Write a shell scripting for below Questions

1)To list down which services are running in my system

list_services.sh

```
#!/bin/bash

echo "Listing all running services on Linux system using systemctl:"

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

# Check if systemctl is available

if command -v systemctl &> /dev/null; then

   # List running services

   systemctl list-units --type=service --state=running

else

   echo "systemctl is not available on this system."

fi
```

Make it executable

chmod +x list_services.sh

Run the script with the process name as an argument

./kill_process.sh list_services.sh

2)Need to kill one process which is running in my system

kill_process.sh

```
#!/bin/bash

# Check if the process name was provided as an argument

if [ $# -eq 0 ]; then

   echo "Usage: $0 <process_name>"

   exit 1

fi

# Get the process name from the argument

process_name=$1

# Find the process ID (PID) of the process

pid=$(pgrep -f "$process_name")

# Check if the process is running

if [ -z "$pid" ]; then
```

```
    echo "Process '$process_name' not found."
    exit 1
fi
# Kill the process
kill $pid
# Check if the kill command was successful
if [ $? -eq 0 ]; then
    echo "Process '$process_name' with PID $pid has been killed."
else
    echo "Failed to kill process '$process_name'."
    exit 1
fi
```

Make it executable

```
chmod +x kill_process.sh
```

Run the script with the process name as an argument

```
./kill_process.sh process_name
```

3)Need to get the disk space and memory space of the system

system_info.sh

```
#!/bin/bash
# Get disk space
echo "Disk Space:"
df -h
# Get memory space
echo "Memory Space:"
free -h
```

To run this script:

1. **Make the script executable**: chmod +x system_info.sh.
2. **Run the script**: ./system_info.sh.

4)List down software's which are installed in my system

Ubuntu:

```
list_installed_software.sh

#!/bin/bash

dpkg --get-selections
```

Make it executable

chmod +x list_installed_software.sh

Run the script

./list_installed_software.sh

CentOS:

```
list_installed_software.sh

#!/bin/bash

rpm -qa
```

Make it executable

chmod +x list_installed_software.sh

Run the script

./list_installed_software.sh

5)To get the service name and stop and start the service like (HTTPD & Nginx & Apache & Docker)

manage_service.sh

```
#!/bin/bash

# Function to check the status of a service

check_status() {

   sudo systemctl is-active --quiet $1 && echo "$1 is running" || echo "$1 is not running"

}

# Function to start a service

start_service() {

   sudo systemctl start $1

   echo "$1 started"

}

# Function to stop a service
```

```
stop_service() {
  sudo systemctl stop $1
  echo "$1 stopped"
}
# Check if the user provided enough arguments
if [ $# -lt 2 ]; then
  echo "Usage: $0 {start|stop|status} {httpd|nginx|apache2|docker}"
  exit 1
fi
# Assign arguments to variables
ACTION=$1
SERVICE=$2
# Perform the action based on the user input
case $ACTION in
  start)
    start_service $SERVICE
    ;;
  stop)
    stop_service $SERVICE
    ;;
  status)
    check_status $SERVICE
    ;;
  *)
    echo "Invalid action. Usage: $0 {start|stop|status} {httpd|nginx|apache2|docker}"
    exit 1
    ;;
esac
```

Make the script executable

```
chmod +x manage_service.sh
```

Run the script

```
./manage_service.sh start nginx

./manage_service.sh stop apache2

./manage_service.sh status docker
```

6)To list down the agent and if agent is stopped state, then start the service and check for every time if its stop script must start the service

manage_agents.sh

```
#!/bin/bash

# Function to start the agent service

start_service() {

   local service_name=$1

   echo "Starting $service_name..."

   sudo systemctl start $service_name

   if [ $? -eq 0 ]; then

      echo "$service_name started successfully."

   else

      echo "Failed to start $service_name."

   fi

}

# Function to check the status of the agent service

check_and_start_service() {

   local service_name=$1

   status=$(sudo systemctl is-active $service_name)

   if [ "$status" == "inactive" ] || [ "$status" == "failed" ]; then

      echo "$service_name is in $status state."

      start_service $service_name

   else

      echo "$service_name is running."

   fi

}

# List of agent services
```

```
agent_services=("agent1" "agent2" "agent3")  # Replace with actual agent service names
# Iterate through each agent service and check its status
for service in "${agent_services[@]}"; do
    check_and_start_service $service
done
```

Make the script executable

```
chmod +x manage_agents.sh
```

Run the script

```
./manage_agents.sh
```

Running the Script Periodically

```
crontab -e
```

To run the script every 5 minutes

```
*/5 * * * * /path/to/manage_agents.sh
```

7)To check the password expiry of the list of created users and if the password is expiring in 3 days, then update the password age for next 15 days

check_password_expiry.sh

```
#!/bin/bash
# List of users to check
users=("user1" "user2" "user3")
# Function to update password age
update_password_age() {
  local user=$1
  echo "Updating password age for user: $user"
  # Set password to expire in 15 days from now
  chage -d $(date +%Y-%m-%d) -M 15 $user
}
# Get the current date in seconds
current_date=$(date +%s)
# Loop through each user
for user in "${users[@]}"; do
  # Get the password expiry date
  expiry_date=$(chage -l $user | grep "Password expires" | cut -d: -f2 | xargs -I{} date -d {} +%s)

  # Calculate the number of days until expiry

  days_until_expiry=$(( (expiry_date - current_date) / 86400 ))

  # Check if the password expires in 3 days or less

  if [ $days_until_expiry -le 3 ]; then

    echo "Password for user $user is expiring in $days_until_expiry days."

    update_password_age $user

  else

    echo "Password for user $user is not expiring soon."

  fi

done
```

Make the script executable: chmod +x check_password_expiry.sh.

Run the script: ./check_password_expiry.sh.

8)To check the particular mount point if it's reached the 70% utilization then move zip the file which is older than 7 days and move those files into /tmp/ directory.

script.sh

```
#!/bin/bash

# Variables

MOUNT_POINT="/your/mount/point"  # Replace with your actual mount point

TARGET_DIR="/your/target/directory"  # Directory to check for old files

TMP_DIR="/tmp"

# Check disk usage

usage=$(df -h | grep "$MOUNT_POINT" | awk '{print $5}' | sed 's/%//g')

# Check if usage is greater than or equal to 70%

if [ "$usage" -ge 70 ]; then

    echo "Disk usage at $MOUNT_POINT is $usage%, which is above the threshold."

    # Find files older than 7 days and zip them

    find "$TARGET_DIR" -type f -mtime +7 -print0 | while IFS= read -r -d '' file; do

        zip_file="${file}.zip"

        zip "$zip_file" "$file"
        mv "$zip_file" "$TMP_DIR/"
    done
else
    echo "Disk usage at $MOUNT_POINT is $usage%, which is below the threshold."
fi
```

Make the script executable: chmod +x script.sh.

Run the script: ./ script.sh.

9)If the particular mount point is reached the 70 % then delete the older files starting 7 days of files

cleanup.sh

```
#!/bin/bash
# Mount point to check
MOUNT_POINT="/path/to/mount"
# Threshold percentage (70%)
THRESHOLD=70
# Directory to clean up
DIR_TO_CLEAN="/path/to/directory"
# Check the disk usage
USAGE=$(df -h "$MOUNT_POINT" | grep -vE '^Filesystem|tmpfs|cdrom' | awk '{ print $5 }' | sed 's/%//g')
# If usage is greater than or equal to the threshold
if [ "$USAGE" -ge "$THRESHOLD" ]; then
  echo "Disk usage is $USAGE%, which is greater than or equal to the threshold of $THRESHOLD%."
  echo "Deleting files older than 7 days in $DIR_TO_CLEAN..."
  # Find and delete files older than 7 days
  find "$DIR_TO_CLEAN" -type f -mtime +7 -exec rm -f {} \;
  echo "Old files deleted."
else
  echo "Disk usage is $USAGE%, which is below the threshold of $THRESHOLD%."
fi
```

Make the script executable: chmod +x cleanup.sh

Run the script: ./ cleanup.sh

10)Every day in the morning at 9am IST and Evening 7 PM IST I need to check the disk space and free memory and need to run the script and make a cron job to it and store the output in /tmp directory as diskspace.txt and process.txt

check_system.sh

```
#!/bin/bash

# Define the output files

DISKSPACE_FILE="/tmp/diskspace.txt"

PROCESS_FILE="/tmp/process.txt"

# Get disk space usage and free memory

df -h > "$DISKSPACE_FILE"

free -h > "$PROCESS_FILE"
```

Make the script executable: chmod +x check_system.sh

Run the script: ./ check_system.sh

Set Up the Cron Job:

crontab -e

To run the script at 9 AM and 7 PM IST

```
0 3 * * * /path/to/check_system.sh

0 13 * * * /path/to/check_system.sh
```