Experiment 7: Shell Programming, Process and Scheduling

Name: Aditya Mishra Roll No.: 590029219 Date: 2025-09-23

Aim:

- To write shell scripts that demonstrate process management.
- To understand how to schedule processes using `cron` and `at`.
- To monitor running processes and practice job control commands.

Requirements

- A Linux machine with bash shell.
- Access to process management commands (`ps`, `top`, `kill`, `jobs`, `fg`, `bg`).
- Access to scheduling utilities (`cron`, `at`).

Theory

Every program running in Linux is a process identified by a unique process ID (PID). Shell programming allows automation of tasks including spawning and controlling processes. Process management commands like `ps`, `top`, `kill`, `jobs`, `bg`, and `fg` let users monitor and control execution. Scheduling utilities such as `cron` (repeated tasks) and `at` (one-time tasks) allow tasks to run automatically at defined times. Combining scripting with scheduling is a core system administration skill.

Procedure & Observations

Task 1

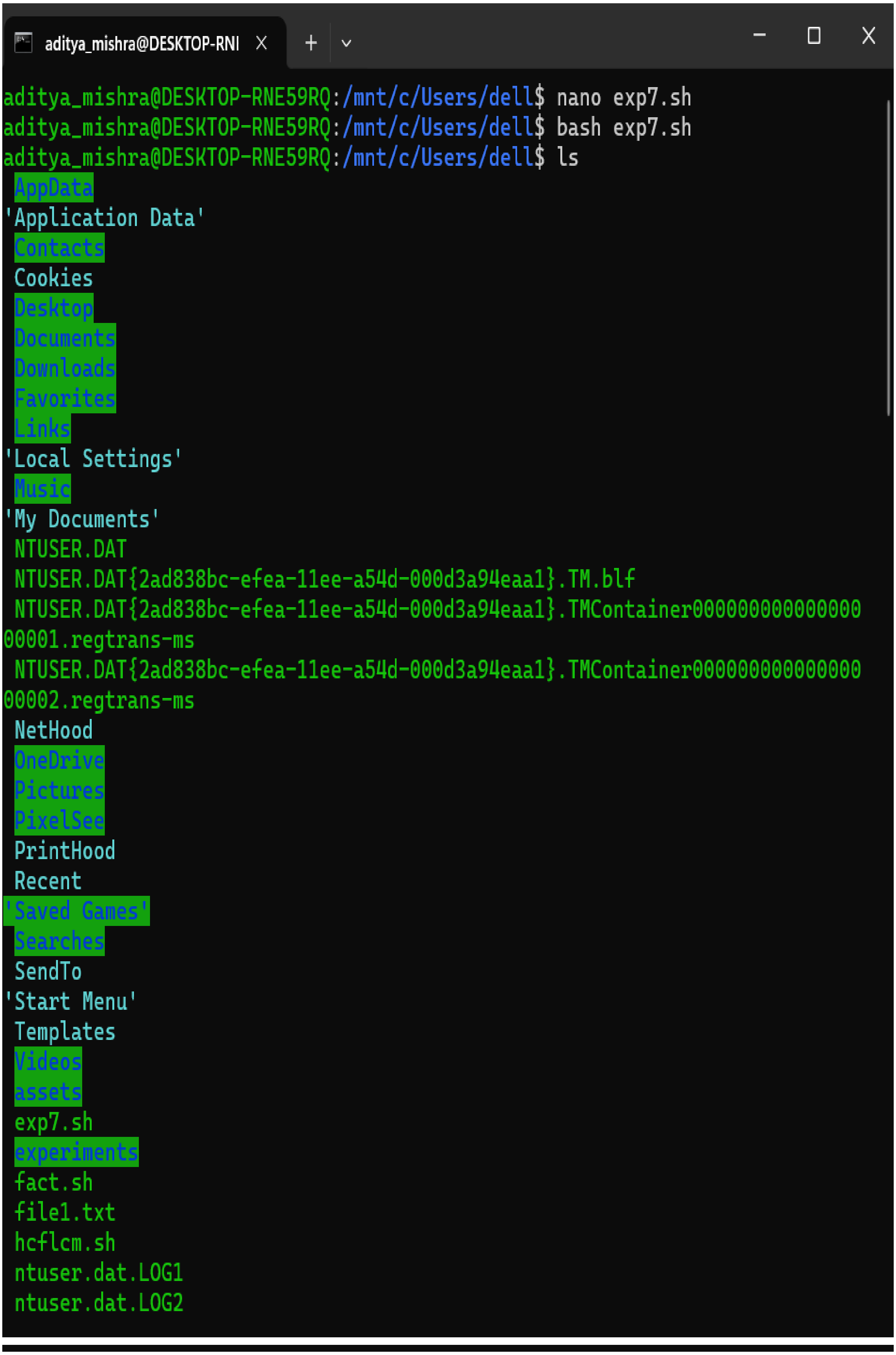
Task Statement:

Write a script that monitors the top 5 processes consuming the most CPU and logs them into a file every 10 seconds.

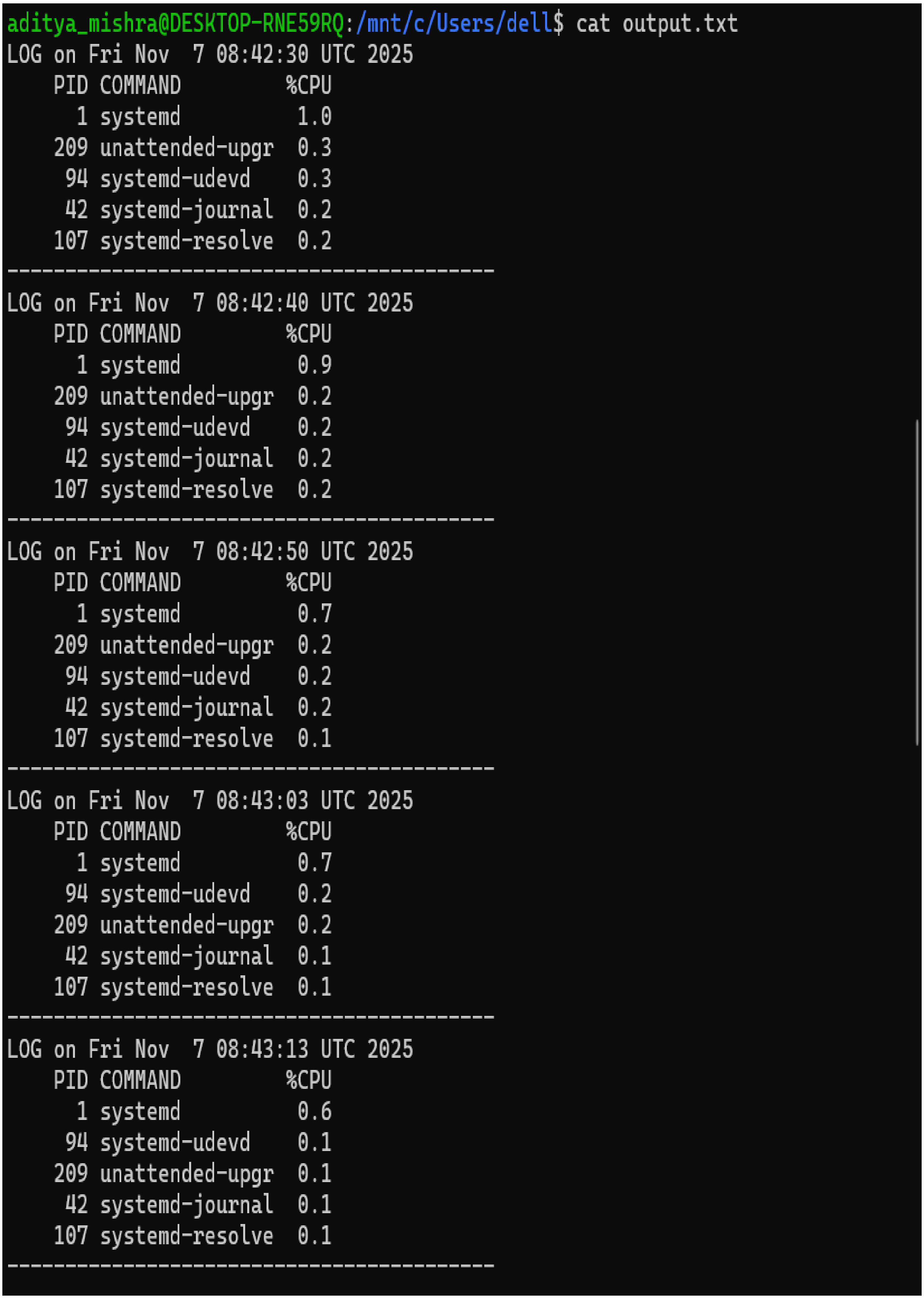
Command(s):

```
for i in {0..5}; do
    echo "LOG on $(date)" >> output.txt
    ps -eo pid,comm,%cpu --sort=-%cpu | head -6 >> output.txt
    echo "-------------------------------------------" >> output.txt
    sleep 10
done
```

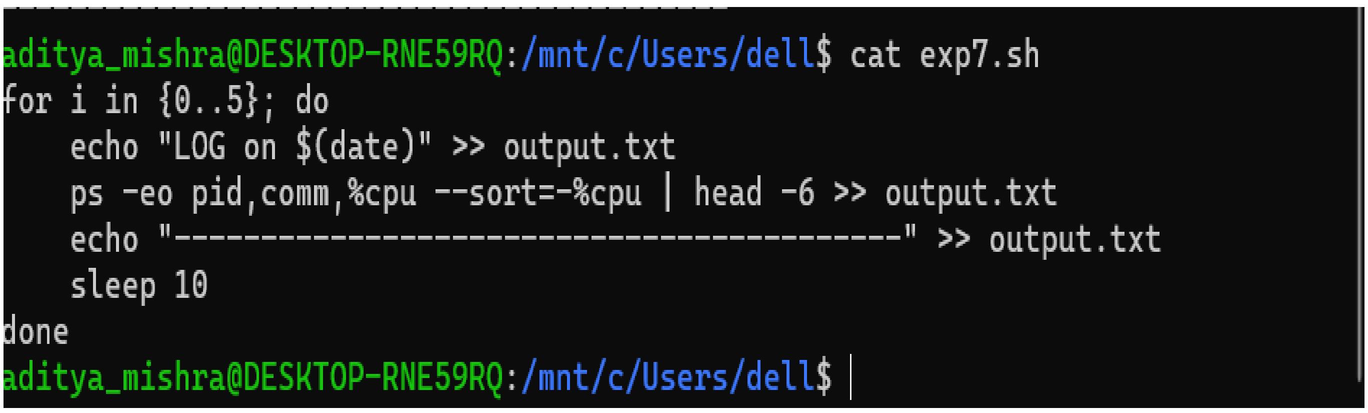
Output:



```
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ nano exp7.sh
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ bash exp7.sh
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ ls
 AppData
'Application Data'
 Contacts
 Cookies
 Desktop
 Documents
 Downloads
 Favorites
 Links
'Local Settings'
 Music
'My Documents'
 NTUSER.DAT
 NTUSER.DAT{2ad838bc-efea-11ee-a54d-000d3a94eaa1}.TM.blf
 NTUSER.DAT{2ad838bc-efea-11ee-a54d-000d3a94eaa1}.TMContainer000000000000000
00001.regtrans-ms
 NTUSER.DAT{2ad838bc-efea-11ee-a54d-000d3a94eaa1}.TMContainer000000000000000
00002.regtrans-ms
 NetHood
 OneDrive
 Pictures
 PixelSee
 PrintHood
 Recent
'Saved Games'
 Searches
 SendTo
'Start Menu'
 Templates
 Videos
 assets
 exp7.sh
 experiments
 fact.sh
 file1.txt
 hcflcm.sh
 ntuser.dat.LOG1
 ntuser.dat.LOG2
```



```
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ cat output.txt
LOG on Fri Nov  7 08:42:30 UTC 2025
    PID COMMAND         %CPU
      1 systemd          1.0
    209 unattended-upgr  0.3
     94 systemd-udevd    0.3
     42 systemd-journal  0.2
    107 systemd-resolve  0.2
------------------------------------------
LOG on Fri Nov  7 08:42:40 UTC 2025
    PID COMMAND         %CPU
      1 systemd          0.9
    209 unattended-upgr  0.2
     94 systemd-udevd    0.2
     42 systemd-journal  0.2
    107 systemd-resolve  0.2
------------------------------------------
LOG on Fri Nov  7 08:42:50 UTC 2025
    PID COMMAND         %CPU
      1 systemd          0.7
    209 unattended-upgr  0.2
     94 systemd-udevd    0.2
     42 systemd-journal  0.2
    107 systemd-resolve  0.1
------------------------------------------
LOG on Fri Nov  7 08:43:03 UTC 2025
    PID COMMAND         %CPU
      1 systemd          0.7
     94 systemd-udevd    0.2
    209 unattended-upgr  0.2
     42 systemd-journal  0.1
    107 systemd-resolve  0.1
------------------------------------------
LOG on Fri Nov  7 08:43:13 UTC 2025
    PID COMMAND         %CPU
      1 systemd          0.6
     94 systemd-udevd    0.1
    209 unattended-upgr  0.1
     42 systemd-journal  0.1
    107 systemd-resolve  0.1
------------------------------------------
```



```
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ cat exp7.sh
for i in {0..5}; do
    echo "LOG on $(date)" >> output.txt
    ps -eo pid,comm,%cpu --sort=-%cpu | head -6 >> output.txt
    echo "----------------------------------------" >> output.txt
    sleep 10
done
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$
```

Task 2

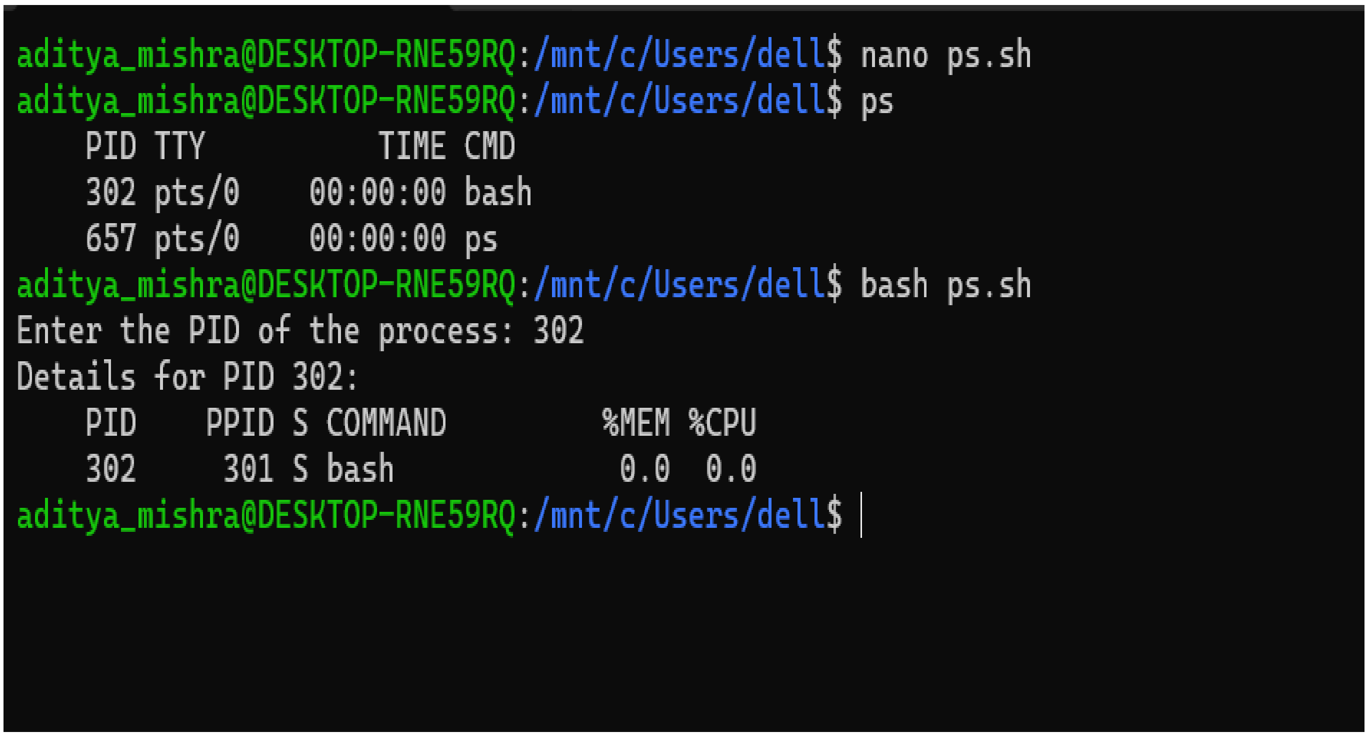
Task Statement:

Write a script that accepts a PID from the user and displays its details (state, parent process, memory usage).

Command(s):

```
#!/bin/bash

read -p "Enter the PID of the process: " pid

echo "Details for PID $pid:"
ps -p "$pid" -o pid,ppid,state,comm,%mem,%cpu
```

Output:



```
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ nano ps.sh
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ ps
    PID TTY          TIME CMD
    302 pts/0    00:00:00 bash
    657 pts/0    00:00:00 ps
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ bash ps.sh
Enter the PID of the process: 302
Details for PID 302:
    PID    PPID S COMMAND         %MEM %CPU
    302     301 S bash             0.0  0.0
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$
```

Task 3

Task Statement:

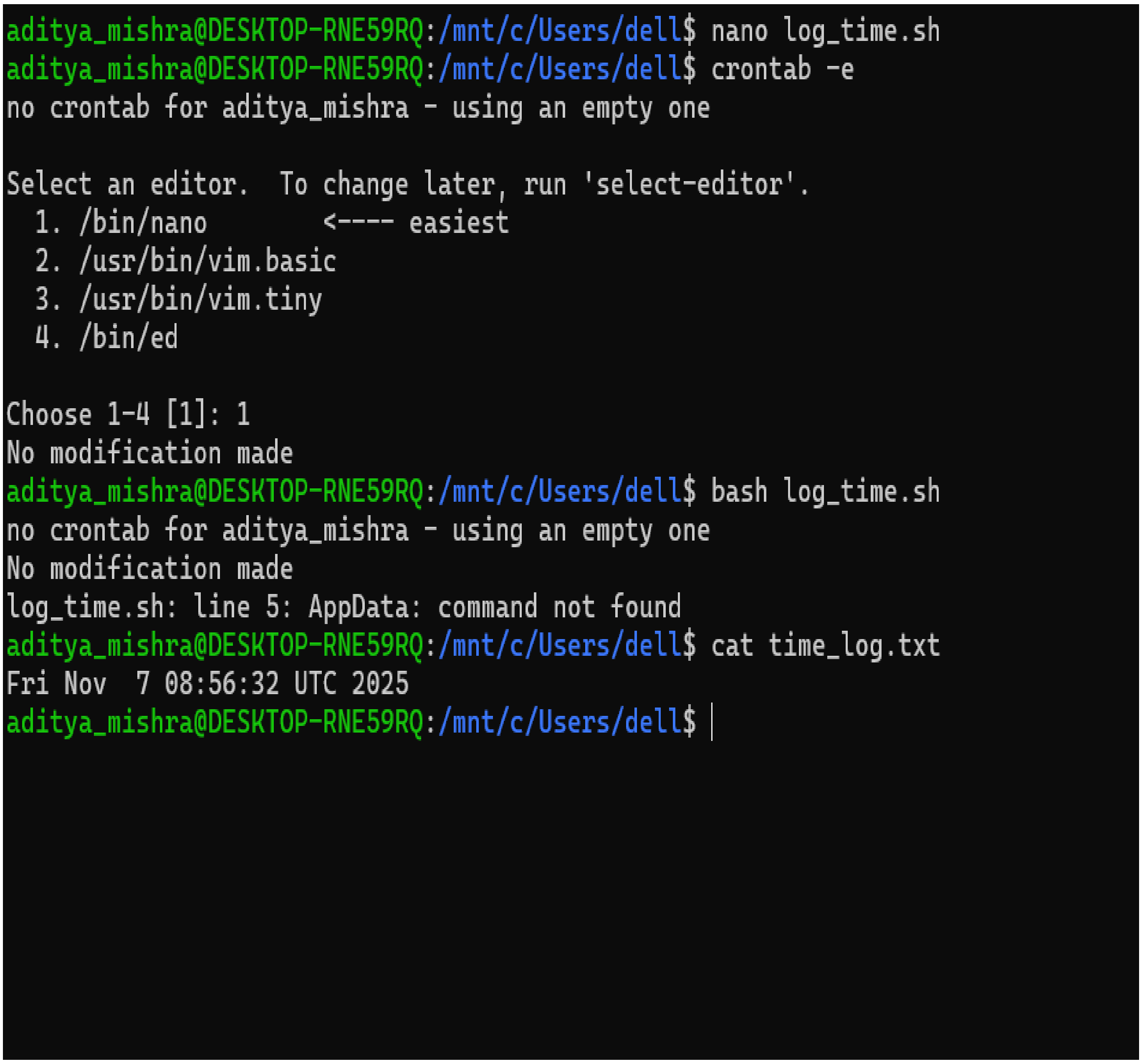
Create a script that schedules a task to append the current date and time to a log file every minute using cron.

Command(s):

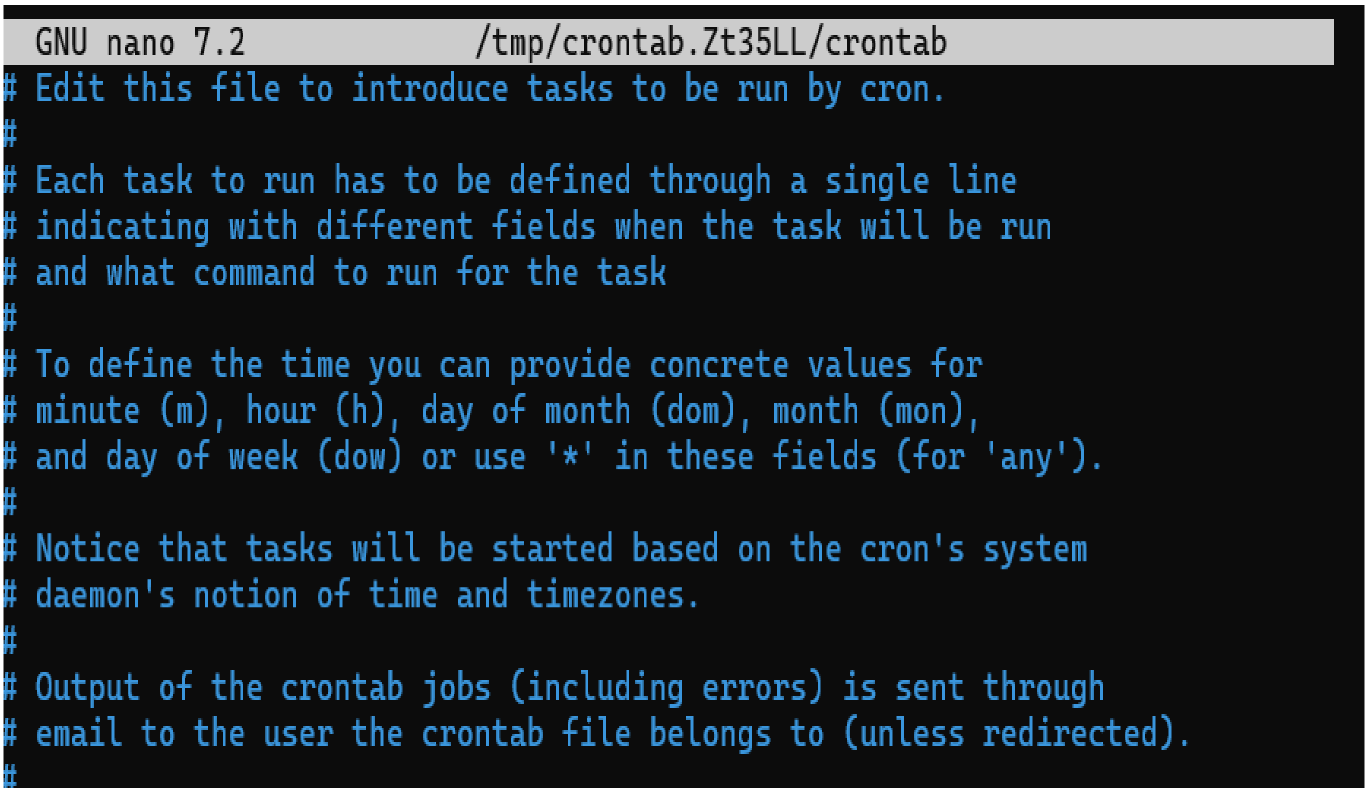
```
#!/bin/bash
echo "$(date)" >> time_log.txt
```

```
crontab -e

* * * * * ~/log_time.sh
```

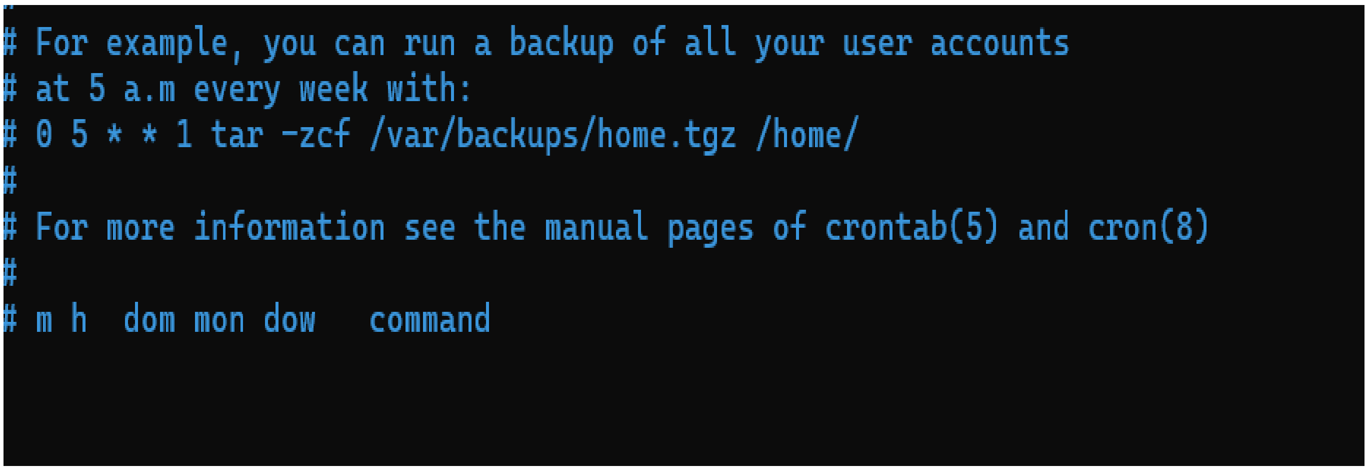
Output:



```
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ nano log_time.sh
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ crontab -e
no crontab for aditya_mishra - using an empty one

Select an editor.  To change later, run 'select-editor'.
  1. /bin/nano        <---- easiest
  2. /usr/bin/vim.basic
  3. /usr/bin/vim.tiny
  4. /bin/ed

Choose 1-4 [1]: 1
No modification made
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ bash log_time.sh
no crontab for aditya_mishra - using an empty one
No modification made
log_time.sh: line 5: AppData: command not found
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ cat time_log.txt
Fri Nov  7 08:56:32 UTC 2025
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$
```



```
  GNU nano 7.2              /tmp/crontab.Zt35LL/crontab
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
```



```
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
```

Task 4:

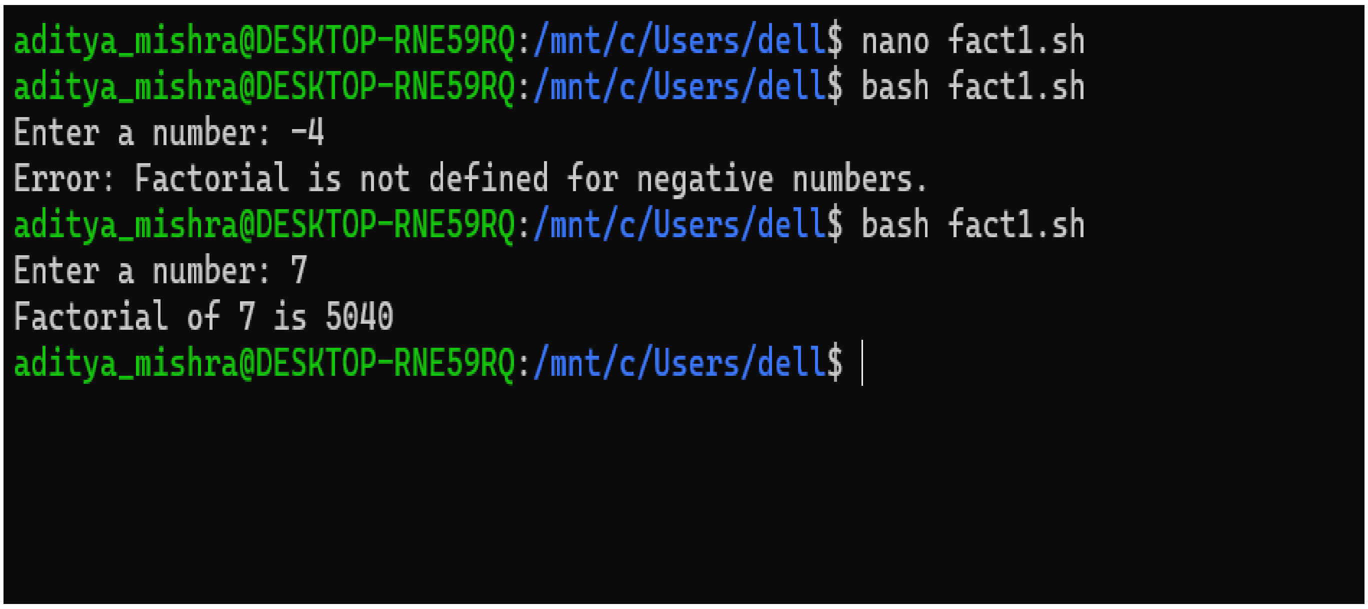
Task Statement:

Modify the factorial function to check if input is negative. If yes, display an error message.

Command(s):

```bash
#!/bin/bash

factorial() {
    local n=$1


    if [ $n -lt 0 ]; then
        echo "Error: Factorial is not defined for negative numbers."
        return 1
    fi

    local fact=1
    for (( i=1; i<=n; i++ )); do
        fact=$((fact * i))
    done
    echo "Factorial of $n is $fact"
}


read -p "Enter a number: " num
factorial $num
```

Output:



```
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ nano fact1.sh
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ bash fact1.sh
Enter a number: -4
Error: Factorial is not defined for negative numbers.
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ bash fact1.sh
Enter a number: 7
Factorial of 7 is 5040
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$
```

Task 5:

Task Statement:

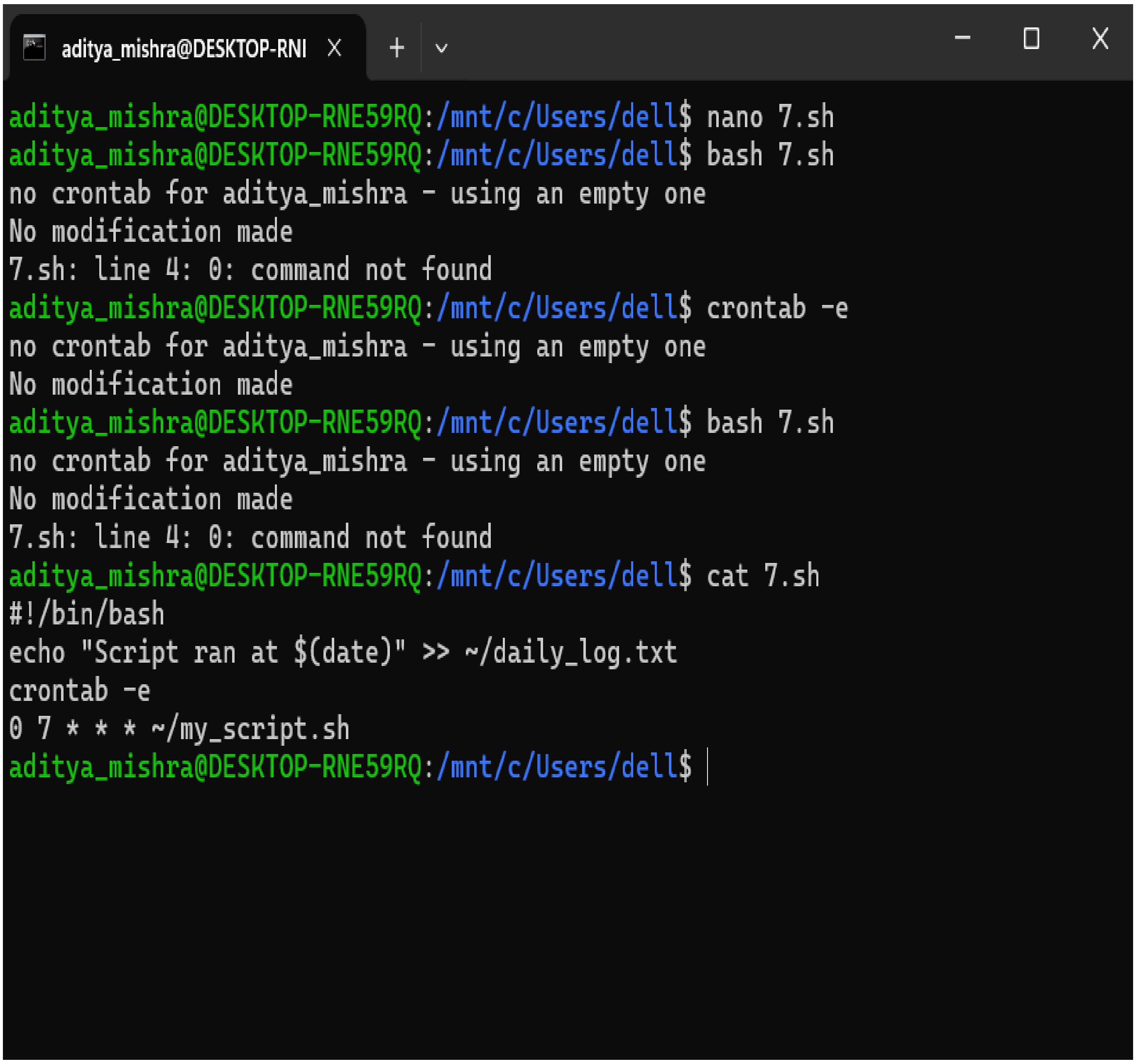
Schedule a script to run every day at 7:00 AM using `cron`.

Command(s):

```
#!/bin/bash
echo "Script ran at $(date)" >> ~/daily_log.txt
```

```
crontab -e
0 7 * * * ~/my_script.sh
```

Output:



```
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ nano 7.sh
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ bash 7.sh
no crontab for aditya_mishra - using an empty one
No modification made
7.sh: line 4: 0: command not found
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ crontab -e
no crontab for aditya_mishra - using an empty one
No modification made
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ bash 7.sh
no crontab for aditya_mishra - using an empty one
No modification made
7.sh: line 4: 0: command not found
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ cat 7.sh
#!/bin/bash
echo "Script ran at $(date)" >> ~/daily_log.txt
crontab -e
0 7 * * * ~/my_script.sh
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$
```

Result

- Learned to create and run shell scripts.
- Managed processes using background, foreground, and kill commands.
- Monitored processes with `ps` and `top`.
- Scheduled recurring tasks with `cron` and one-time tasks with `at`.

Challenges Faced & Learning Outcomes

- Challenge 1: Remembering the `crontab` time format. Solved by using online crontab generators and practice.
- Challenge 2: Ensuring `atd` service is running for `at` command. Fixed by starting the service with `systemctl start atd`.

Learning:

- Gained hands-on knowledge of process creation and termination.

- Learned job control and scheduling using `cron` and `at`.

Conclusion

This experiment provided practical experience with shell scripting, process management, and scheduling. These are critical skills for system administrators to automate and control Linux environments effectively.