**Experiment Number:3**

Problem Statement: **Shell scripting programs.**

NAME: **Manoj Dhanraj Mule**  ROLLNO: **71**

CLASS: IT-B BATCH: B3

DATE OF PERFORMANCE: 25-07-2024

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

PROGARM:

**Print Sum of Digits of a given number using command line argument**

**Program 1:**

code:

#!/bin/bash

if [ $# -ne 1 ]; then

echo "Usage: $0 <number>"

exit 1

fi

number=$1

sum=0

while [ $number -gt 0 ]; do

digit=$(( $number % 10 ))

sum=$(( $sum + $digit ))

number=$(( $number / 10 ))

done

echo "Sum of digits of $1 is: $sum"

**output:**

manoj@manoj-VirtualBox:~/shellscriptprogram$ ./sum.sh 123

Sum of digits of 123 is: 6

**2)Write a shell script using function for following:**

**1)average of given numbers**

**2) Max  digit from given number and**

**3) min digit  from given number**

**program 2:**

code:

#!/bin/bash

# Function to calculate the average of numbers

average() {

local total=0

local count=0

for num in "$@"; do

total=$((total + num))

count=$((count + 1))

done

if [ $count -gt 0 ]; then

echo "Average: $((total / count))"

else

echo "No numbers provided."

fi

}

# Function to find the maximum digit

max\_digit() {

local number=$1

local max=0

while [ $number -gt 0 ]; do

digit=$((number % 10))

if [ $digit -gt $max ]; then

max=$digit

fi

number=$((number / 10))

done

echo "Max digit: $max"

}

# Function to find the minimum digit

min\_digit() {

local number=$1

local min=9

while [ $number -gt 0 ]; do

digit=$((number % 10))

if [ $digit -lt $min ]; then

min=$digit

fi

number=$((number / 10))

done

echo "Min digit: $min"

}

# Main script

# Read input from the user

echo "Enter numbers separated by spaces:"

read -r -a numbers

if [ ${#numbers[@]} -eq 0 ]; then

echo "No numbers provided. Exiting."

exit 1

fi

# Call functions with the user input

average "${numbers[@]}"

max\_digit "${numbers[0]}"

min\_digit "${numbers[0]}"

**output:**

manoj@manoj-VirtualBox:~/shellscriptprogram$ ./function.sh

Enter numbers separated by spaces:

123 456 789

Average: 456

Max digit: 3

Min digit: 1

**3) Perform sorting on given array elements**

**program 3:**

#!/bin/bash

# Prompt user to enter numbers

echo "Enter numbers separated by spaces:"

read -r -a array

# Sort array in ascending order

sorted=($(printf '%s\n' "${array[@]}" | sort -n))

# Print sorted array

echo "Sorted array:"

echo "${sorted[@]}"

**output:**

manoj@manoj-VirtualBox:~/shellscriptprogram$ ./sort.sh

Enter numbers separated by spaces:

6 5 4 3 2

Sorted array:

2 3 4 5 6

**4) Program to find factorial of a given number with and without recursion**

**program 4:**

#!/bin/bash

# Factorial with recursion

factorial\_recursive() {

if [ $1 -eq 0 ]; then

echo 1

else

local prev=$(factorial\_recursive $(( $1 - 1 )))

echo $(( $1 \* $prev ))

fi

}

# Factorial without recursion

factorial\_non\_recursive() {

local number=$1

local result=1

while [ $number -gt 0 ]; do

result=$(( result \* number ))

number=$(( number - 1 ))

done

echo $result

}

# Main script

echo "Enter a number:"

read -r num

if ! [[ $num =~ ^[0-9]+$ ]]; then

echo "Please enter a valid positive integer."

exit 1

fi

echo "Factorial of $num (recursive): $(factorial\_recursive $num)"

echo "Factorial of $num (non-recursive): $(factorial\_non\_recursive $num)"

**output:**

manoj@manoj-VirtualBox:~/shellscriptprogram$ ./factorial.sh

Enter a number:

5

Factorial of 5 (recursive): 120

Factorial of 5 (non-recursive): 120

**5) Program to check file type and permission for a given file**

**Program**

**program 5:**

#!/bin/bash

# Function to check file type and permissions

check\_file() {

local file="$1"

if [ ! -e "$file" ]; then

echo "File $file does not exist."

else

echo "File type of $file:"

file "$file"

echo "Permissions of $file:"

ls -l "$file"

fi

}

# Main script

echo "Enter the file path:"

read -r file\_path

check\_file "$file\_path"

**output:**

manoj@manoj-VirtualBox:~/shellscriptprogram$ ./filetype.sh

Enter the file path:

/home/manoj/shellscriptprogram/sum.sh

File type of /home/manoj/shellscriptprogram/sum.sh:

/home/manoj/shellscriptprogram/sum.sh: Bourne-Again shell script, ASCII text executable

Permissions of /home/manoj/shellscriptprogram/sum.sh:

-rwxrwxr-x 1 manoj manoj 257 Aug 10 15:01 /home/manoj/shellscriptprogram/sum.sh

**6)Check entered string is palindrome or not?**

**program 6:**

#!/bin/bash

# Function to check if string is palindrome

is\_palindrome() {

local string="$1"

local reversed=$(echo "$string" | rev)

if [ "$string" = "$reversed" ]; then

echo "Entered string is a palindrome."

else

echo "Entered string is not a palindrome."

fi

}

# Main script

read -p "Enter a string: " input\_string

is\_palindrome "$input\_string"

**output:**

manoj@manoj-VirtualBox:~/shellscriptprogram$ ./palindrome.sh

Enter a string: aba

Entered string is a palindrome.

manoj@manoj-VirtualBox:~/shellscriptprogram$ ./palindrome.sh

Enter a string: abc

Entered string is not a palindrome.

manoj@manoj-VirtualBox:~/shellscriptprogram$

**7) File Handling Program**

**program 7:**

#!/bin/bash

# Function to perform read operation

perform\_read\_operation() {

echo "Contents of '$file\_path':"

cat "$file\_path"

}

# Function to perform write operation (overwrite)

perform\_write\_operation() {

echo "Enter content to write:"

read new\_content

echo "$new\_content" > "$file\_path"

echo "Write operation complete."

}

# Function to perform append operation

perform\_append\_operation() {

echo "Enter content to append:"

read appended\_content

echo "$appended\_content" >> "$file\_path"

echo "Append operation complete."

}

# Function to check and display file type

check\_file\_type() {

if [ -f "$file\_path" ]; then

echo "'$file\_path' is a regular file."

elif [ -d "$file\_path" ]; then

echo "'$file\_path' is a directory."

else

echo "'$file\_path' is neither a regular file nor a directory."

fi

}

# Main script starts here

echo "Enter the file name:"

read file\_path

# Check if file exists

if [ -e "$file\_path" ]; then

echo "File '$file\_path' exists."

# Display file type

check\_file\_type

# Menu for operations

echo "Select an operation:"

echo "1. Read from file"

echo "2. Write to file (overwrite)"

echo "3. Append to file"

read choice

case $choice in

1)

# Perform read operation

if [ -r "$file\_path" ]; then

perform\_read\_operation

else

echo "Read permission denied."

fi

;;

2)

# Perform write operation

if [ -w "$file\_path" ]; then

perform\_write\_operation

else

echo "Write permission denied."

fi

;;

3)

# Perform append operation

if [ -a "$file\_path" ]; then

perform\_append\_operation

else

echo "Append permission denied."

fi

;;

\*)

echo "Invalid choice. Exiting."

;;

esac

else

echo "File '$file\_path' does not exist."

fi

**output:**

manoj@manoj-VirtualBox:~/shellscriptprogram$ ./filehandle.sh

Enter the file name:

demo

File 'demo' exists.

'demo' is a regular file.

Select an operation:

1. Read from file

2. Write to file (overwrite)

3. Append to file

1

Contents of 'demo':

i am a third year student

manoj@manoj-VirtualBox:~/shellscriptprogram$ ./filehandle.sh

Enter the file name:

demo

File 'demo' exists.

'demo' is a regular file.

Select an operation:

1. Read from file

2. Write to file (overwrite)

3. Append to file

2

Enter content to write:

i am it student

Write operation complete.

manoj@manoj-VirtualBox:~/shellscriptprogram$ ./filehandle.sh

Enter the file name:

demo

File 'demo' exists.

'demo' is a regular file.

Select an operation:

1. Read from file

2. Write to file (overwrite)

3. Append to file

1

Contents of 'demo':

i am it student

manoj@manoj-VirtualBox:~/shellscriptprogram$

**All output :**



