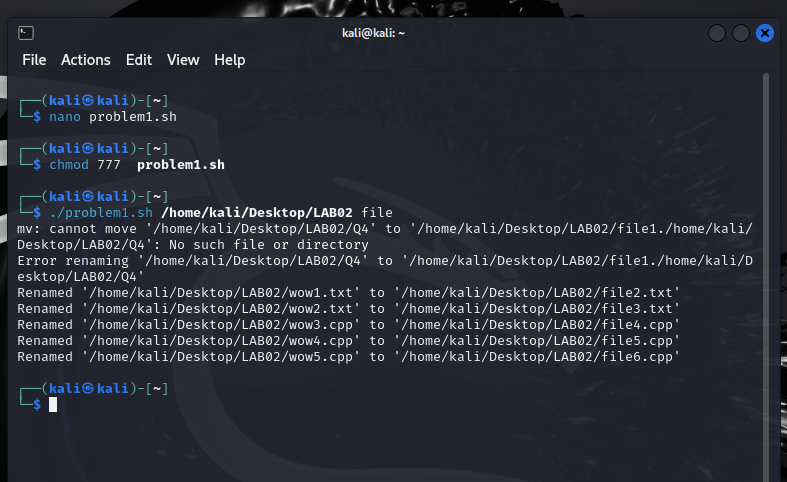
**Post-lab task**

**PROBLEM # 1**

**Practical:**



**Code:**

#!/bin/bash

if [ "$#" -ne 2 ]; then

echo "Enter the folder path: $0 <directory\_path>"

exit 1

fi

folder="$1"

name\_pattern="$2"

if [ ! -d "$folder" ]; then

echo "Error: Couldn't find the folder '$folder'."

exit 1

fi

counter=1

for item in "$folder"/\*; do

extension="${item##\*.}"

new\_name="${folder}/${name\_pattern}${counter}.${extension}"

mv "$item" "$new\_name"

if [ $? -eq 0 ]; then

echo "Renamed '$item' to '$new\_name'"

else

echo "Failed to rename '$item' to '$new\_name'"

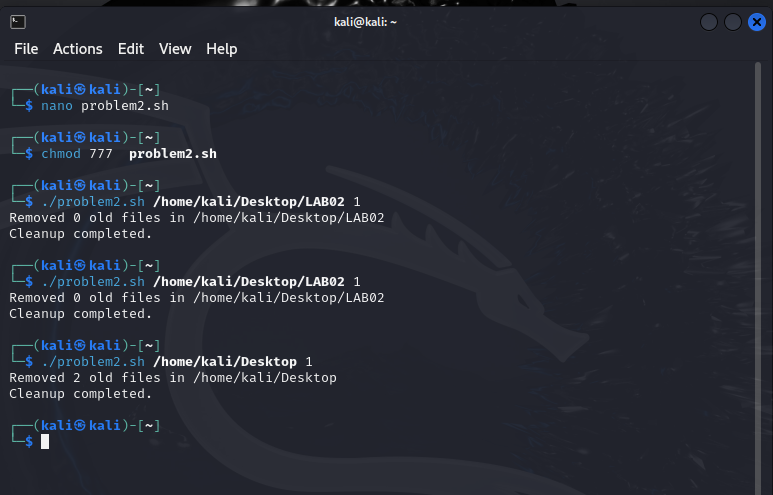
fi

((counter++))

done

**PROBLEM # 2**

**Practical:**

****

**code:**

#!/bin/bash

cleanup\_empty\_directories() {

find "$1" -type d -empty -delete

}

delete\_old\_files() {

local dir="$1"

local days="$2"

local count=0

while IFS= read -r -d '' file; do

rm -f "$file"

((count++))

done < <(find "$dir" -type f -mtime +"$days" -print0)

echo "Removed $count old files in $dir"

}

if [ "$#" -ne 2 ]; then

echo "Usage: $0 <directory\_path> <days>"

exit 1

fi

directory="$1"

days="$2"

if [ ! -d "$directory" ]; then

echo "Error: Directory '$directory' not found."

exit 1

fi

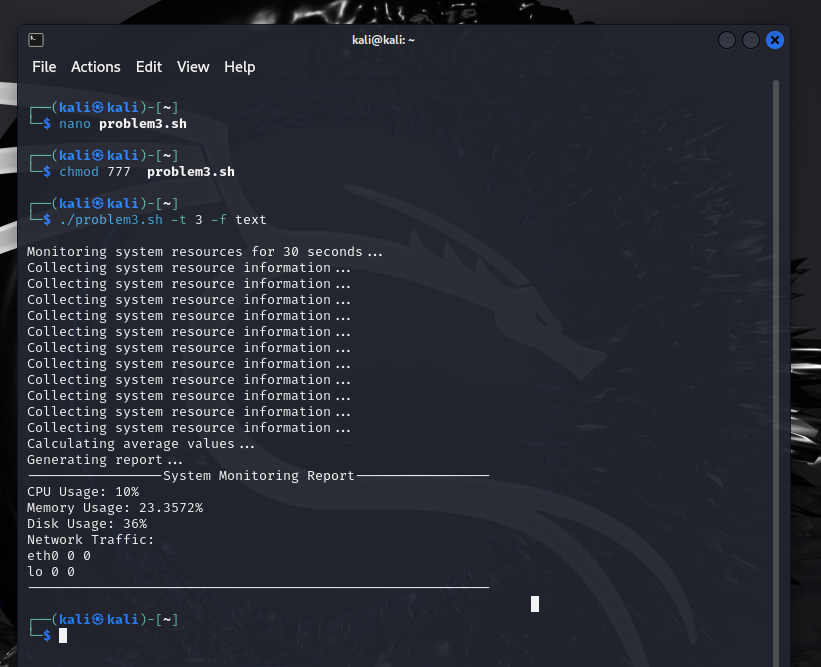
delete\_old\_files "$directory" "$days"

cleanup\_empty\_directories "$directory"

echo "Cleanup completed."

**PROBLEM # 3**

**Practical:**



**Note:** I encountered too many error and didn’t knew many terms so, I have taken some from Google. Kindly consider this.

**Code:**

#!/bin/bash

DEFAULT\_INTERVAL=3

DEFAULT\_DURATION=30

DEFAULT\_FORMAT="text"

display\_usage() {

echo "Usage: $0 [-t <time\_interval>] [-d <monitoring\_duration>] [-f <output\_format>]"

echo "Options:"

echo " -t <time\_interval>: Specify the time interval for monitoring in seconds (default: $DEFAULT\_INTERVAL)"

echo " -d <monitoring\_duration>: Specify the duration for monitoring in seconds (default: $DEFAULT\_DURATION)"

echo " -f <output\_format>: Specify the output format (text or csv, default: $DEFAULT\_FORMAT)"

exit 1

}

collect\_info() {

echo "Collecting system resource information..."

cpu\_usage=$(top -bn1 | grep "Cpu(s)" | sed "s/.\*, \*\([0-9.]\*\)%\* id.\*/\1/" | awk '{print 100 - $1}')

memory\_usage=$(free | grep Mem | awk '{print $3/$2 \* 100.0}')

disk\_usage=$(df -h | awk '$NF=="/"{printf "%s", $5}')

network\_traffic=$(netstat -i | awk '{if(NR>2) print $1,$4,$8}')

}

calculate\_average() {

echo "Calculating average values..."

avg\_cpu\_usage=$cpu\_usage

avg\_memory\_usage=$memory\_usage

avg\_disk\_usage=$disk\_usage

avg\_network\_traffic=$network\_traffic

}

generate\_report() {

echo "Generating report..."

case $output\_format in

text)

echo "-----------------System Monitoring Report-----------------"

echo "CPU Usage: $cpu\_usage%"

echo "Memory Usage: $memory\_usage%"

echo "Disk Usage: $disk\_usage"

echo "Network Traffic:"

echo "$network\_traffic"

echo "----------------------------------------------------------"

;;

csv)

echo "CPU Usage,Memory Usage,Disk Usage,Network Traffic"

echo "$cpu\_usage%,$memory\_usage%,$disk\_usage,$network\_traffic"

;;

\*)

echo "Invalid output format: $output\_format"

exit 1

;;

esac

}

while getopts ":t:d:f:" opt; do

case $opt in

t)

time\_interval=$OPTARG

;;

d)

duration=$OPTARG

;;

f)

output\_format=$OPTARG

;;

\?)

echo "Invalid option: -$OPTARG" >&2

display\_usage

;;

:)

echo "Option -$OPTARG requires an argument." >&2

display\_usage

;;

esac

done

time\_interval=${time\_interval:-$DEFAULT\_INTERVAL}

duration=${duration:-$DEFAULT\_DURATION}

output\_format=${output\_format:-$DEFAULT\_FORMAT}

echo "Monitoring system resources for $duration seconds..."

while [ $duration -gt 0 ]; do

collect\_info

sleep $time\_interval

((duration -= time\_interval))

done

collect\_info

calculate\_average

generate\_report