Experiment 8: Shell Scripting – Jobs, File Comparison, Process & Memory Monitoring

Name: Aditya Mishra Roll No.: 590029219 Date: 2025-10-02

Aim:

- To learn about background and foreground job control in Linux.
- To compare files and check for content differences.
- To count and monitor processes and memory usage.
- To practice searching patterns in files using shell scripting.

Requirements:

- A Linux machine with Bash shell.
- Access to commands: `jobs`, `fg`, `bg`, `kill`, `cmp`, `diff`, `ps`, `free`, `grep`.
- Knowledge of loops, conditionals, and user input in shell scripting.

Theory:

Shell scripting in Linux allows automation of tasks and process management. Jobs can run in the **background** or **foreground**, and commands like `jobs`, `fg`, and `kill` allow control over them. File comparison utilities like `cmp` and `diff` help verify content differences. Monitoring system resources like processes and memory is essential for system performance management. Pattern searching using `grep` allows extraction and counting of specific text lines in files. Combining these commands into scripts enhances productivity and automates repetitive tasks.

Procedure & Observations

Task 1

Task Statement:

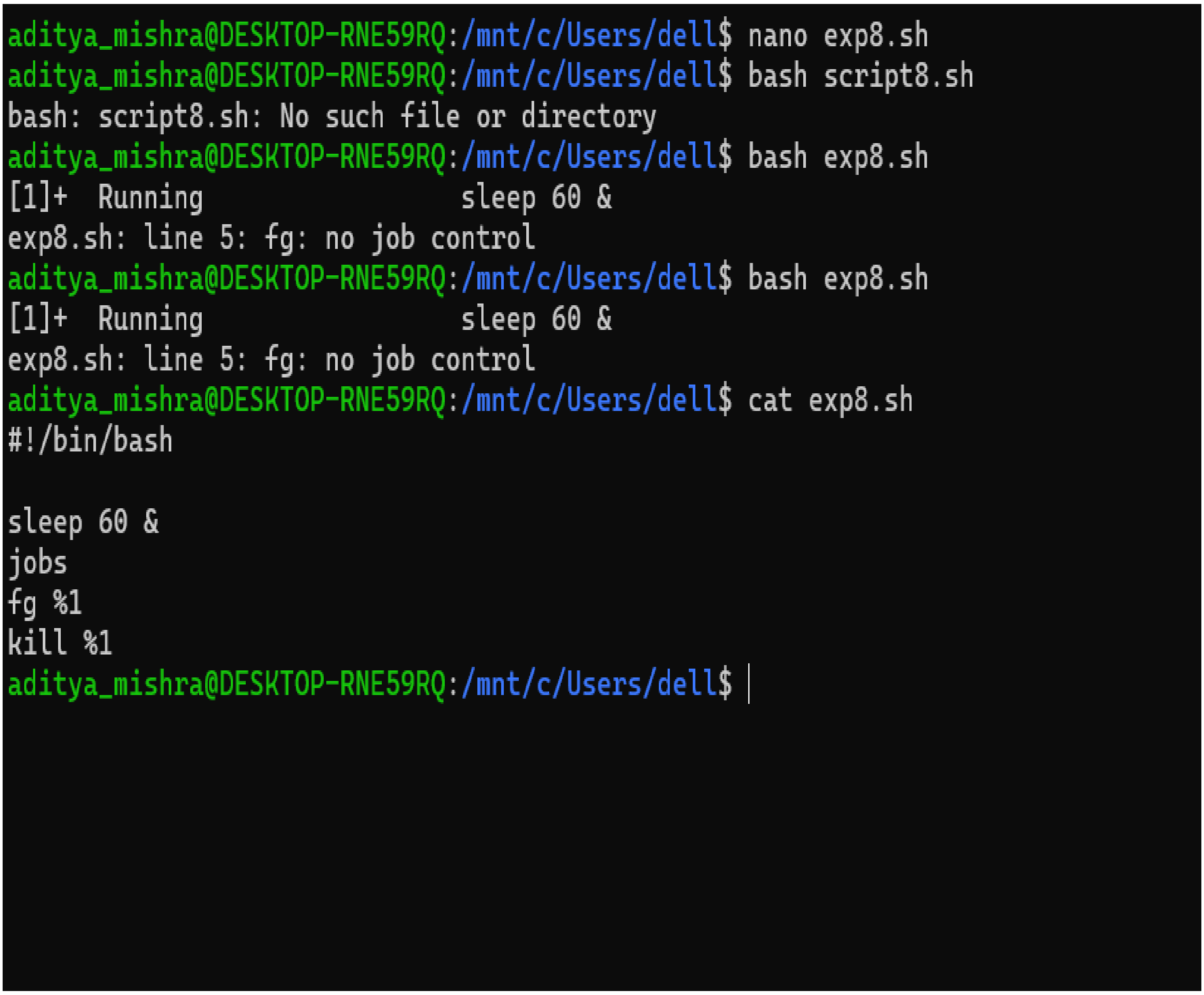
Write a script that starts a background job (e.g., `sleep 60`), lists all jobs, brings the job to the foreground, and then terminates it.

Command(s):

```
#!/bin/bash

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

Output:



```
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ nano exp8.sh
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ bash script8.sh
bash: script8.sh: No such file or directory
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ bash exp8.sh
[1]+  Running                 sleep 60 &
exp8.sh: line 5: fg: no job control
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ bash exp8.sh
[1]+  Running                 sleep 60 &
exp8.sh: line 5: fg: no job control
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ cat exp8.sh
#!/bin/bash

sleep 60 &
jobs
fg %1
kill %1
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$
```

Task 2

Task Statement:

Create a script that compares two files and displays whether their contents are identical or different.

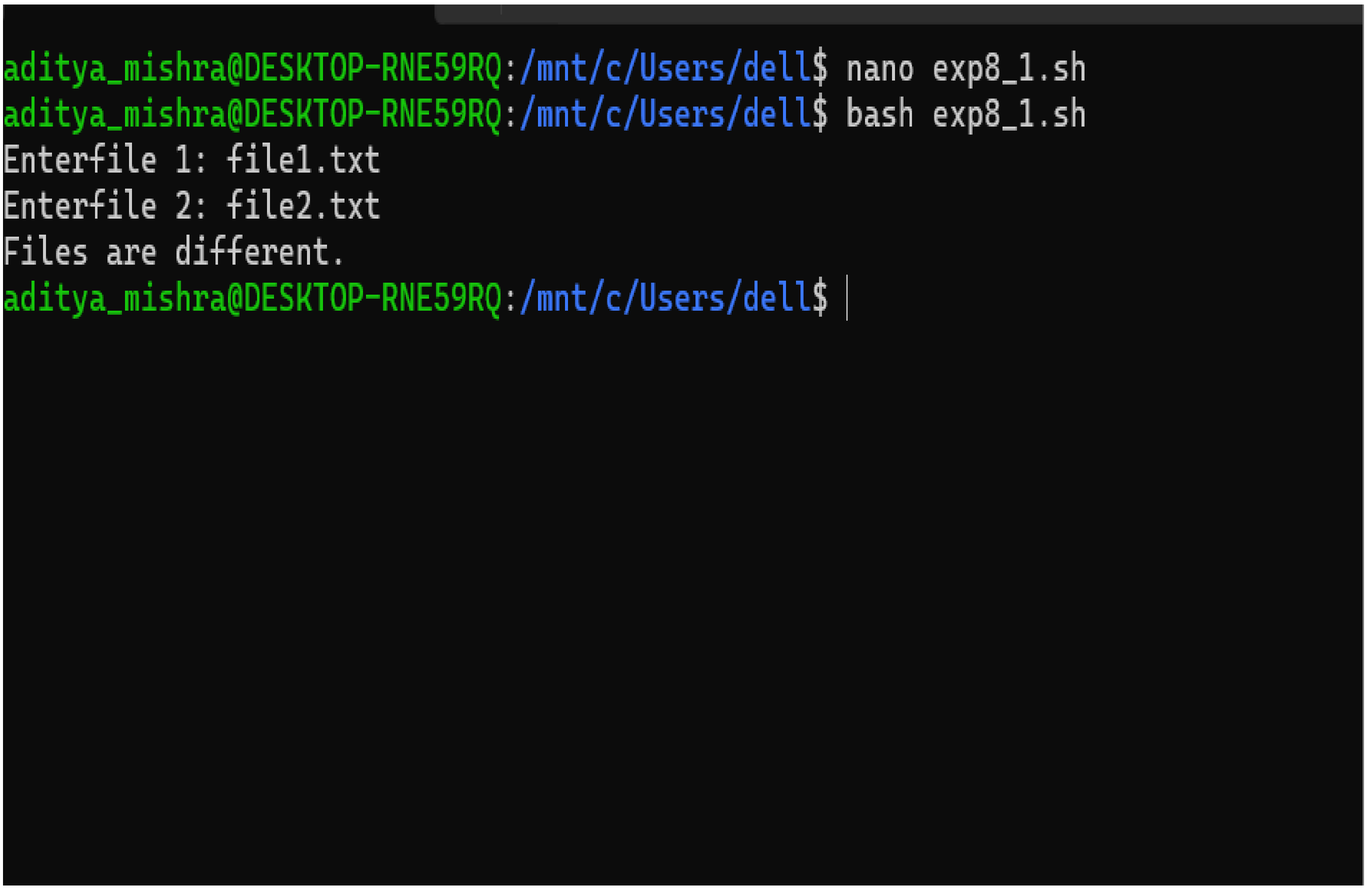
Command(s):

```
#!/bin/bash

read -p "Enterfile 1: " file1
read -p "Enterfile 2: " file2

if cmp -s "$file1" "$file2"; then
    echo "Files are identical."
else
    echo "Files are different."
fi
```

Output:



```
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ nano exp8_1.sh
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ bash exp8_1.sh
Enterfile 1: file1.txt
Enterfile 2: file2.txt
Files are different.
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$
```

Task 3

Task Statement:

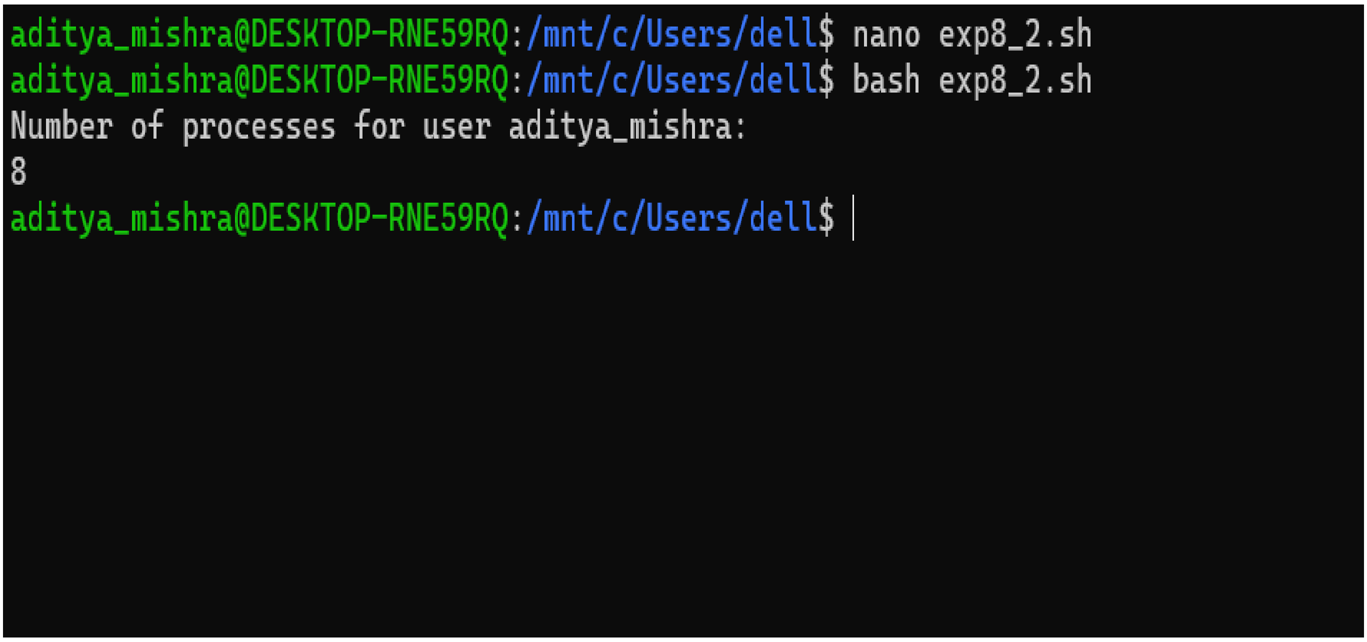
Write a script that counts the number of processes currently being run by your user.

Command(s):

```
#!/bin/bash

echo "Number of processes for user $USER:"
ps -u $USER | wc -l
```

Output:



```
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ nano exp8_2.sh
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ bash exp8_2.sh
Number of processes for user aditya_mishra:
8
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$
```

Task 4

Task Statement:

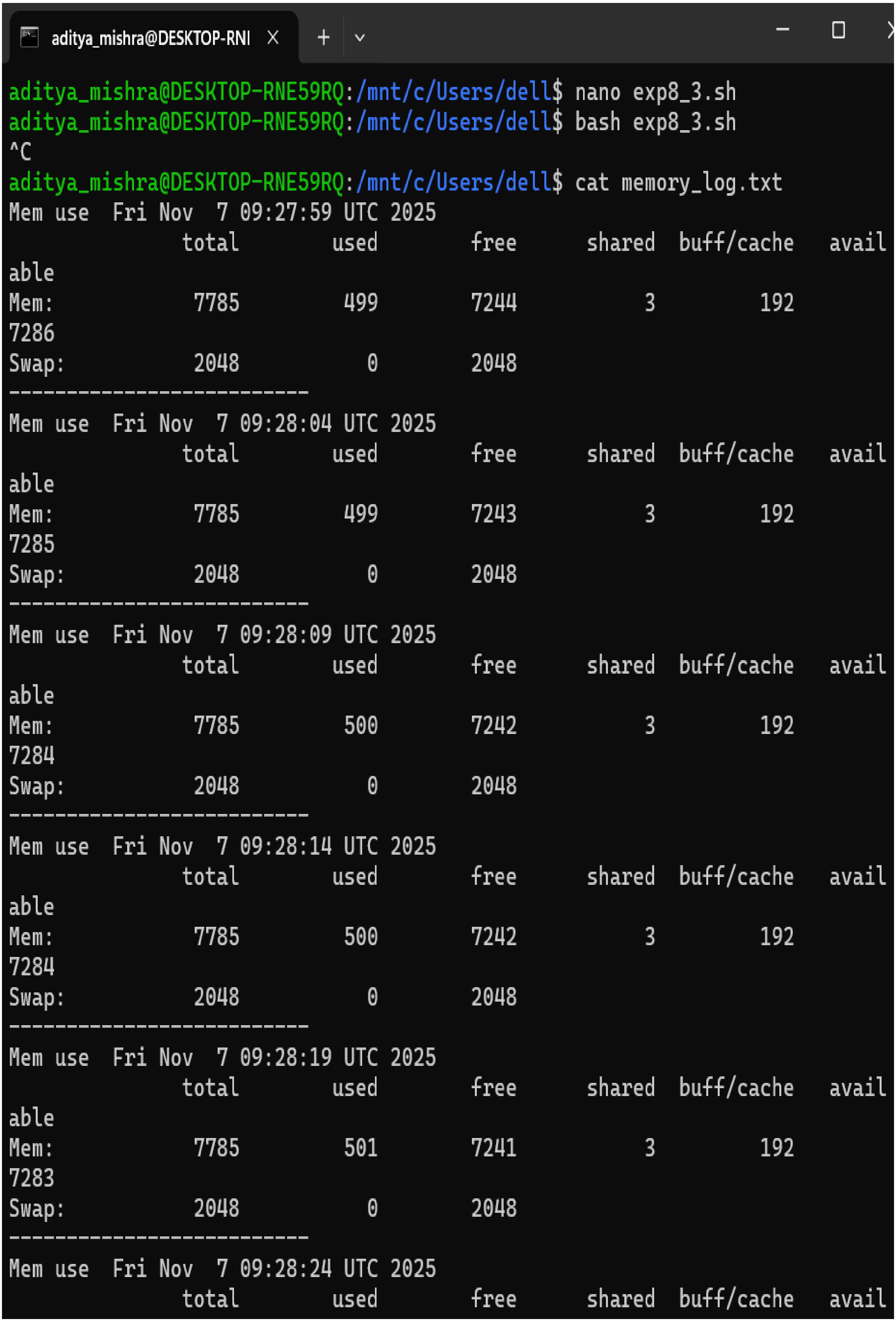
Develop a script that monitors memory usage every 5 seconds and logs it into a file.

Command(s):

```
#!/bin/bash

while true; do
    echo "Mem use  $(date)" >> memory_log.txt
    free -m >> memory_log.txt
    echo "--------------------------" >> memory_log.txt
    sleep 5
done
```

Output:



```
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ nano exp8_3.sh
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ bash exp8_3.sh
^C
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ cat memory_log.txt
Mem use  Fri Nov  7 09:27:59 UTC 2025
               total        used        free      shared  buff/cache   avail
able
Mem:            7785         499        7244           3         192
7286
Swap:           2048           0        2048
--------------------------
Mem use  Fri Nov  7 09:28:04 UTC 2025
               total        used        free      shared  buff/cache   avail
able
Mem:            7785         499        7243           3         192
7285
Swap:           2048           0        2048
--------------------------
Mem use  Fri Nov  7 09:28:09 UTC 2025
               total        used        free      shared  buff/cache   avail
able
Mem:            7785         500        7242           3         192
7284
Swap:           2048           0        2048
--------------------------
Mem use  Fri Nov  7 09:28:14 UTC 2025
               total        used        free      shared  buff/cache   avail
able
Mem:            7785         500        7242           3         192
7284
Swap:           2048           0        2048
--------------------------
Mem use  Fri Nov  7 09:28:19 UTC 2025
               total        used        free      shared  buff/cache   avail
able
Mem:            7785         501        7241           3         192
7283
Swap:           2048           0        2048
--------------------------
Mem use  Fri Nov  7 09:28:24 UTC 2025
               total        used        free      shared  buff/cache   avail
```

Task 5

Task Statement:

Write a script that prompts for a filename and a search pattern, then displays the count of matching lines.

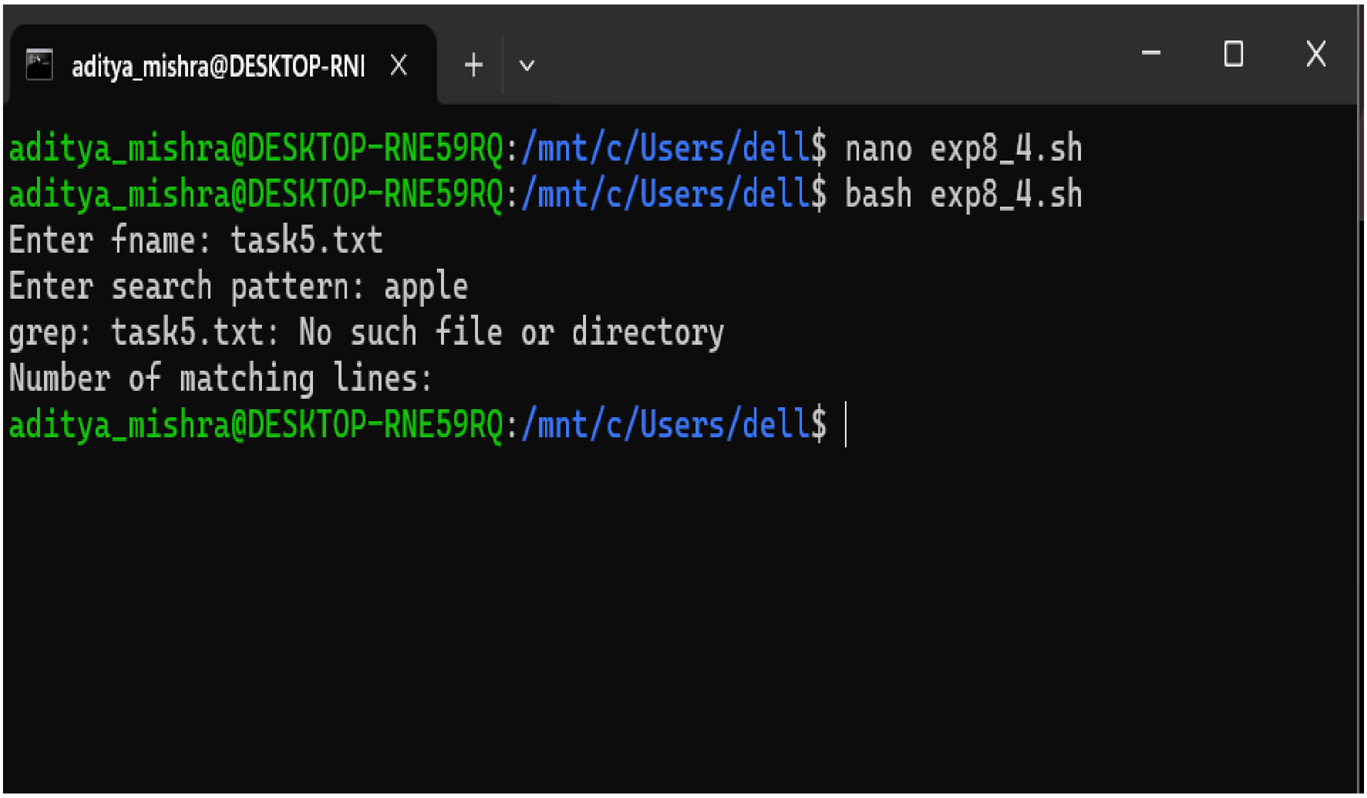
Command(s):

```
#!/bin/bash

read -p "Enter fname: " file
read -p "Enter search pattern: " pattern

count=$(grep -c "$pattern" "$file")
echo "Number of matching lines: $count"
```

Output:



```
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ nano exp8_4.sh
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$ bash exp8_4.sh
Enter fname: task5.txt
Enter search pattern: apple
grep: task5.txt: No such file or directory
Number of matching lines:
aditya_mishra@DESKTOP-RNE59RQ:/mnt/c/Users/dell$
```

Result

- Learned to manage background and foreground jobs.
- Compared file contents using shell scripting.
- Counted user processes and monitored memory usage automatically.
- Practiced searching patterns and counting occurrences in files.

Challenges Faced & Learning Outcomes

- Challenge 1: Correctly identifying job numbers for `fg` and `kill`. Solved by carefully checking `jobs` output.
- Challenge 2: Infinite loops in memory monitoring script. Fixed by planning termination conditions or using `Ctrl+C`.

Learning:

- Gained hands-on experience with job control commands.
- Learned file comparison techniques using `cmp`.
- Learned process counting and memory monitoring.
- Practiced pattern searching and line counting with `grep`.

Conclusion

This experiment provided practical experience with shell scripting for **jobs, file comparison, process monitoring, memory logging, and pattern searching**, which are essential skills for Linux system administration and automation.