Experiment 7: Shell Programming, Process and Scheduling

Name: Vaishnavi Kumari Roll No.: 590029048 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

Exercise 1: Writing a basic shell script

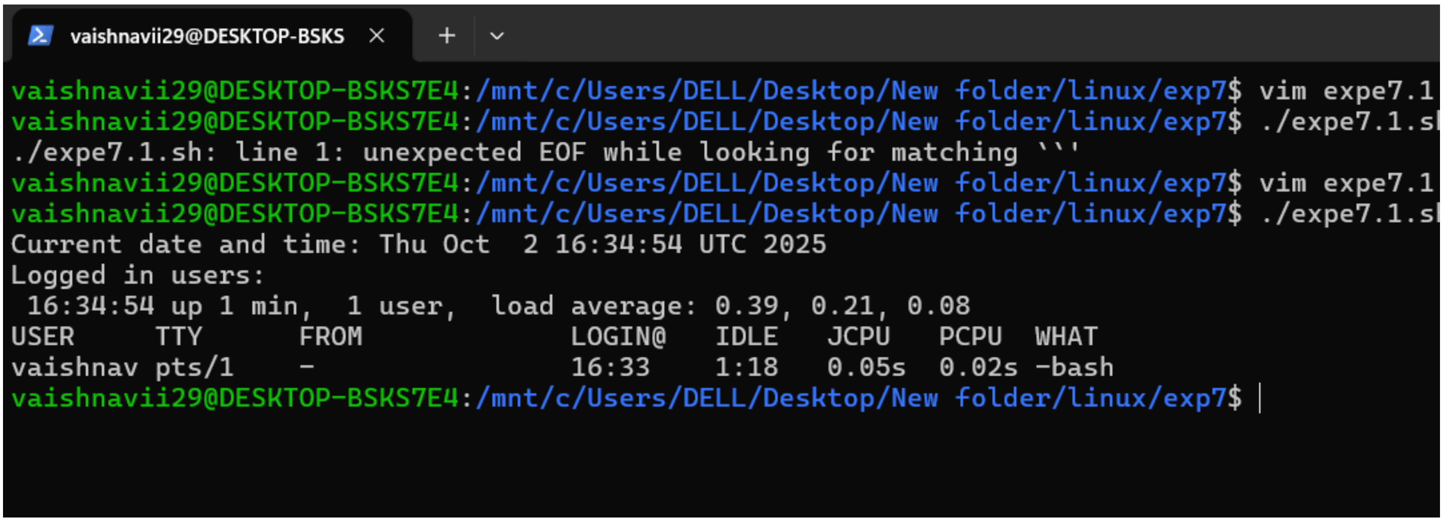
Task Statement:

Create a shell script that prints the current date, time, and the list of logged-in users.

Command(s):

```
#!/bin/bash
echo "Current date and time: $(date)"
echo "Logged in users:"
w
```

Output:



vaishnavii29@DESKTOP-BSKS

```
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Desktop/New folder/linux/exp7$ vim expe7.1
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Desktop/New folder/linux/exp7$ ./expe7.1.s
./expe7.1.sh: line 1: unexpected EOF while looking for matching `''
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Desktop/New folder/linux/exp7$ vim expe7.1
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Desktop/New folder/linux/exp7$ ./expe7.1.s
Current date and time: Thu Oct  2 16:34:54 UTC 2025
Logged in users:
 16:34:54 up 1 min,  1 user,  load average: 0.39, 0.21, 0.08
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU  WHAT
vaishnav pts/1    -                16:33    1:18   0.05s  0.02s -bash
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Desktop/New folder/linux/exp7$ |
```

Exercise 2: Background and foreground processes

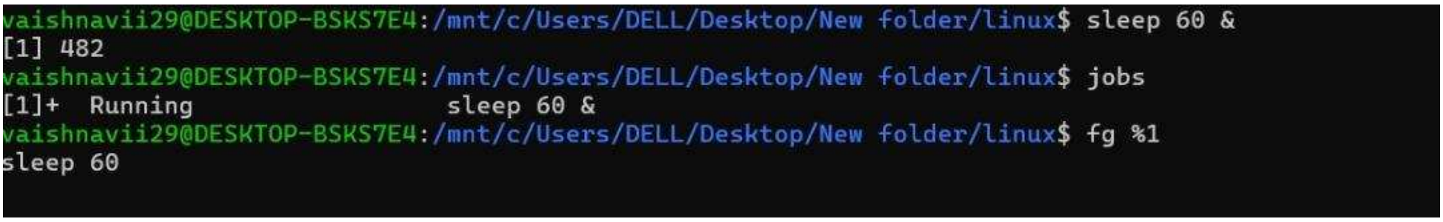
Task Statement:

Run a process in background and bring it to the foreground.

Command(s):

```
sleep 60 &
jobs
fg %1
```

Output:



```
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Desktop/New folder/linux$ sleep 60 &
[1] 482
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Desktop/New folder/linux$ jobs
[1]+  Running                 sleep 60 &
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Desktop/New folder/linux$ fg %1
sleep 60
```

Exercise 3: Killing a process

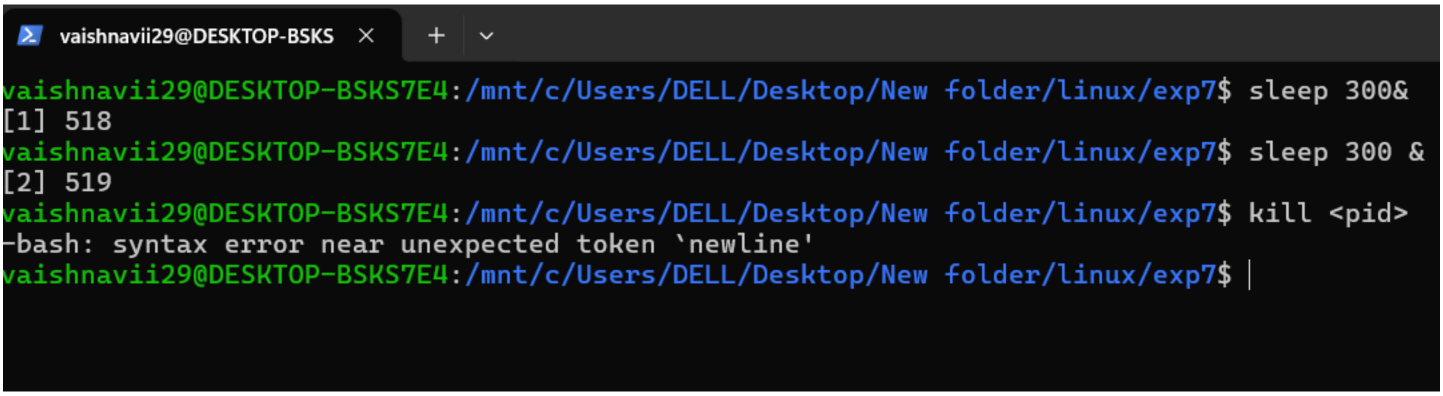
Task Statement:

Start a process and terminate it using `kill`.

Command(s):

```
sleep 300 &
kill <pid>
```

Output:



```
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Desktop/New folder/linux/exp7$ sleep 300&
[1] 518
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Desktop/New folder/linux/exp7$ sleep 300 &
[2] 519
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Desktop/New folder/linux/exp7$ kill <pid>
-bash: syntax error near unexpected token `newline'
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Desktop/New folder/linux/exp7$ |
```

Exercise 4: Monitoring processes

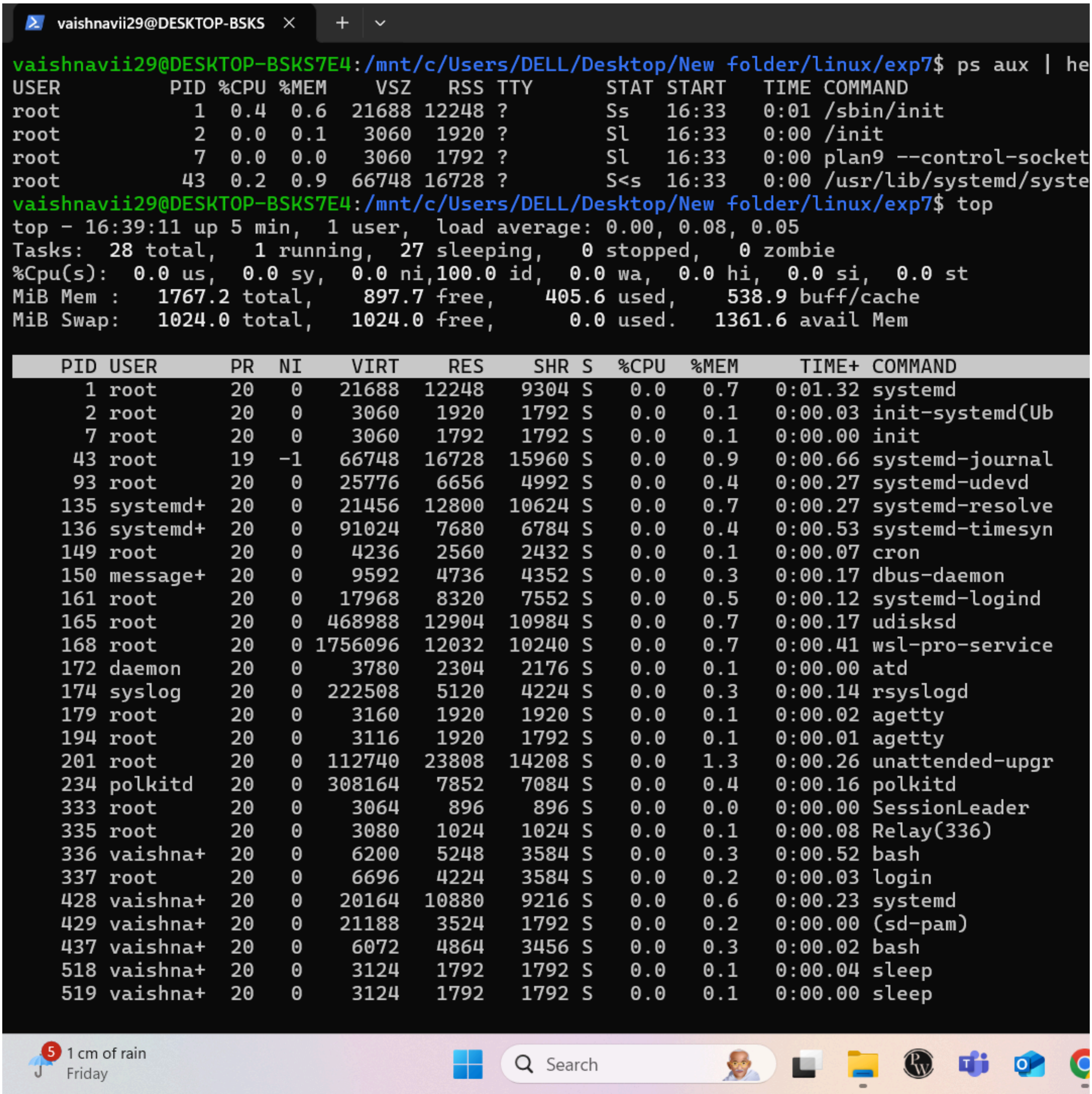
Task Statement:

Use `ps` and `top` to monitor processes.

Command(s):

```
ps aux | head -5
top
```

Output:



```
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Desktop/New folder/linux/exp7$ ps aux | he
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root           1  0.4  0.6  21688 12248 ?        Ss   16:33   0:01 /sbin/init
root           2  0.0  0.1   3060  1920 ?        Sl   16:33   0:00 /init
root           7  0.0  0.0   3060  1792 ?        Sl   16:33   0:00 plan9 --control-socket
root          43  0.2  0.9  66748 16728 ?        S<s  16:33   0:00 /usr/lib/systemd/syste
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Desktop/New folder/linux/exp7$ top
top - 16:39:11 up 5 min,  1 user,  load average: 0.00, 0.08, 0.05
Tasks:  28 total,   1 running,  27 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :   1767.2 total,    897.7 free,    405.6 used,    538.9 buff/cache
MiB Swap:   1024.0 total,   1024.0 free,      0.0 used.   1361.6 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
      1 root      20   0   21688  12248   9304 S   0.0   0.7   0:01.32 systemd
      2 root      20   0    3060   1920   1792 S   0.0   0.1   0:00.03 init-systemd(Ub
      7 root      20   0    3060   1792   1792 S   0.0   0.1   0:00.00 init
     43 root      19  -1   66748  16728  15960 S   0.0   0.9   0:00.66 systemd-journal
     93 root      20   0   25776   6656   4992 S   0.0   0.4   0:00.27 systemd-udevd
    135 systemd+  20   0   21456  12800  10624 S   0.0   0.7   0:00.27 systemd-resolve
    136 systemd+  20   0   91024   7680   6784 S   0.0   0.4   0:00.53 systemd-timesyn
    149 root      20   0    4236   2560   2432 S   0.0   0.1   0:00.07 cron
    150 message+  20   0    9592   4736   4352 S   0.0   0.3   0:00.17 dbus-daemon
    161 root      20   0   17968   8320   7552 S   0.0   0.5   0:00.12 systemd-logind
    165 root      20   0  468988  12904  10984 S   0.0   0.7   0:00.17 udisksd
    168 root      20   0 1756096  12032  10240 S   0.0   0.7   0:00.41 wsl-pro-service
    172 daemon    20   0    3780   2304   2176 S   0.0   0.1   0:00.00 atd
    174 syslog    20   0  222508   5120   4224 S   0.0   0.3   0:00.14 rsyslogd
    179 root      20   0    3160   1920   1920 S   0.0   0.1   0:00.02 agetty
    194 root      20   0    3116   1920   1792 S   0.0   0.1   0:00.01 agetty
    201 root      20   0  112740  23808  14208 S   0.0   1.3   0:00.26 unattended-upgr
    234 polkitd   20   0  308164   7852   7084 S   0.0   0.4   0:00.16 polkitd
    333 root      20   0    3064    896    896 S   0.0   0.0   0:00.00 SessionLeader
    335 root      20   0    3080   1024   1024 S   0.0   0.1   0:00.08 Relay(336)
    336 vaishna+  20   0    6200   5248   3584 S   0.0   0.3   0:00.52 bash
    337 root      20   0    6696   4224   3584 S   0.0   0.2   0:00.03 login
    428 vaishna+  20   0   20164  10880   9216 S   0.0   0.6   0:00.23 systemd
    429 vaishna+  20   0   21188   3524   1792 S   0.0   0.2   0:00.00 (sd-pam)
    437 vaishna+  20   0    6072   4864   3456 S   0.0   0.3   0:00.02 bash
    518 vaishna+  20   0    3124   1792   1792 S   0.0   0.1   0:00.04 sleep
    519 vaishna+  20   0    3124   1792   1792 S   0.0   0.1   0:00.00 sleep
```

Exercise 5: Using cron for scheduling

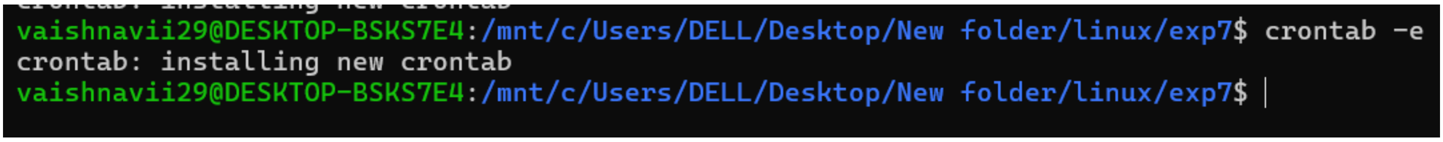
Task Statement:

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

Command(s):

```
crontab -e
# Add the following line
0 7 * * * /home/user/myscript.sh
```

Output:



```
crontab: installing new crontab
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Desktop/New folder/linux/exp7$ crontab -e
crontab: installing new crontab
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Desktop/New folder/linux/exp7$ |
```

Exercise 6: Using `at` for one-time scheduling

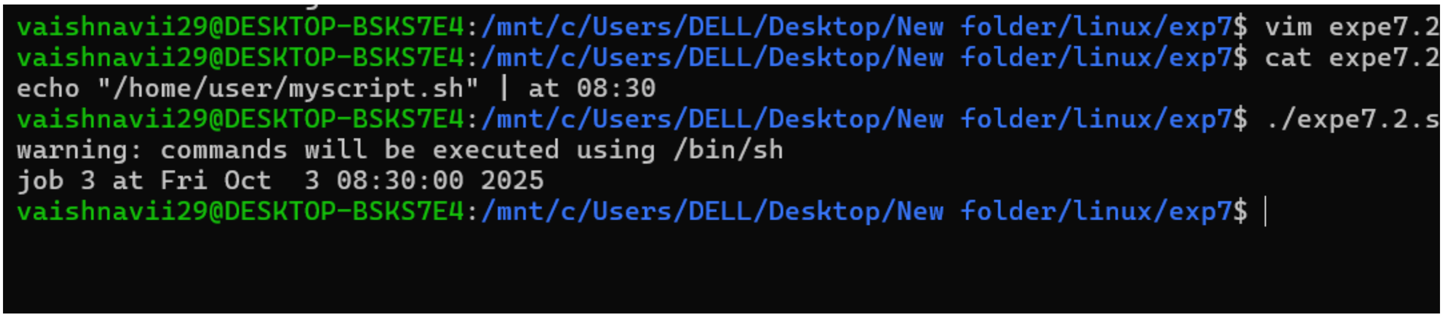
Task Statement:

Schedule a script to run once at a specified time using `at`.

Command(s):

```
echo "/home/user/myscript.sh" | at 08:30
atq
```

Output:



```
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Desktop/New folder/linux/exp7$ vim expe7.2
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Desktop/New folder/linux/exp7$ cat expe7.2
echo "/home/user/myscript.sh" | at 08:30
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Desktop/New folder/linux/exp7$ ./expe7.2.s
warning: commands will be executed using /bin/sh
job 3 at Fri Oct  3 08:30:00 2025
vaishnavii29@DESKTOP-BSKS7E4:/mnt/c/Users/DELL/Desktop/New folder/linux/exp7$ |
```

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`.

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.