Experiment 7: Shell Programming, Process and Scheduling

Experiment 7: Shell Programming, Process and Scheduling

Name: Rishika Purushotham Roll No.: 590029145 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.

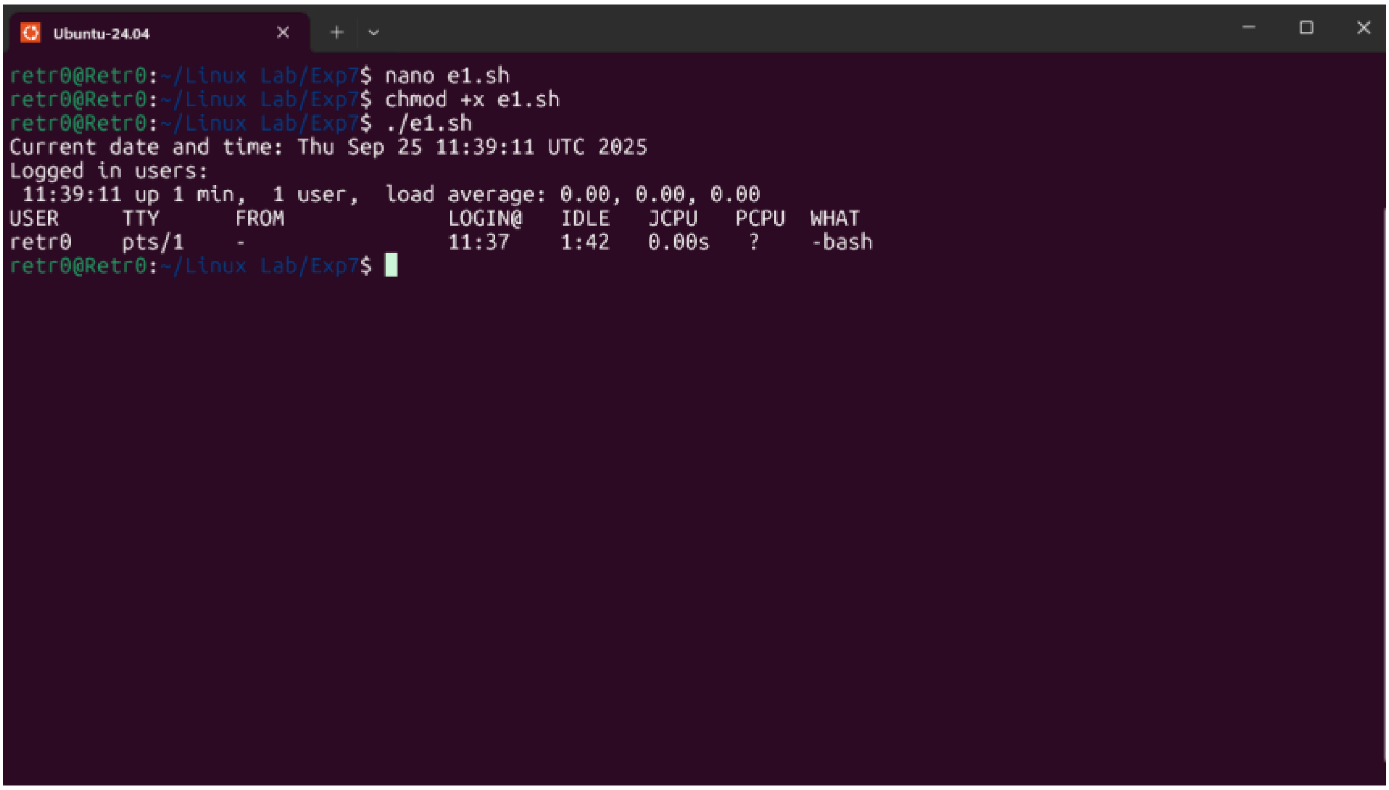
Requirements
A Linux machine with bash shell.
Access to process management commands (ps ,top , kill , jobs , fg , bg).
Access to scheduling utilities (cron , at).

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:



```
retr0@Retr0:~/Linux Lab/Exp7$ nano e1.sh
retr0@Retr0:~/Linux Lab/Exp7$ chmod +x e1.sh
retr0@Retr0:~/Linux Lab/Exp7$ ./e1.sh
Current date and time: Thu Sep 25 11:39:11 UTC 2025
Logged in users:
 11:39:11 up 1 min,  1 user,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU  WHAT
retr0    pts/1    -                11:37    1:42   0.00s  ?     -bash
retr0@Retr0:~/Linux Lab/Exp7$
```

Exercise 2: Background and foreground processes

Task Statement:

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

Command(s):

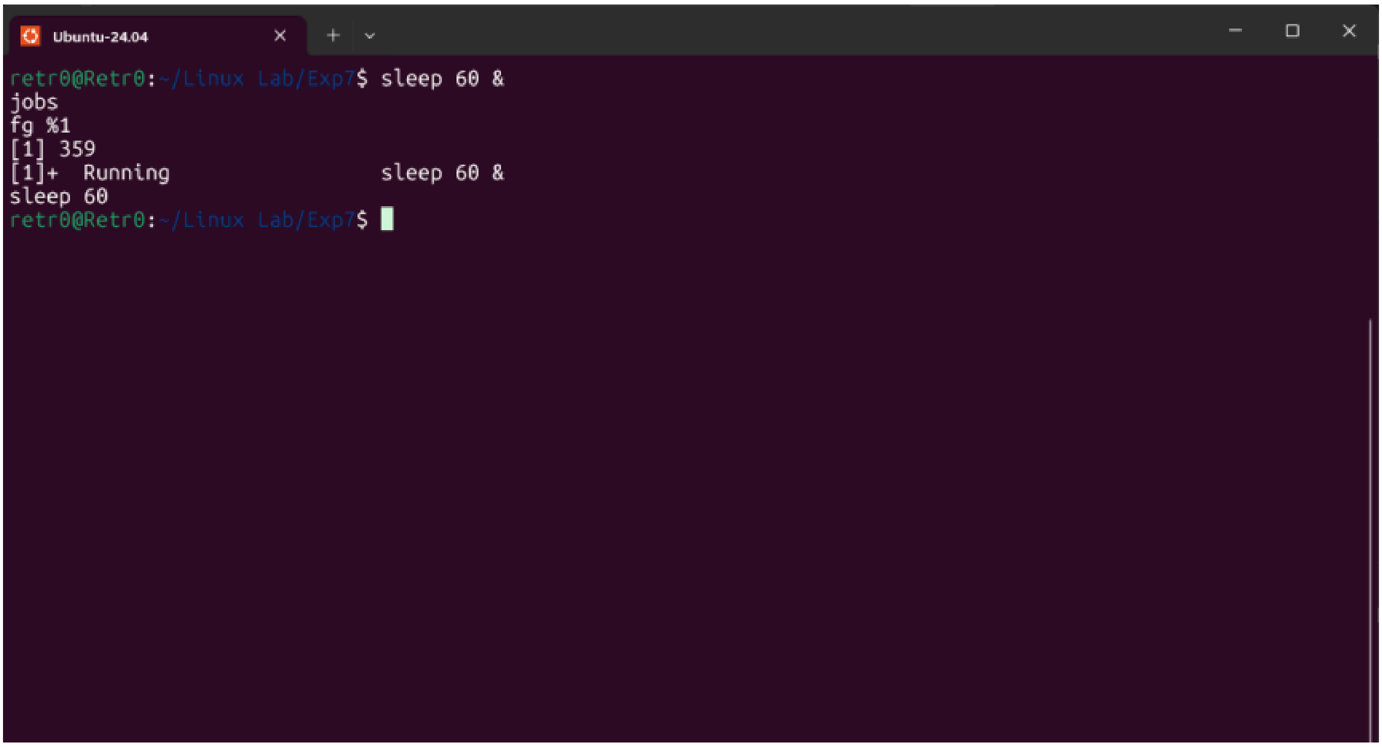
#!/bin/bash

sleep 60 &

jobs

fg %1

Output:



```
retr0@Retr0:~/Linux Lab/Exp7$ sleep 60 &
jobs
fg %1
[1] 359
[1]+  Running                 sleep 60 &
sleep 60
retr0@Retr0:~/Linux Lab/Exp7$
```

Exercise 3: Killing a process

Task Statement:

Start a process and terminate it using kill .

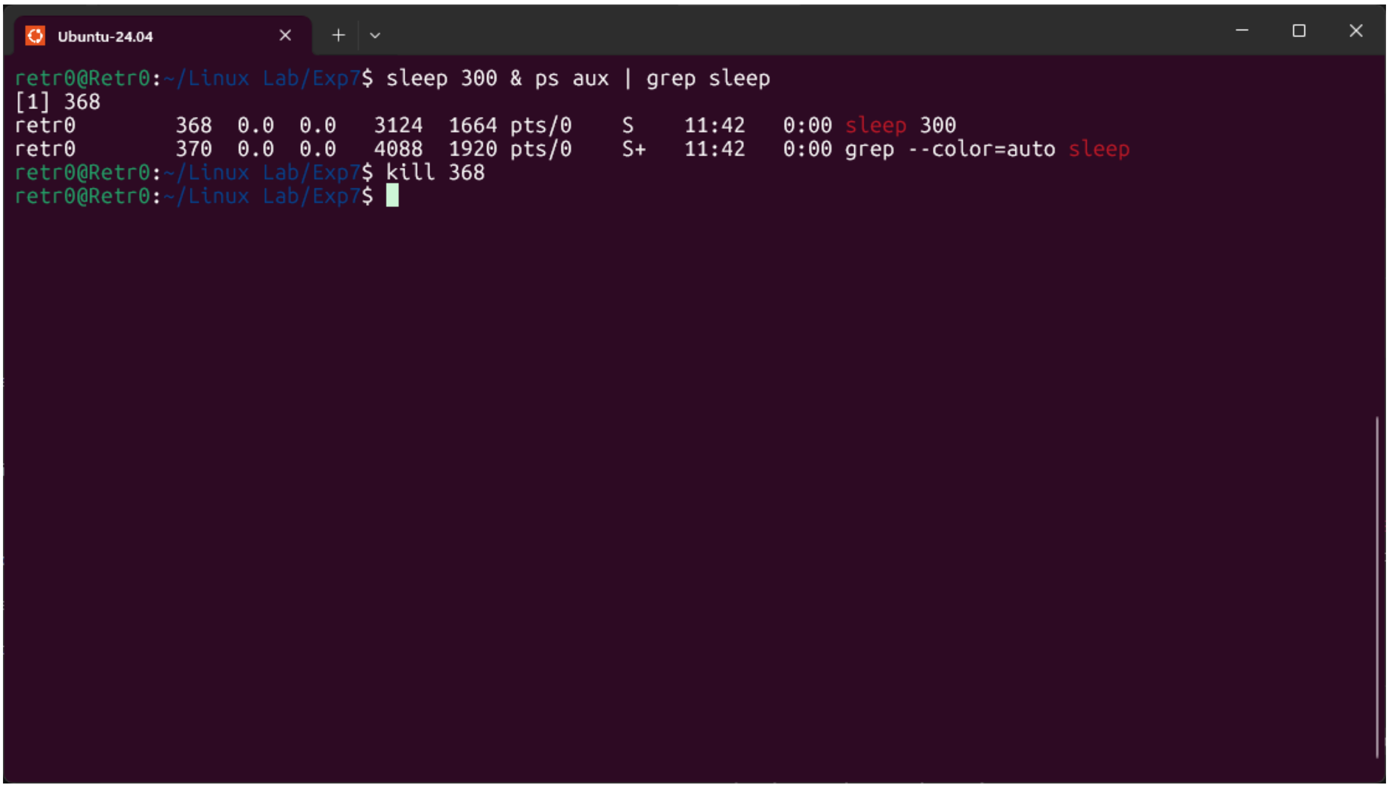
Command(s):

sleep 300 &

ps aux | grep sleep

kill

Output:



```
retr0@Retr0:~/Linux Lab/Exp7$ sleep 300 & ps aux | grep sleep
[1] 368
retr0        368  0.0  0.0   3124  1664 pts/0    S    11:42   0:00 sleep 300
retr0        370  0.0  0.0   4088  1920 pts/0    S+   11:42   0:00 grep --color=auto sleep
retr0@Retr0:~/Linux Lab/Exp7$ kill 368
retr0@Retr0:~/Linux Lab/Exp7$ 
```

Exercise 4: Monitoring processes

Task Statement:

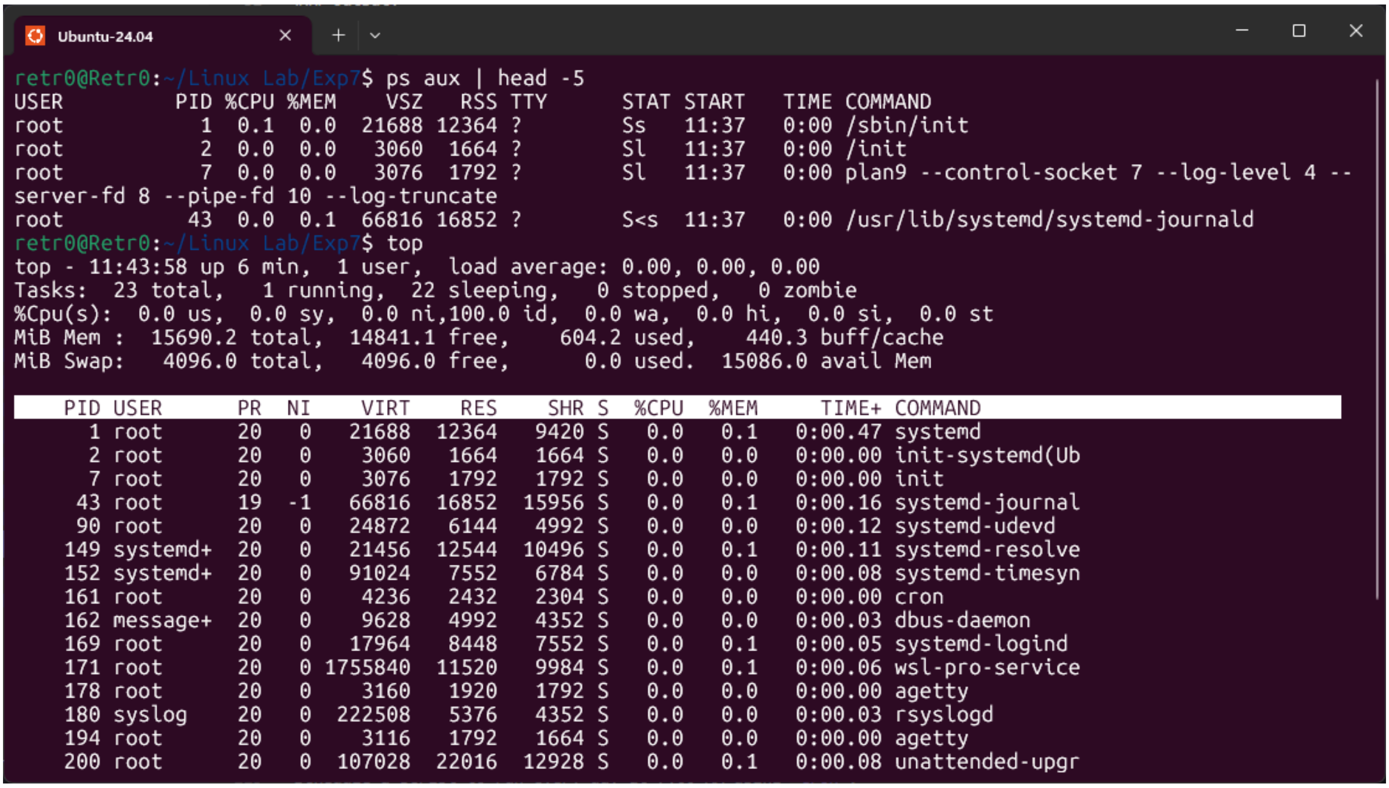
Use ps and top to monitor processes.

Command(s):

ps aux | head -5

top

Output:



```
retr0@Retr0:~/Linux Lab/Exp7$ ps aux | head -5
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root           1  0.1  0.0  21688 12364 ?        Ss   11:37   0:00 /sbin/init
root           2  0.0  0.0   3060  1664 ?        Sl   11:37   0:00 /init
root           7  0.0  0.0   3076  1792 ?        Sl   11:37   0:00 plan9 --control-socket 7 --log-level 4 --
server-fd 8 --pipe-fd 10 --log-truncate
root          43  0.0  0.1  66816 16852 ?        S<s  11:37   0:00 /usr/lib/systemd/systemd-journald
retr0@Retr0:~/Linux Lab/Exp7$ top
top - 11:43:58 up 6 min,  1 user,  load average: 0.00, 0.00, 0.00
Tasks:  23 total,   1 running,  22 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 :  15690.2 total,  14841.1 free,    604.2 used,    440.3 buff/cache
MiB Swap:   4096.0 total,   4096.0 free,      0.0 used.  15086.0 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
      1 root      20   0   21688  12364   9420 S   0.0   0.1   0:00.47 systemd
      2 root      20   0    3060   1664   1664 S   0.0   0.0   0:00.00 init-systemd(Ub
      7 root      20   0    3076   1792   1792 S   0.0   0.0   0:00.00 init
     43 root      19  -1   66816  16852  15956 S   0.0   0.1   0:00.16 systemd-journal
     90 root      20   0   24872   6144   4992 S   0.0   0.0   0:00.12 systemd-udevd
    149 systemd+  20   0   21456  12544  10496 S   0.0   0.1   0:00.11 systemd-resolve
    152 systemd+  20   0   91024   7552   6784 S   0.0   0.0   0:00.08 systemd-timesyn
    161 root      20   0    4236   2432   2304 S   0.0   0.0   0:00.00 cron
    162 message+  20   0    9628   4992   4352 S   0.0   0.0   0:00.03 dbus-daemon
    169 root      20   0   17964   8448   7552 S   0.0   0.1   0:00.05 systemd-logind
    171 root      20   0 1755840  11520   9984 S   0.0   0.1   0:00.06 wsl-pro-service
    178 root      20   0    3160   1920   1792 S   0.0   0.0   0:00.00 agetty
    180 syslog    20   0  222508   5376   4352 S   0.0   0.0   0:00.03 rsyslogd
    194 root      20   0    3116   1792   1664 S   0.0   0.0   0:00.00 agetty
    200 root      20   0  107028  22016  12928 S   0.0   0.1   0:00.08 unattended-upgr
```

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

0 7 * * * /home/retr0/myscript.sh

Output:

exp7_cron

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

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, andscheduling. These are critical skills for system administrators to automate and control Linux environments effectively.
