

## ASSIGNMENT:2

ADITYA MISHRA, BATCH-78 ,SAP.ID.590029219

Q1. Write a script that adds a user-defined prefix or suffix to all files in a directory.

### 1. Create the Script File

Open a terminal and create a new file: `nano rename_files.sh`

### 2. Paste the Following Bash Script

```
#!/bin/bash

# Ask for directory
read -p "Enter target directory: " directory

# Validate directory
if [[ ! -d "$directory" ]]; then
    echo "Error: Directory does not exist."
    exit 1
fi

# Ask for prefix and suffix
read -p "Enter prefix (leave blank for none): " prefix
read -p "Enter suffix (leave blank for none): " suffix

# Loop through files
for file in "$directory"/*; do
    # Skip if no files exist
    [[ -e "$file" ]] || continue

    # Skip directories
    if [[ -d "$file" ]]; then
        continue
    fi

    filename=$(basename "$file")
    name="${filename%.*}"      # file name without extension
    ext="${filename##*.}"       # extension

    # If file has no extension
    if [[ "$filename" == "$ext" ]]; then
        newname="${prefix}${name}${suffix}"
    else
        newname="${prefix}${name}${suffix}.${ext}"
    fi

    mv "$file" "$directory/$newname"
    echo "Renamed: $filename → $newname"
done
```

```
echo "Done!"
```

### 3. Save and Exit

In nano: CTRL + O (write) ENTER (confirm) CTRL + X (exit)

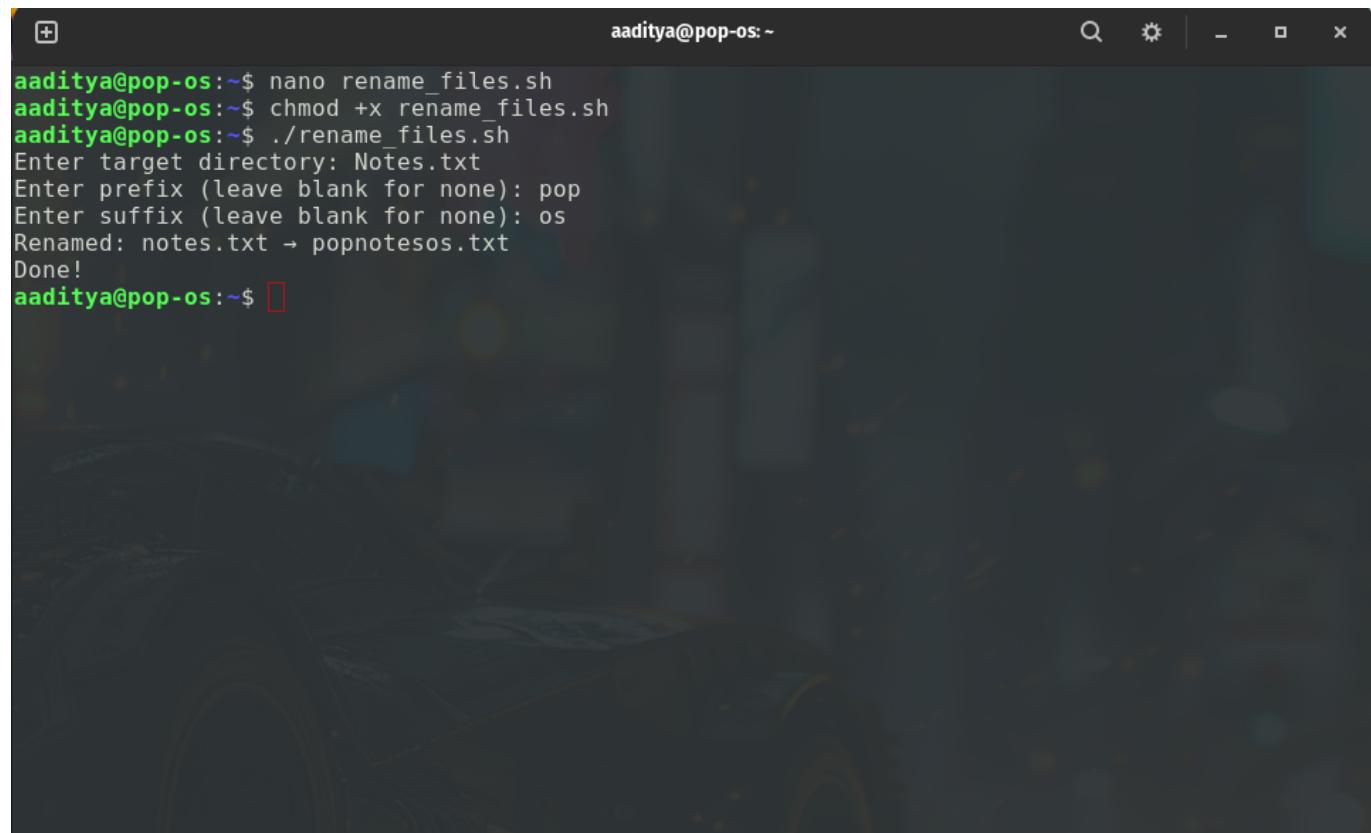
### 4. Make the Script Executable

```
chmod +x rename_files.sh
```

### 5. Run the Script

```
./rename_files.sh
```

Output:



The screenshot shows a terminal window with a dark background. The title bar says "aaditya@pop-os:~". The terminal output is as follows:

```
aaditya@pop-os:~$ nano rename_files.sh
aaditya@pop-os:~$ chmod +x rename_files.sh
aaditya@pop-os:~$ ./rename_files.sh
Enter target directory: Notes.txt
Enter prefix (leave blank for none): pop
Enter suffix (leave blank for none): os
Renamed: notes.txt → popnotesos.txt
Done!
aaditya@pop-os:~$
```

Q2. Search recursively for files with a given extension or larger than a specified size.

### 1. Overview

The find command is one of the most powerful tools in Linux for searching directories recursively.

You can use it to search by:

- File extension
- File size

- File type
- Name patterns
- And much more

## 2. Search for Files by Extension

Step 1 — Choose your starting directory

Example: search inside the current directory: `cd /path/to/start/directory` Step 2 — Run the find command Syntax `find <directory> -type f -name "*."`

## 3. Search for Files Larger Than a Certain Size

Step 1 — Pick your size unit

find supports:

k → kilobytes

M → megabytes

G → gigabytes

Step 2 — Use the -size flag Syntax `find <directory> -type f -size +<value><unit>`

## 4. Combine Extension AND Size

You can combine conditions using logical operators.

Find .`log` files larger than 5 MB `find . -type f -name "*." -size +5M`

Output:

```
aaditya@pop-os:~$ cd Downloads
aaditya@pop-os:~/Downloads$ find . -size +5M
find: './Downloads': No such file or directory
find: 'f': No such file or directory
aaditya@pop-os:~/Downloads$ find . -size +5M
find: './Downloads': No such file or directory
aaditya@pop-os:~/Downloads$ cd
aaditya@pop-os:~$ find . -size +5M
find: './Downloads': No such file or directory
aaditya@pop-os:~$ cd Downloads
aaditya@pop-os:~/Downloads$ find . -Lavanda-Dark -size +1M
find: './Lavanda-Dark': No such file or directory
aaditya@pop-os:~/Downloads$ find Lavanda-Dark -size +1M
aaditya@pop-os:~/Downloads$ find Lavanda-Dark -size +5M
aaditya@pop-os:~/Downloads$ find . -type f -name "*.log" -size +5M
aaditya@pop-os:~/Downloads$ find . -type f -name "*.mp4" -size +1G
aaditya@pop-os:~/Downloads$ cd
aaditya@pop-os:~$ find . -type f -name "*.mp4" -size +1G
aaditya@pop-os:~$ find . -type f -name "*.log" -size +5M
aaditya@pop-os:~$ find Lavanda-Dark -size +5M
find: 'Lavanda-Dark': No such file or directory
aaditya@pop-os:~$ cd Downloads
aaditya@pop-os:~/Downloads$ find . -type f -name "*.pdf"
./Assignment-1.pdf
./Assignment 2.pdf
aaditya@pop-os:~/Downloads$ find . -type f \(-name "*.jpg" -o -name "*.png" \)
./Orchis-Red/Orchis-Red/gtk-2.0/assets/scrollbar-vert-ltr-slider-disabled.png
./Orchis-Red/Orchis-Red/gtk-2.0/assets/treeview-ltr-button-hover.png
./Orchis-Red/Orchis-Red/gtk-2.0/assets/scale-horz-trough.png
./Orchis-Red/Orchis-Red/gtk-2.0/assets/scrollbar-vert-rtl-slider-active.png
./Orchis-Red/Orchis-Red/gtk-2.0/assets/scrollbar-vert-ltr-slider-active.png
./Orchis-Red/Orchis-Red/gtk-2.0/assets/progressbar-trough.png
./Orchis-Red/Orchis-Red/gtk-2.0/assets/combo-right-entry-disabled.png
./Orchis-Red/Orchis-Red/gtk-2.0/assets/entry.png
./Orchis-Red/Orchis-Red/gtk-2.0/assets/combo-right-entry.png
./Orchis-Red/Orchis-Red/gtk-2.0/assets/spin-rtl-up-disabled.png
./Orchis-Red/Orchis-Red/gtk-2.0/assets/pan-down-disabled.png
./Orchis-Red/Orchis-Red/gtk-2.0/assets/button.png
./Orchis-Red/Orchis-Red/gtk-2.0/assets/checkbox-unchecked-hover.png
./Orchis-Red/Orchis-Red/gtk-2.0/assets/pan-right-alt.png
./Orchis-Red/Orchis-Red/gtk-2.0/assets/pan-right-semi.png
./Orchis-Red/Orchis-Red/gtk-2.0/assets/scrollbar-vert-rtl-trough.png
./Orchis-Red/Orchis-Red/gtk-2.0/assets/flat-button-hover.png
./Orchis-Red/Orchis-Red/gtk-2.0/assets/menu-radio-unchecked-disabled.png
./Orchis-Red/Orchis-Red/gtk-2.0/assets/handle-vert.png
./Orchis-Red/Orchis-Red/gtk-2.0/assets/radio-checked-disabled.png
./Orchis-Red/Orchis-Red/gtk-2.0/assets/scale-horz-trough-disabled.png
./Orchis-Red/Orchis-Red/gtk-2.0/assets/scrollbar-horz-slider-hover.png
./Orchis-Red/Orchis-Red/gtk-2.0/assets/spin-ltr-up-active.png
./Orchis-Red/Orchis-Red/gtk-2.0/assets/border.png
./Orchis-Red/Orchis-Red/gtk-2.0/assets/scrollbar-vert-ltr-slider-hover.png
```

**Q3. Generate Fibonacci series up to a given number of terms.**

Example Output: Enter limit: 8 0 1 1 2 3 5 8 13

### 1. Create a New Bash Script

Open a terminal and create a script file `nano fibonacci.sh`

### 2. Paste the Following Script

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

```
# Read number of terms
read -p "Enter limit: " n

a=0
b=1

# Print Fibonacci series
for (( i=0; i<n; i++ ))
do
    echo -n "$a "
    fn=$((a + b))
    a=$b
    b=$fn
done

echo
```

### 3. Save and Exit

In nano: CTRL + O (save) ENTER CTRL + X (exit)

### 4. Make the Script Executable

```
chmod +x fibonacci.sh
```

### 5. Run the Script

```
./fibonacci.sh
```

Output:

```
aaditya@pop-os:~/Downloads$ nano fibonacci.sh
aaditya@pop-os:~/Downloads$ chmod +x fibonacci.sh
aaditya@pop-os:~/Downloads$ ./fibonacci.sh
Enter limit: 2
0 1
aaditya@pop-os:~/Downloads$
```

Q4.Check if a file is readable, writable, or executable by the user.

### 1. Create a Bash Script

Open a terminal and create a new script: `nano check_permissions.sh`

### 2. Paste the Following Script

```
#!/bin/bash

# Ask for the file name
read -p "Enter file name: " file
```

```
# Check if file exists
if [[ ! -e "$file" ]]; then
    echo "Error: File does not exist."
    exit 1
fi

# Check readability
if [[ -r "$file" ]]; then
    echo "Readable: YES"
else
    echo "Readable: NO"
fi

# Check writability
if [[ -w "$file" ]]; then
    echo "Writable: YES"
else
    echo "Writable: NO"
fi

# Check executability
if [[ -x "$file" ]]; then
    echo "Executable: YES"
else
    echo "Executable: NO"
fi
```

### 3. Save and Exit

In nano: CTRL + O ENTER CTRL + X

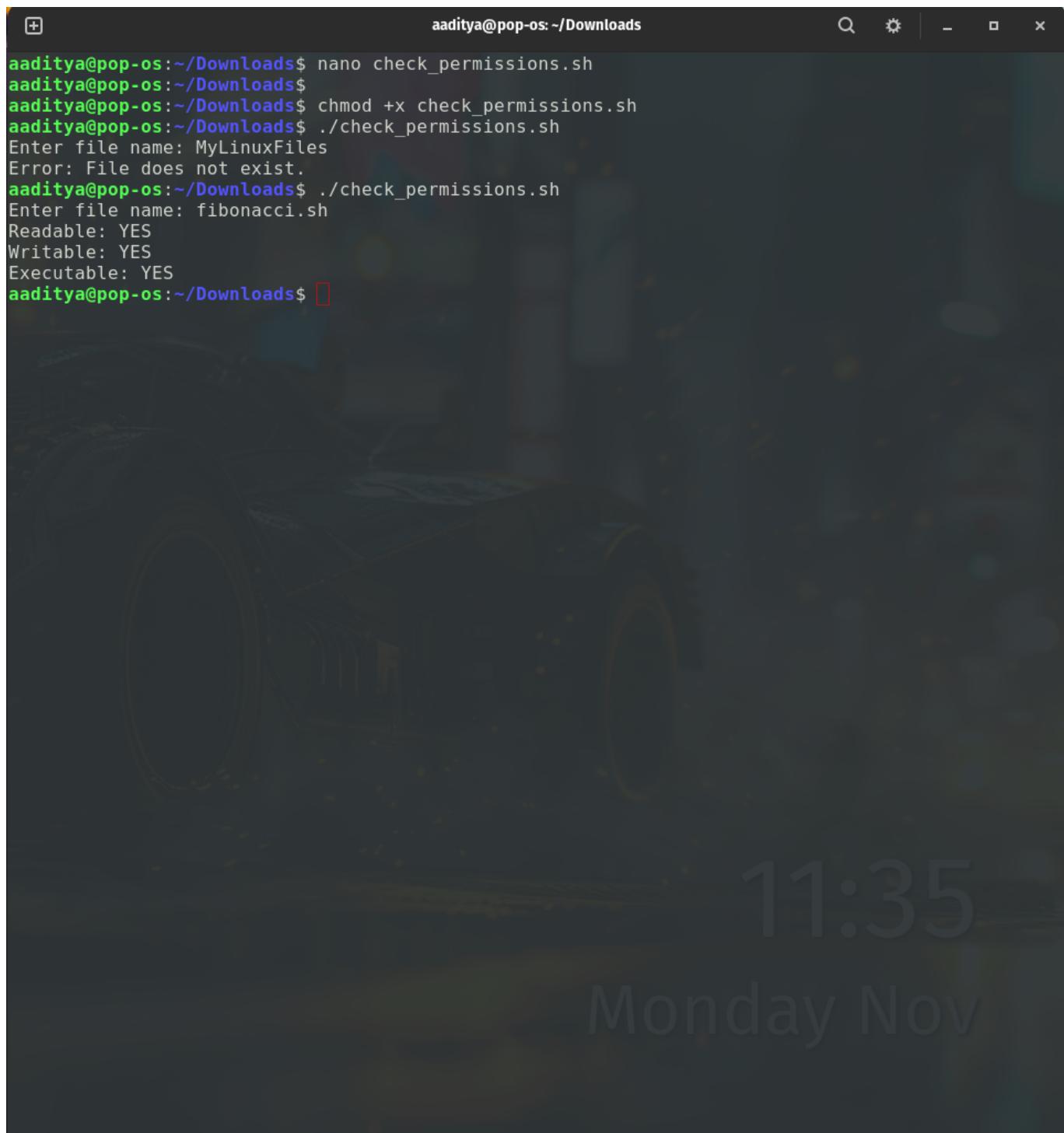
### 4. Make the Script Executable

```
chmod +x check_permissions.sh
```

### 5.5. Run the Script

```
./check_permissions.sh
```

Output:



The screenshot shows a terminal window titled "aaditya@pop-os: ~/Downloads". The user runs a script named "check\_permissions.sh" which prompts for a file name. When "MyLinuxFiles" is entered, it returns an error message. Then, when "fibonacci.sh" is entered, it displays its permissions: Readable: YES, Writable: YES, Executable: YES.

```
aaditya@pop-os:~/Downloads$ nano check_permissions.sh
aaditya@pop-os:~/Downloads$ chmod +x check_permissions.sh
aaditya@pop-os:~/Downloads$ ./check_permissions.sh
Enter file name: MyLinuxFiles
Error: File does not exist.
aaditya@pop-os:~/Downloads$ ./check_permissions.sh
Enter file name: fibonacci.sh
Readable: YES
Writable: YES
Executable: YES
aaditya@pop-os:~/Downloads$
```

Q5. Display system information (date, uptime, users, memory, disk usage)

## 1. Create the Script File

Open your terminal and create a new script: `nano system_info.sh`

## 2.2. Paste the Following Script

```
#!/bin/bash

echo "===== SYSTEM INFORMATION ====="
```

```
# Date and Time
echo -e "\n Date & Time:"
date

# System Uptime
echo -e "\n Uptime:"
uptime -p # human-readable uptime

# Logged-in Users
echo -e "\n Logged-in Users:"
who

# Memory Usage
echo -e "\n Memory Usage:"
free -h

# Disk Usage
echo -e "\n Disk Usage:"
df -h

echo -e "\n===== END ====="
```

### 3. Save and Exit

In nano: CTRL + O ENTER CTRL + X

### 4. Make the Script Executable

chmod +x system\_info.sh

### 5. Run the Script

./system\_info.sh

Output:

```
aaditya@pop-os:~/Downloads$ nano system_info.sh
aaditya@pop-os:~/Downloads$ chmod +x system_info.sh
aaditya@pop-os:~/Downloads$ ./system_info.sh
===== SYSTEM INFORMATION =====

⌚ Date & Time:
Mon Nov 24 11:43:31 AM IST 2025

⏳ Uptime:
up 1 day, 1 minute

👤 Logged-in Users:
aaditya :1           2025-11-23 11:42 (:1)

🧠 Memory Usage:
              total        used        free      shared  buff/cache   available
Mem:       15Gi       5.2Gi      3.4Gi     1.9Gi       6.7Gi      7.1Gi
Swap:      19Gi          0B      19Gi

💾 Disk Usage:
Filesystem  Size  Used  Avail Use% Mounted on
tmpfs       1.6G  2.1M  1.6G   1% /run
efivarfs    438K  346K  88K  80% /sys/firmware/efi/efivars
/dev/nvme0n1p3 460G  27G  410G   7% /
tmpfs       7.7G  92M  7.6G   2% /dev/shm
tmpfs       5.0M   0  5.0M   0% /run/lock
/dev/nvme0n1p2 4.0G  2.4G  1.6G  61% /recovery
/dev/nvme0n1p1 1020M 337M  684M  34% /boot/efi
tmpfs       1.6G  232K  1.6G   1% /run/user/1000

===== END =====
aaditya@pop-os:~/Downloads$
```

Q6. Continuously monitor and log top memory-consuming processes.

## 1. Create a Bash Script

Open a terminal and create a script: `nano monitor_memory.sh`

## 2. Paste the Following Script

```
#!/bin/bash

LOGFILE="memory_log.txt"
INTERVAL=5    # seconds between checks
```

```
TOP=10      # number of top processes to log

echo "Starting memory monitor..."
echo "Logging top $TOP memory-consuming processes every $INTERVAL seconds."
echo "Log file: $LOGFILE"
echo "-----" >> $LOGFILE

while true
do
    echo -e "\n===== $(date) =====" >> $LOGFILE
    ps aux --sort=-%mem | head -n $((TOP + 1)) >> $LOGFILE
    echo "Logged at $(date)"
    sleep $INTERVAL
done
```

### 3. Save and Exit

In nano: CTRL + O ENTER CTRL + X

### 4. Make the Script Executable

```
chmod +x monitor_memory.sh
```

### 5. Run the Script:

```
./monitor_memory.sh
```

Output:

```
aaditya@pop-os:~/Downloads$ cd
aaditya@pop-os:~$ nano monitor_memory.sh
aaditya@pop-os:~$ chmod +x monitor_memory.sh
aaditya@pop-os:~$ ./monitor_memory.sh
Starting memory monitor...
Logging top 10 memory-consuming processes every 5 seconds.
Log file: memory_log.txt
Logged at Mon Nov 24 12:01:35 PM IST 2025
Logged at Mon Nov 24 12:01:40 PM IST 2025
Logged at Mon Nov 24 12:01:45 PM IST 2025
Logged at Mon Nov 24 12:01:50 PM IST 2025
Logged at Mon Nov 24 12:01:55 PM IST 2025
Logged at Mon Nov 24 12:02:00 PM IST 2025
Logged at Mon Nov 24 12:02:05 PM IST 2025
Logged at Mon Nov 24 12:02:10 PM IST 2025
Logged at Mon Nov 24 12:02:15 PM IST 2025
Logged at Mon Nov 24 12:02:20 PM IST 2025
Logged at Mon Nov 24 12:02:25 PM IST 2025
Logged at Mon Nov 24 12:02:30 PM IST 2025
Logged at Mon Nov 24 12:02:35 PM IST 2025
^C
aaditya@pop-os:~$
```

Q7.Take a filename as input and display the number of lines, words, and characters.

## 1. Create the Script File

Open your terminal and create a script: `nano file_stats.sh`

## 2. Paste the Following Script

```
#!/bin/bash

# Ask for filename
read -p "Enter filename: " file
```

```
# Check if file exists
if [[ ! -f "$file" ]]; then
    echo "Error: File does not exist."
    exit 1
fi

# Count lines, words, and characters using wc
lines=$(wc -l < "$file")
words=$(wc -w < "$file")
chars=$(wc -m < "$file")

echo "File: $file"
echo "Lines: $lines"
echo "Words: $words"
echo "Characters: $chars"
```

### 3. Save and Exit

In nano: CTRL + O ENTER CTRL + X

### 4. Make the Script Executable

```
chmod +x file_stats.sh
```

### 5. Run the Script

```
./file_stats.sh
```

#### Output:

The screenshot shows a terminal window titled 'aaditya@pop-os: ~/MyLinuxFiles'. The terminal displays the following command-line session:

```
aaditya@pop-os:~$ ls
assignment1    exp9.sh          Music           Public
cpu_usage.log  file_stats.sh   MyLinuxFiles   rename_files.sh
Desktop        largest.sh      MyLinuxFiles.tar.gz rtw89
Documents       linux-lab-assingments Notes.txt     sum_natural.sh
Downloads      memory_log.txt 'Pankaj Sir Assingments' Templates
exp10.sh        monitor_memory.sh Pictures      Videos
aaditya@pop-os:~$ cd MyLinuxFiles
aaditya@pop-os:~/MyLinuxFiles$ ls
file1.txt  file2.txt  file3.txt  MyLinuxFiles.tar.gz  sorted.txt
aaditya@pop-os:~/MyLinuxFiles$ nano file_stats.sh
aaditya@pop-os:~/MyLinuxFiles$ chmod +x file_stats.sh
aaditya@pop-os:~/MyLinuxFiles$ ./file_stats.sh
file1.txt: file1.txt
File: file1.txt
Lines: 0
Words: 0
Characters: 0
aaditya@pop-os:~/MyLinuxFiles$
```

The desktop background features a blurred image of a car with the time '12:15' and date 'Monday Nov' overlaid.

Q8.Accept multiple numbers and sort them in ascending order.

### 1. Create a New Script

Open your terminal: `nano sort_numbers.sh`

### 2. Paste the Following Bash Script

```
#!/bin/bash

# Ask user for numbers
read -p "Enter numbers separated by spaces: " -a nums
```

```
# Sort the numbers
sorted=$(printf "%s\n" "${nums[@]}" | sort -n)

echo "Sorted numbers (ascending):"
echo "$sorted"
```

### 3. Save and Exit

In nano: CTRL + O ENTER CTRL + X

### 4. Make the Script Executable

```
chmod +x sort_numbers.sh
```

### 5. Run the Script

```
./sort_numbers.sh
```

Output:

```
aaditya@pop-os:~/MyLinuxFiles$ nano sort_numbers.sh
aaditya@pop-os:~/MyLinuxFiles$ chmod +x sort_numbers.sh
aaditya@pop-os:~/MyLinuxFiles$ ./sort_numbers.sh
Enter numbers separated by spaces: 5
Sorted numbers (ascending):
5
aaditya@pop-os:~/MyLinuxFiles$ ./sort_numbers.sh
Enter numbers separated by spaces: 1 2 3 4 5 6 7 8 9
Sorted numbers (ascending):
1
2
3
4
5
6
7
8
9
aaditya@pop-os:~/MyLinuxFiles$
```

Q9.Calculate the GCD and LCM of two given numbers.

### 1. Create a New Script

Open your terminal: `nano gcd_lcm.sh`

### 2. Paste the Following Bash Script

```
#!/bin/bash

# Ask for two numbers
read -p "Enter first number: " a
read -p "Enter second number: " b
```

```
# Validate input
if ! [[ "$a" =~ ^[0-9]+$ ]] || ! [[ "$b" =~ ^[0-9]+$ ]]; then
    echo "Error: Please enter valid positive integers."
    exit 1
fi

# Function to calculate GCD using Euclidean algorithm
gcd() {
    x=$1
    y=$2
    while [ $y -ne 0 ]; do
        r=$((x % y))
        x=$y
        y=$r
    done
    echo $x
}

# Function to calculate LCM
lcm() {
    x=$1
    y=$2
    # LCM = (a * b) / GCD
    echo $(( (x * y) / $(gcd $x $y) ))
}

GCD=$(gcd $a $b)
LCM=$(lcm $a $b)

echo "GCD of $a and $b is: $GCD"
echo "LCM of $a and $b is: $LCM"
```

### 3. Save and Exit

In nano: CTRL + O ENTER CTRL + X

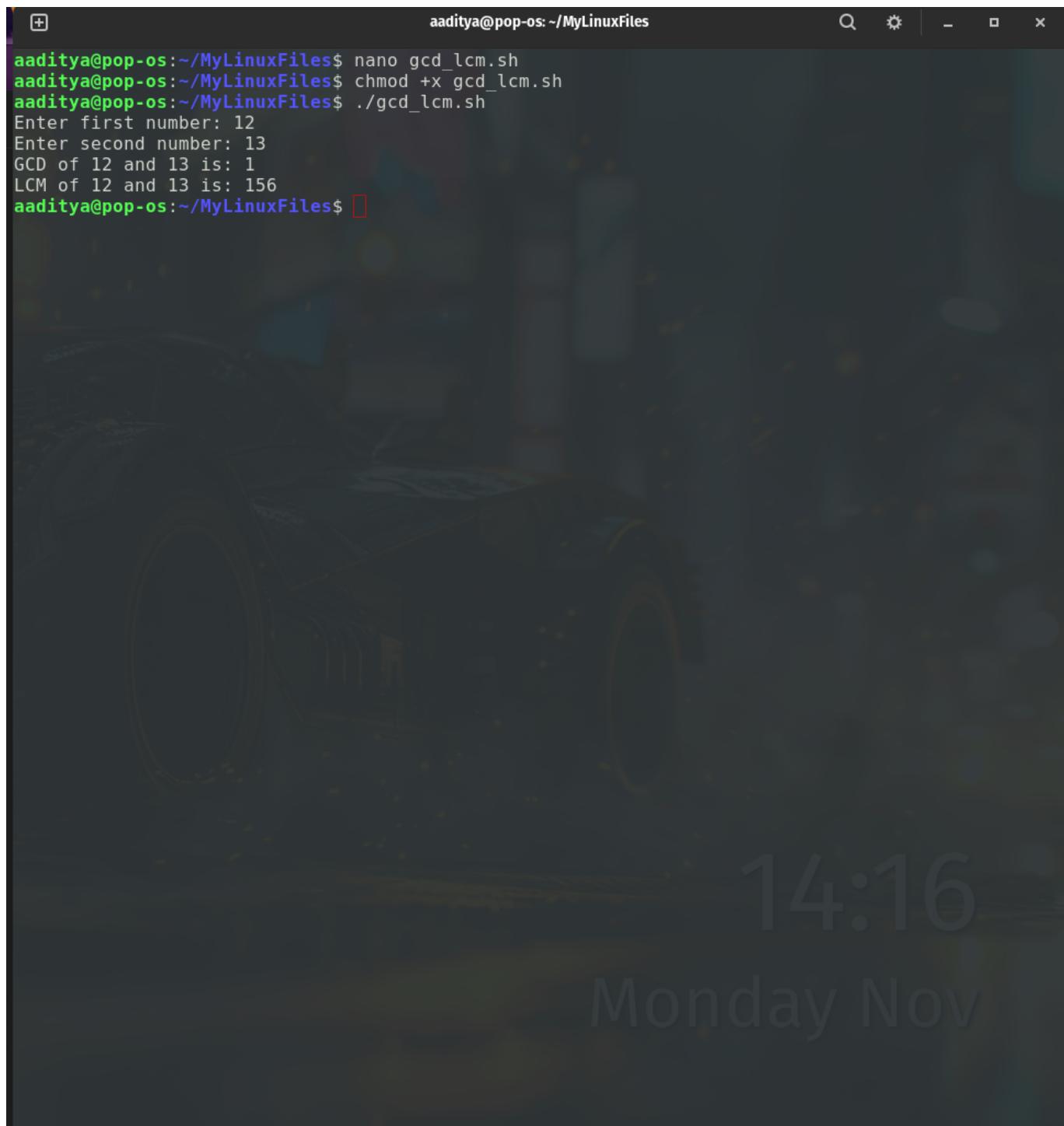
### 4. Make the Script Executable

```
chmod +x gcd_lcm.sh
```

### 5. Run the Script

```
./gcd_lcm.sh
```

Output:



```
aaditya@pop-os:~/MyLinuxFiles$ nano gcd_lcm.sh
aaditya@pop-os:~/MyLinuxFiles$ chmod +x gcd_lcm.sh
aaditya@pop-os:~/MyLinuxFiles$ ./gcd_lcm.sh
Enter first number: 12
Enter second number: 13
GCD of 12 and 13 is: 1
LCM of 12 and 13 is: 156
aaditya@pop-os:~/MyLinuxFiles$
```

Q10.Check whether an entered string is a palindrome or not.

### 1. Create a New Script

Open your terminal: `nano palindrome_check.sh`

### 2. Paste the Following Bash Script

```
#!/bin/bash

# Ask for input string
read -p "Enter a string: " str
```

```
# Remove spaces and convert to lowercase for uniformity
clean_str=$(echo "$str" | tr -d ' ' | tr '[:upper:]' '[:lower:]')

# Reverse the string
rev_str=$(echo "$clean_str" | rev)

# Check if palindrome
if [ "$clean_str" == "$rev_str" ]; then
    echo "\"$str\"" is a palindrome."
else
    echo "\"$str\"" is not a palindrome."
fi
```

### 3. Save and Exit

In nano: CTRL + O ENTER CTRL + X

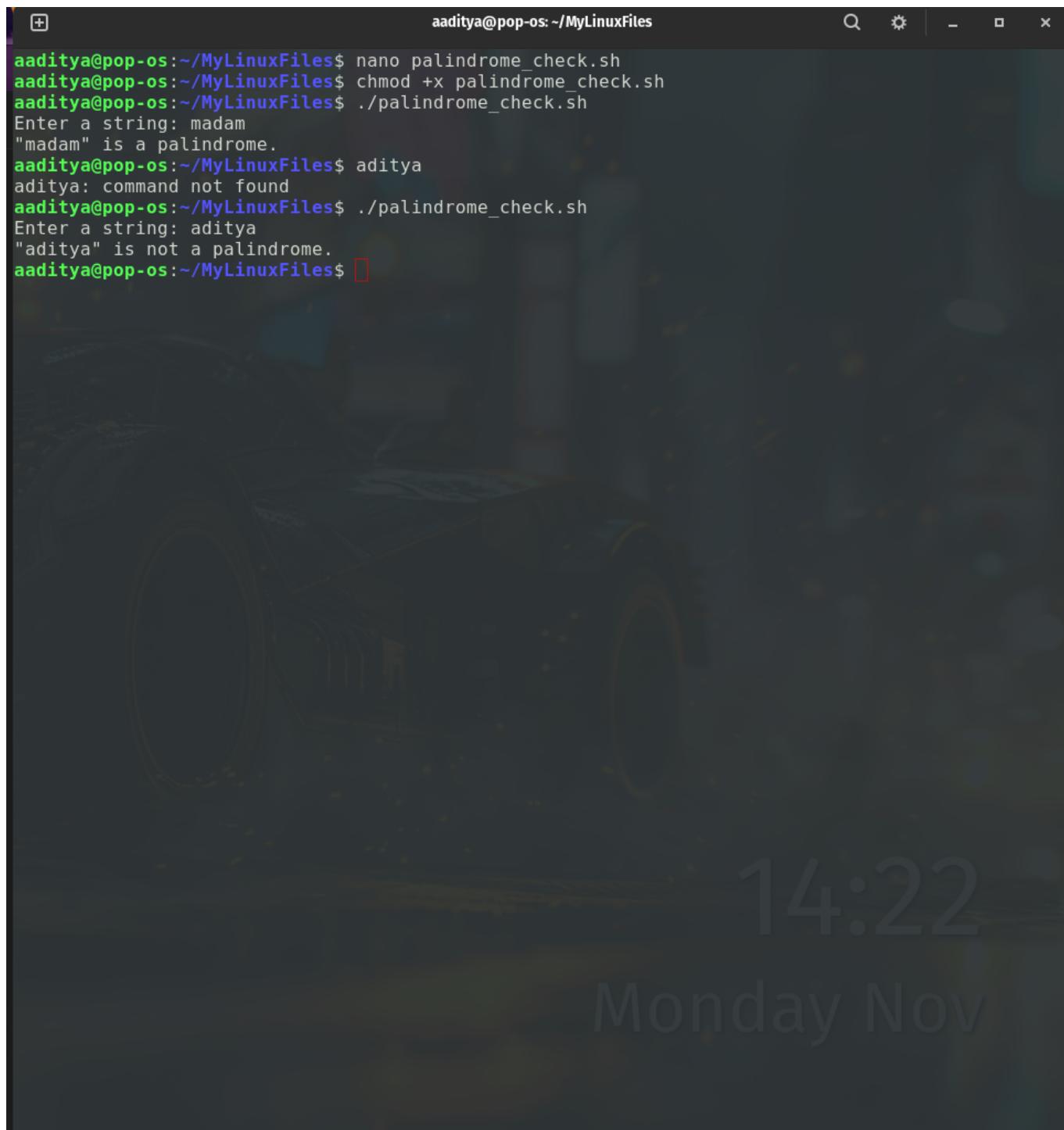
### 4. Make the Script Executable

chmod +x palindrome\_check.sh

### 5. Run the Script

./palindrome\_check.sh

Output:



The screenshot shows a terminal window titled "aaditya@pop-os: ~/MyLinuxFiles". The user has run a script named "palindrome\_check.sh". The script asks for a string input ("Enter a string:"). When the user types "madam", it outputs "madam" is a palindrome. However, when the user types "aditya", it outputs "aditya" is not a palindrome. The terminal window has a dark background with a blurred image of a car in the background.

```
aaditya@pop-os:~/MyLinuxFiles$ nano palindrome_check.sh
aaditya@pop-os:~/MyLinuxFiles$ chmod +x palindrome_check.sh
aaditya@pop-os:~/MyLinuxFiles$ ./palindrome_check.sh
Enter a string: madam
"madam" is a palindrome.
aaditya@pop-os:~/MyLinuxFiles$ aditya
aditya: command not found
aaditya@pop-os:~/MyLinuxFiles$ ./palindrome_check.sh
Enter a string: aditya
"aditya" is not a palindrome.
aaditya@pop-os:~/MyLinuxFiles$
```

Q11. Calculate and display the length of a string

### 1. Create a New Script

Open your terminal: `nano string_length.sh`

### 2. Paste the Following Bash Script

```
#!/bin/bash

# Ask for input string
read -p "Enter a string: " str
```

```
# Calculate length
length=${#str}

# Display result
echo "The length of the string is: $length"
```

### 3. Save and Exit

In nano: CTRL + O ENTER CTRL + X

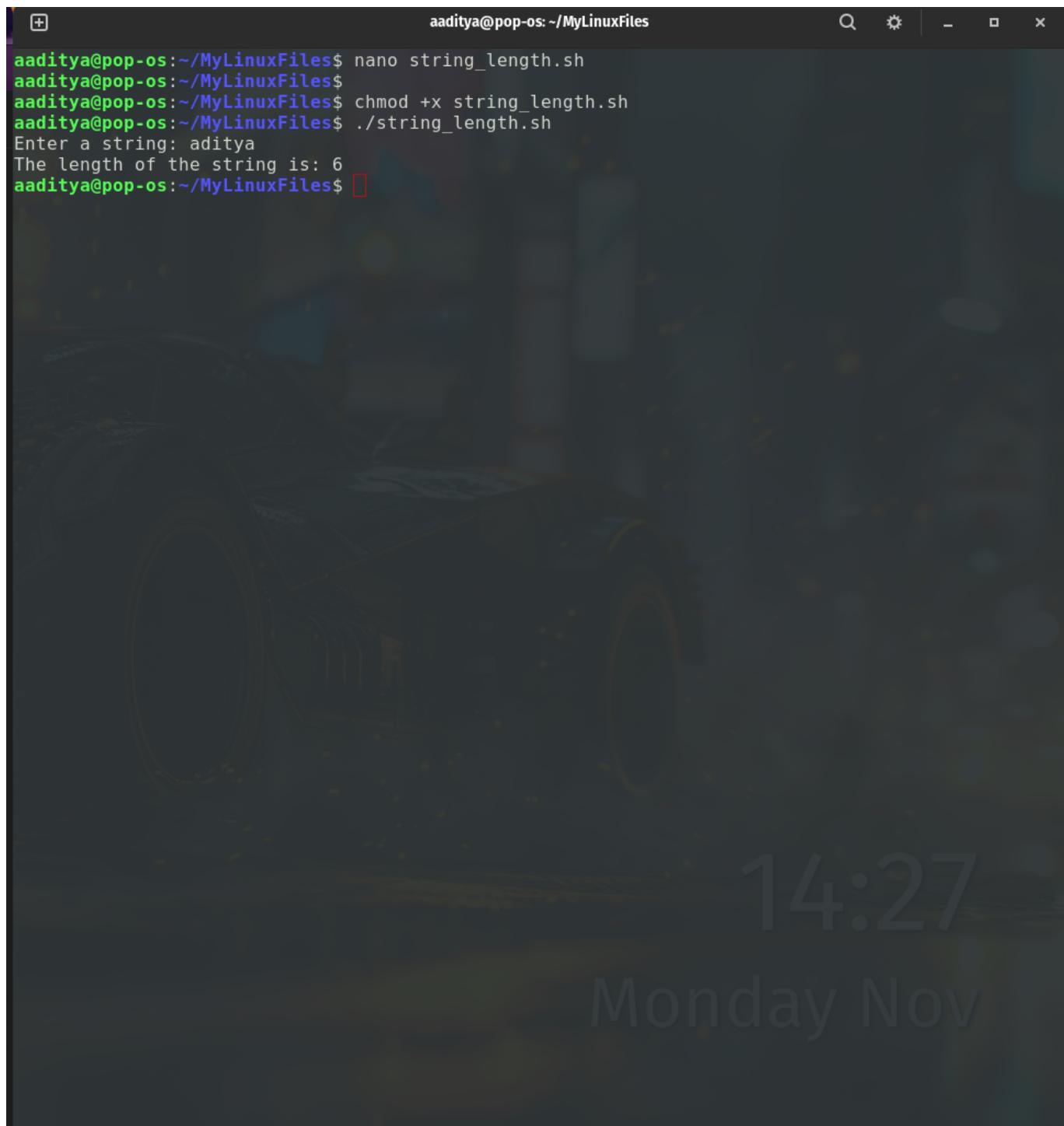
### 4. Make the Script Executable

```
chmod +x string_length.sh
```

### 5. Run the Script

```
./string_length.sh
```

Output:



```
aaditya@pop-os:~/MyLinuxFiles$ nano string_length.sh
aaditya@pop-os:~/MyLinuxFiles$ chmod +x string_length.sh
aaditya@pop-os:~/MyLinuxFiles$ ./string_length.sh
Enter a string: aditya
The length of the string is: 6
aaditya@pop-os:~/MyLinuxFiles$
```

## Q12. Reverse a given string

### 1. Create a New Script

Open your terminal: `nano reverse_string.sh`

### 2. Paste the Following Bash Script

```
#!/bin/bash

# Ask for input string
read -p "Enter a string: " str
```

```
# Reverse the string  
rev_str=$(echo "$str" | rev)  
  
# Display the reversed string  
echo "Reversed string: $rev_str"
```

### 3. Save and Exit

In nano: CTRL + O ENTER CTRL + X

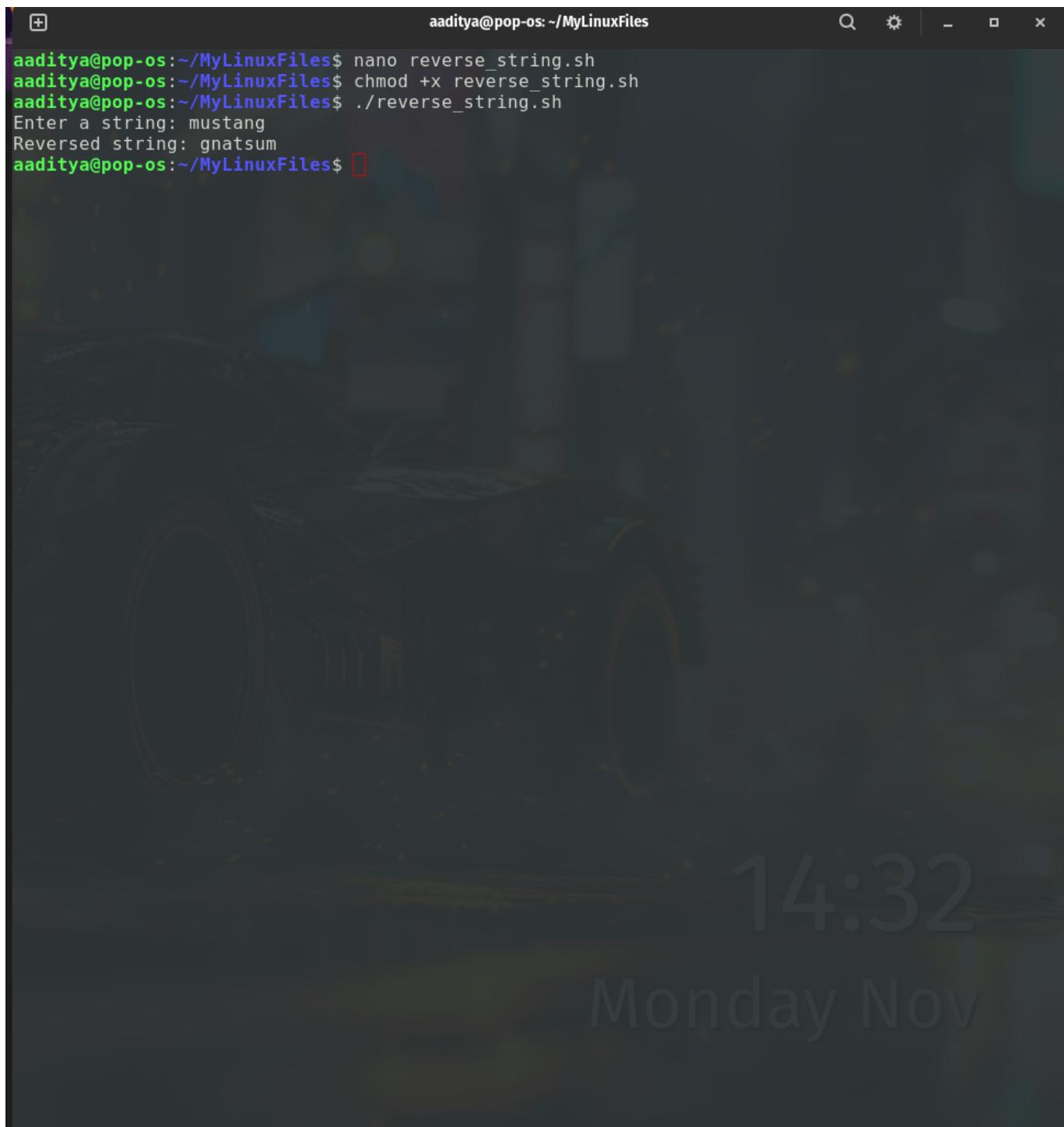
### 4. Make the Script Executable

```
chmod +x reverse_string.sh
```

### 5. Run the Script

```
./reverse_string.sh
```

Output:



The screenshot shows a terminal window titled "aaditya@pop-os: ~/MyLinuxFiles". The terminal displays the following command-line session:

```
aaditya@pop-os:~/MyLinuxFiles$ nano reverse_string.sh
aaditya@pop-os:~/MyLinuxFiles$ chmod +x reverse_string.sh
aaditya@pop-os:~/MyLinuxFiles$ ./reverse_string.sh
Enter a string: mustang
Reversed string: gnatsum
aaditya@pop-os:~/MyLinuxFiles$
```

Q13.Concatenate two input strings.

### 1. Create a New Script

Open your terminal: `nano concatenate_strings.sh`

### 2. Paste the Following Bash Script

```
#!/bin/bash

# Ask for first string
read -p "Enter first string: " str1
```

```
# Ask for second string
read -p "Enter second string: " str2

# Concatenate strings
concat="$str1$str2"

# Display result
echo "Concatenated string: $concat"
```

### 3. Save and Exit

In nano: CTRL + O ENTER CTRL + X

### 4. Make the Script Executable

```
chmod +x concatenate_strings.sh
```

### 5. Run the Script

```
./concatenate_strings.sh
```

Output:

```
[+] aaditya@pop-os: ~/MyLinuxFiles
aaditya@pop-os:~/MyLinuxFiles$ nano concatenate_strings.sh
aaditya@pop-os:~/MyLinuxFiles$ chmod +x concatenate_strings.sh
aaditya@pop-os:~/MyLinuxFiles$ ./concatenate_strings.sh
Enter first string: hello
Enter second string: jarvis
Concatenated string: hellojarvis
aaditya@pop-os:~/MyLinuxFiles$
```

14:36

Monday Nov