fi
            break
                    ;;
                "Quit")
                                break
                                ;;
        esac
done
```

8 Print the last day (name) of the month, given the month and year information, e.g.,given 08-2018, it should print "Friday".

```
#!/bin/bash

#Calculating last date of month, which is set to 0.
for i in {0..24};
```

```
do
                date -d "-$(date +%d) days +$i month" >> last_date.txt
done
```

9 Calculate the factorial of a given number.

```
#!/bin/bash

Factorial()
{
                num=$1
                fact=1
                while [ $num -ge 1 ]
                do
                   fact=`expr $fact \* $num `
                   num=`expr $num - 1`
                done
                echo  "$fact" | bc
}

Factorial $1
```

10 Print any message in bold, blink effect, and in different colors.

```
#!/bin/bash
#Ref-1. https://misc.flogisoft.com/bash/tip_colors_and_formatting

read -p "Type your message.." msg

#Bold effect
echo -e "\033[1m $msg"

#echo -e "\033[7m $msg"
#Red Color
echo -e "\033[31m$msg"

#Green Color
echo -e "\033[32m$msg"

#Yellow Color
echo -e "\033[33m$msg"

#Blink effect
echo -e "\e[5m$msg"
```

```
#Normal
echo -e "\033[0m $msg"
```

11 Collect system information such as CPU, disks, etc. and store (append) it in a file with date/time information.

```
#!/bin/bash

CPU=$(top -b -n1 >> Log.txt)
MEM=$(top -b -n1 | grep "KiB Mem" | head -1 >> Memory_log.txt )
```