Name: Saatvik Kosta

Roll no.: 23I4061

**Lab Assignment 3: Linux Shell Scripts**

**1. Find the Largest of Three Numbers**

echo "Enter three numbers:"

read a b c

if [ $a -ge $b ] && [ $a -ge $c ]; then

echo "$a is the largest"

elif [ $b -ge $a ] && [ $b -ge $c ]; then

echo "$b is the largest"

else

echo "$c is the largest"

fi

2. **Check Leap Year**

echo "Enter a year:"

read year

if (( (year % 4 == 0 && year % 100 != 0) || (year % 400 == 0) )); then

echo "$year is a leap year"

else

echo "$year is not a leap year"

fi

3. **Check Triangle Validity**

echo "Enter three angles of a triangle:"

read a b c

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

if [ $sum -eq 180 ]; then

echo "Valid triangle"

else

echo "Invalid triangle"

fi

**4. Check Character Type**

echo "Enter a character:"

read ch

if [[ $ch =~ [a-zA-Z] ]]; then

echo "Alphabet"

elif [[ $ch =~ [0-9] ]]; then

echo "Digit"

else

echo "Special character"

fi

**5. Calculate Profit or Loss**

echo "Enter cost price:"

read cp

echo "Enter selling price:"

read sp

if [ $sp -gt $cp ]; then

profit=$((sp - cp))

echo "Profit: $profit"

elif [ $cp -gt $sp ]; then

loss=$((cp - sp))

echo "Loss: $loss"

else

echo "No profit no loss"

fi

6. **Print Even and Odd Numbers from 1 to 10**

for ((i=1; i<=10; i++))

do

if (( i % 2 == 0 )); then

echo "$i is Even"

else

echo "$i is Odd"

fi

done

7. **Print Table of a Given Number**

echo "Enter a number:"

read n

for ((i=1; i<=10; i++))

do

echo "$n x $i = $((n \* i))"

done

8. **Find Factorial of a Number**

echo "Enter a number:"

read n

fact=1

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

do

fact=$((fact \* i))

done

echo "Factorial of $n is $fact"

**9. Sum of All Even Numbers from 1 to 10**

sum=0

for ((i=2; i<=10; i+=2))

do

sum=$((sum + i))

done

echo "Sum of all even numbers from 1 to 10 is $sum"

**10.** **Sum of Digits of a Number**

echo "Enter a number:"

read n

sum=0

while [ $n -gt 0 ]

do

digit=$((n % 10))

sum=$((sum + digit))

n=$((n / 10))

done

echo "Sum of digits is $sum"

**11. Basic Calculator**

echo "Enter two numbers:"

read a b

echo "Choose operation (+ - \* /):"

read op

case $op in

+) echo "$a + $b = $((a + b))";;

-) echo "$a - $b = $((a - b))";;

\\*) echo "$a \* $b = $((a \* b))";;

/) echo "$a / $b = $((a / b))";;

\*) echo "Invalid operation";;

Esac

**12. Print Days of a Week**

days=("Sunday" "Monday" "Tuesday" "Wednesday" "Thursday" "Friday" "Saturday")

for day in "${days[@]}"

do

echo "$day"

done

**13. Print Starting 4 Months Having 31 Days**

months=("January" "March" "May" "July" "August" "October" "December")

for ((i=0; i<4; i++))

do

echo "${months[$i]}"

done

**14. Using Functions**

**a. Armstrong Number**

is\_armstrong() {

num=$1

sum=0

n=$num

while [ $n -gt 0 ]

do

digit=$((n % 10))

sum=$((sum + digit \* digit \* digit))

n=$((n / 10))

done

if [ $sum -eq $num ]; then

echo "$num is an Armstrong number"

else

echo "$num is not an Armstrong number"

fi

}

echo "Enter a number:"

read n

is\_armstrong $n

**b. Palindrome Number**

is\_palindrome() {

num=$1

rev=0

n=$num

while [ $n -gt 0 ]

do

digit=$((n % 10))

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

n=$((n / 10))

done

if [ $rev -eq $num ]; then

echo "$num is a palindrome"

else

echo "$num is not a palindrome"

fi

}

echo "Enter a number:"

read n

is\_palindrome $n

**c. Fibonacci Series**

fibonacci() {

n=$1

a=0

b=1

echo "Fibonacci series up to $n terms:"

for ((i=0; i<n; i++))

do

echo -n "$a "

fn=$((a + b))

a=$b

b=$fn

done

echo

}

echo "Enter number of terms:"

read n

fibonacci $n

**d. Prime or Composite**

is\_prime() {

num=$1

if [ $num -le 1 ]; then

echo "$num is neither prime nor composite"

return

fi

for ((i=2; i<=num/2; i++))

do

if ((num % i == 0)); then

echo "$num is composite"

return

fi

done

echo "$num is prime"

}

echo "Enter a number:"

read n

is\_prime $n

**e. Decimal to Binary**

decimal\_to\_binary() {

num=$1

bin=""

while [ $num -gt 0 ]

do

rem=$((num % 2))

bin="$rem$bin"

num=$((num / 2))

done

echo "Binary: $bin"

}

echo "Enter a decimal number:"

read n

decimal\_to\_binary $n