MC-214 LAB–0 REPORT

NISARG SUTHAR

202003030

EXERCISE 1: CREATE A SHELL SCRIPT TO ADD, SUBTRACT AND MULTIPLY TWO HARD-CODED NUMBERS.

SHELL SCRIPT:

```
# Ex1 : Shell script to find sum, difference and product of two hard-coded numbers.
# 13 and 7 for example.

echo -e "\n The numbers are : 13 and 7\n"   # -e is used to interpret the \n escape character

echo " Sum        : " `expr 13 + 7`    # sum of the numbers
echo " Difference : " `expr 13 - 7`    # difference between numbers
echo " Product    : " `expr 13 \* 7`   # product of numbers

echo  # prints empty line to make the output look nice

# script completed
```

INPUT:

Nil

OUTPUT:

```
 The numbers are : 13 and 7

 Sum        :  20
 Difference :  6
 Product    :  91
```

EXPLANATION:

Here 13 and 7 are taken as two sample numbers.
Then, their sum, difference and product are evaluated using the **expr** command and printed on the terminal using the **echo** command.

EXERCISE 2: CREATE A SHELL SCRIPT TO ADD, SUBTRACT AND MULTIPLY TWO NUMBERS AS INPUT FROM USERS.

SHELL SCRIPT:

```
# Ex2 : Shell script to find sum, difference and product of two
numbers taken as input from user.

echo  # prints empty line to make the output look nice

read -p " Enter first number  : " a         # user input using read
command
read -p " Enter second number : " b

echo

echo " Sum        : " `expr $a + $b`   # sum of the numbers
echo " Difference : " `expr $a - $b`   # difference between numbers
echo " Product    : " `expr $a \* $b`  # product of numbers

echo

# script completed
```

INPUT:

```
 Enter first number  : 13
 Enter second number : 7
```

OUTPUT:

```
 Sum        :  20
 Difference :  6
 Product    :  91
```

EXPLANATION:

Here 13 and 7 are taken as input from user and stored into variables ‘a’ and ‘b’.
To access the value of the variable **$** is used before the variable.
Then, their sum, difference and product are evaluated using the **expr** command and printed on the terminal using the **echo** command.

EXERCISE 3: CREATE A SHELL SCRIPT TO ADD, SUBTRACT AND MULTIPLY TWO NUMBERS AS INPUT FROM COMMAND LINE ARGUMENTS. ALSO, ADD THE OPERATION OF NEGATION IF THERE IS ONLY ONE COMMAND-LINE ARGUMENT.

SHELL SCRIPT:

```
# Ex3 : Shell script to find sum, difference and product of two
numbers taken as input from user.

if [ $# -eq 2 ] # this condition checks if the input is incorrect
then

	a=$1  # place holders for user input
	b=$2

	echo -e "\n The numbers are : $a and $b \n" # -e is used to
interpret the \n escape character

	echo " Sum        : " `expr $a + $b`   # sum of the numbers
	echo " Difference : " `expr $a - $b`   # difference between
numbers
	echo " Product    : " `expr $a \* $b`  # product of numbers

elif [ $# -eq 1 ]
then

	a=$1  # place holder for user input

	echo -e "\n Negation : -$a"# prints the negation of input

else

	echo -e "\n Incorrect Input : Either 1 or 2 numbers expected.
\n"  # error message
	exit  # exit from program execution

fi         # end of if condition

echo  # prints empty line to make the output look nice

# script completed
```

INPUT (As Command Line Arguments)**:**

```
3
23 17
```

OUTPUT:

```
Negation : -3

Sum        : 40
Difference : 6
Product    : 391
```

EXPLANATION:

In the first case we give only `3` as the command line argument. We can know the number of arguments using **`$#`** command. As we have provided only one argument, using **`if`** condition, we output only the negation of `3`. The value of the command line argument is accessed using **`$1`**.

In the second case we give two arguments `23` and `17`. As the number of arguments is two, using the **`if`** condition, we output the sum, difference and product of the numbers accessed by using **`$1`** and **`$2`**.

EXERCISE 4: CREATE A SHELL SCRIPT TO COPY YOUR SHELL SCRIPT FILE TO MULTIPLE LOCATIONS. THE COMMAND-LINE ARGUMENTS WILL BE THE ARBITRARY NUMBER OF LOCATIONS TO COPY THE FILE TO.

SHELL SCRIPT:

```
# Ex4 : Shell script to copy the file to multiple locations given as
command line arguments.

echo  # prints a blank line to make output look better

if [ $# -eq 0 ]       # if condition to notify user of no input
then

     echo " Warning : No arguments supplied"          # Warning
message

fi         # end of if

for i in $@           # accessing all the command line arguments

do

     mkdir -p $i           # making the specified directories.
                           # -p option does not give error if
directory already exits

     cp ex4.sh $i     # copies the shell script to specified
directories

     echo " Copied ex4.sh to directory : $i"          # outputs the
status

done  # loop completed

echo

# script completed
```

INPUT (As Command Line Arguments)**:**

```
 dir1 dir2 dir3
```

OUTPUT:

```
Copied ex4.sh to directory : dir1
Copied ex4.sh to directory : dir2
Copied ex4.sh to directory : dir3
```

EXPLANATION:

Here we provide directory names as command line arguments which are accessed using a for loop. **`$@`** command expands into a list of all the command line arguments. The loop variable **`‘i’`** assumes the values one by one.
The **`mkdir -p $i`** command is used to make the directory as specified in the **`‘i’`** variable. The **`-p`** option does not give an error if the specified directory already exists. Then after creating the directory the **`cp ex4.sh $i`** command copies the **`ex4.sh`** shell script into that directory.