**1. Automate the creation of a new user with specific permissions and home directory**

#!/bin/bash

# Check if the script is run as root

if [ "$EUID" -ne 0 ]; then

echo "Please run as root."

exit 1

fi

# Create a new user with a specific home directory and shell

read -p "Enter the username: " USERNAME

read -p "Enter the home directory (e.g., /home/$USERNAME): " HOMEDIR

useradd -m -d "$HOMEDIR" -s /bin/bash "$USERNAME"

# Set user password

passwd "$USERNAME"

# Set specific permissions for the user (e.g., add to sudoers for sudo permissions)

usermod -aG sudo "$USERNAME"

echo "User $USERNAME created with home directory $HOMEDIR and sudo permissions."

**2. Shell script to find all large files greater than 1 GB in a directory and move them to another directory**

#!/bin/bash

# Check if sufficient arguments are provided

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

echo "Usage: $0 <source\_directory> <destination\_directory>"

exit 1

fi

SOURCE\_DIR=$1

DEST\_DIR=$2

# Find and move files larger than 1GB

find "$SOURCE\_DIR" -type f -size +1G -exec mv {} "$DEST\_DIR" \;

echo "All files larger than 1GB have been moved from $SOURCE\_DIR to $DEST\_DIR."

**3. Script to automatically update all installed packages on a system and reboot if needed**

#!/bin/bash

# Update all packages

sudo apt-get update && sudo apt-get upgrade -y

# Check if a reboot is required

if [ -f /var/run/reboot-required ]; then

echo "Reboot required. System will reboot in 1 minute."

sudo shutdown -r +1

else

echo "No reboot required."

fi

**4. Script to count the number of lines in all .log files in a specified directory**

#!/bin/bash

# Check if directory is provided

if [ -z "$1" ]; then

echo "Usage: $0 <directory>"

exit 1

fi

LOG\_DIR=$1

# Count lines in all .log files

find "$LOG\_DIR" -type f -name "\*.log" -exec wc -l {} \;

**5. Script that checks for the presence of specific software (e.g., Docker or Git) and installs if missing**

#!/bin/bash

# Function to check and install software

install\_if\_missing() {

if ! command -v "$1" &> /dev/null; then

echo "$1 is not installed. Installing..."

sudo apt-get install -y "$1"

else

echo "$1 is already installed."

fi

}

# Check for Docker and Git

install\_if\_missing docker

install\_if\_missing git

**6. Additional Shell Script: Check Free Memory and Alert (From earlier)**

#!/bin/bash

THRESHOLD=10

TOTAL\_MEM=$(free | grep Mem | awk '{print $2}')

FREE\_MEM=$(free | grep Mem | awk '{print $7}')

FREE\_PERCENT=$(( FREE\_MEM \* 100 / TOTAL\_MEM ))

if [ "$FREE\_PERCENT" -lt "$THRESHOLD" ]; then

echo "Warning: Free memory is below ${THRESHOLD}%! Only ${FREE\_PERCENT}% available."

else

echo "Memory is fine. Free memory: ${FREE\_PERCENT}%"

fi

These scripts automate system administration tasks, providing useful checks, package management, and file operations.

Here’s a shell script that checks the available free memory on the system and sends an alert if it falls below a 10% threshold:

#!/bin/bash

# Define the memory threshold percentage

THRESHOLD=10

# Get total and free memory values (in KB) using `free` command

TOTAL\_MEM=$(free | grep Mem | awk '{print $2}')

FREE\_MEM=$(free | grep Mem | awk '{print $7}')

# Calculate free memory percentage

FREE\_PERCENT=$(( FREE\_MEM \* 100 / TOTAL\_MEM ))

# Alert if free memory falls below the threshold

if [ "$FREE\_PERCENT" -lt "$THRESHOLD" ]; then

echo "Warning: Free memory is below ${THRESHOLD}%! Only ${FREE\_PERCENT}% available."

else

echo "Memory is fine. Free memory: ${FREE\_PERCENT}%"

fi

**How the script works:**

1. **THRESHOLD**: Sets the alert threshold to 10% (you can change it as needed).
2. **free command**: Retrieves memory usage data.
3. **Calculations**: It calculates the percentage of free memory available.
4. **Conditional check**: If the available free memory is less than 10%, it will display an alert.

You can make this script executable by running:

chmod +x check\_memory.sh

Run it manually or schedule it with cron to check memory periodically.