

# **Experiment 8: Shell Programming**

**Name: Biswanbandya Mohanty Roll No.: 590029274 Date: 2025-11-1**

## **Aim:**

- To extend shell programming concepts by using conditional statements, advanced scripting constructs, and command-line arguments.
- To practice writing scripts that perform decision-making and parameter handling.

## **Requirements**

- A Linux system with bash shell.
- Text editor and permission to create/execute shell scripts.

## **Theory**

Conditional execution in shell scripts allows branching logic using `if`, `elif`, `else`, and `case` statements. Scripts can accept command-line arguments using `$1`, `$2`, ... and `$@` for all arguments. Control flow constructs combined with user input and arguments allow dynamic and reusable scripts.

## **Procedure & Observations**

### **Exercise 1: Using if-else**

#### **Task Statement:**

Write a script to check whether a given number is positive, negative, or zero.

#### **Explanation:**

We used an `if-elif-else` construct to compare the number against 0.

## Command(s):

```
#!/bin/bash
num=$1
if [ $num -gt 0 ]; then
    echo "$num is positive"
elif [ $num -lt 0 ]; then
    echo "$num is negative"
else
    echo "$num is zero"
fi
```

## Output:

```
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ls
background.sh  countlines.sh  exp1.sh  exp4.sh  exp83.sh  fibonacci.sh  file.sh  gcdlcm.sh  interation.sh  numeric  pstneg.sh  sleep.sh  symbols2.sh  while1.sh
arithmetic.sh  Biswa  dttime.sh  exp2.sh  exp5.sh  exp84.sh  file2.sh  file.txt  greeting2.sh  kill.sh  palindrome.sh  reverse.sh  sorting.sh  symbols.sh  while2.sh
armstrong.sh  brkctn.sh  escape.sh  exp3.sh  exp82.sh  exp85.sh  file3.sh  for.sh  greeting.sh  monitor.sh  prime.sh  slayslay  sum.sh  vote.sh  x.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ./pstneg.sh 1
1 is positive
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ./pstneg.sh -1
-1 is negative
biswa@biswa-virtual-machine:~/Desktop/Linux lab$
```

## Exercise 2: Using case

### Task Statement:

Write a script that takes a character as input and classifies it as vowel, consonant, digit, or special character.

## **Explanation:**

The `case` statement provides pattern matching for multiple options.

## **Command(s):**

```
#!/bin/bash
ch=$1
case $ch in
[aeiouAEIOU]) echo "$ch is a vowel" ;;
[bcdfghjklmnpqrstuvwxyzBCDFGHJKLMNOPQRSTUVWXYZ]) echo "$ch is a consonant" ;;
[@-9]) echo "$ch is a digit" ;;
*) echo "$ch is a special character" ;;
esac
```

## **Output:**



## **Exercise 3: Command-line arguments**

### **Task Statement:**

Write a script that accepts filename(s) as arguments and prints the number of lines in each file.

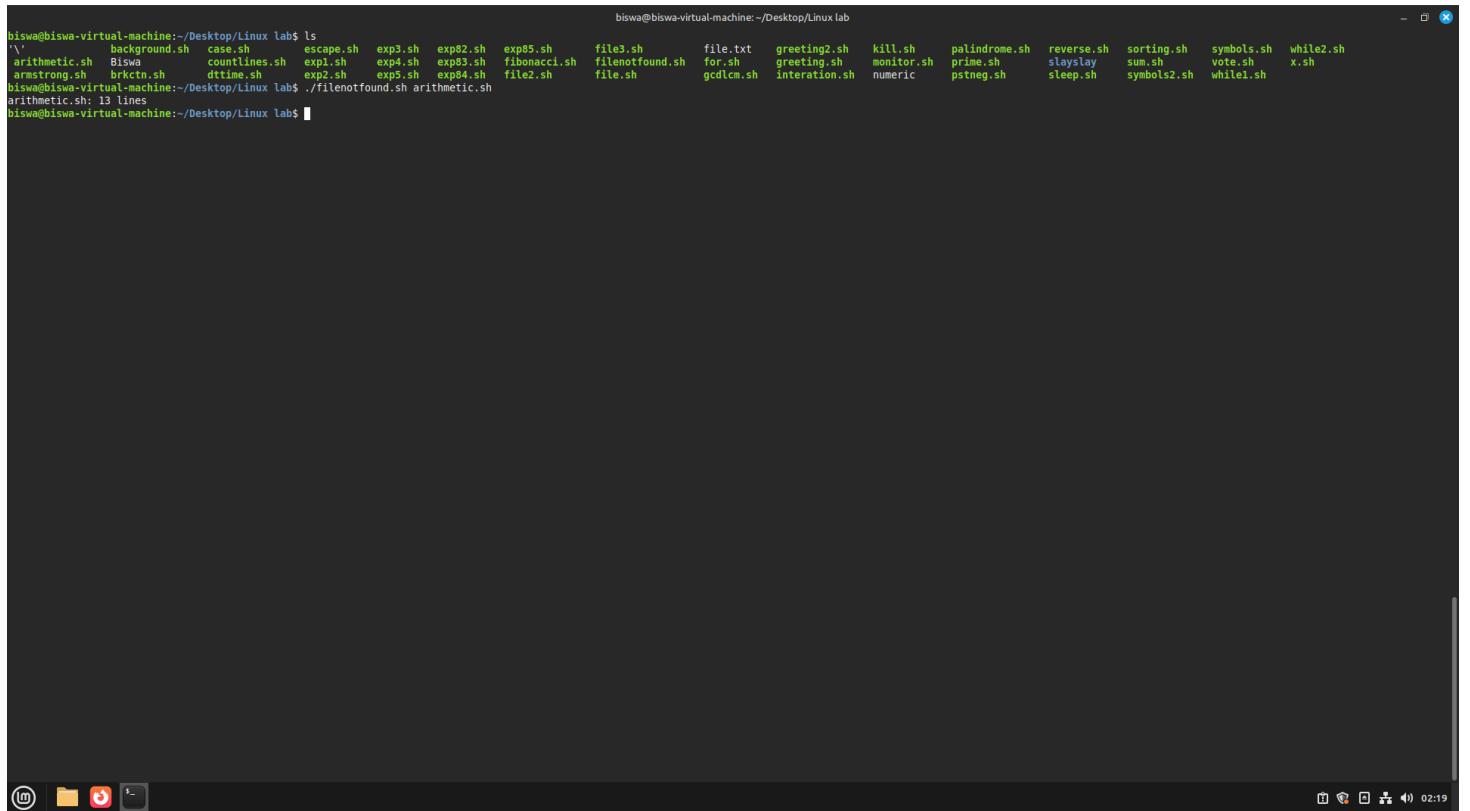
## **Explanation:**

Command-line arguments are accessed using `$@`. Looping through each argument allows file-wise operations.

## Command(s):

```
#!/bin/bash
for file in "$@"; do
    if [ -f "$file" ]; then
        echo "$file: $(wc -l < "$file") lines"
    else
        echo "$file not found"
    fi
done
```

## Output:



```
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ls
'.' background.sh case.sh escape.sh exp3.sh exp82.sh exp85.sh file3.sh file.txt greeting2.sh kill.sh palindrome.sh reverse.sh sorting.sh symbols.sh while2.sh
arithmetic.sh Biswa countlines.sh expli.sh exp4.sh exp83.sh fibonacci.sh filenotfound.sh for.sh greeting.sh monitor.sh prime.sh slayslay sum.sh vote.sh x.sh
armstrong.sh brkctn.sh dttime.sh exp2.sh exp5.sh exp84.sh file2.sh file.sh gcdlcm.sh interation.sh numeric pstrneg.sh sleep.sh symbols2.sh while1.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ./filenotfound.sh arithmetic.sh
arithmetic.sh: 13 lines
biswa@biswa-virtual-machine:~/Desktop/Linux lab$
```

## Exercise 4: Nested conditionals

### Task Statement:

Write a script to check if a year is a leap year.

## Explanation:

A leap year is divisible by 4, but if divisible by 100 it must also be divisible by 400.

## Command(s):

```
#!/bin/bash
year=$1
if (( year % 400 == 0 )); then
    echo "$year is a leap year"
elif (( year % 100 == 0 )); then
    echo "$year is not a leap year"
elif (( year % 4 == 0 )); then
    echo "$year is a leap year"
else
    echo "$year is not a leap year"
fi
```

## **Output:**

```
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ls
`\'`      background.sh  case.sh    escape.sh  exp3.sh   exp82.sh  exp85.sh   file3.sh    file.txt   greeting2.sh  kill.sh    numeric    pstneg.sh  sleep.sh  symbols2.sh  while1.sh
arithmetic.sh Biswa    countlines.sh  exp1.sh   exp4.sh   exp83.sh  fibonacci.sh filenotfound.sh for.sh   greeting.sh  leap.sh   palindrome.sh reverse.sh  sorting.sh symbols.sh   while2.sh
armstrong.sh  rkctn.sh dttime.sh   exp2.sh   exp5.sh   exp84.sh  file2.sh    file.sh    gcdlcm.sh  interation.sh monitor.sh prime.sh   slayslay   sum.sh    vote.sh   x.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ./leap.sh 2025
2025 is not a leap year
biswa@biswa-Virtual-machine:~/Desktop/Linux lab$
```

# Task 1

## Task Statement:

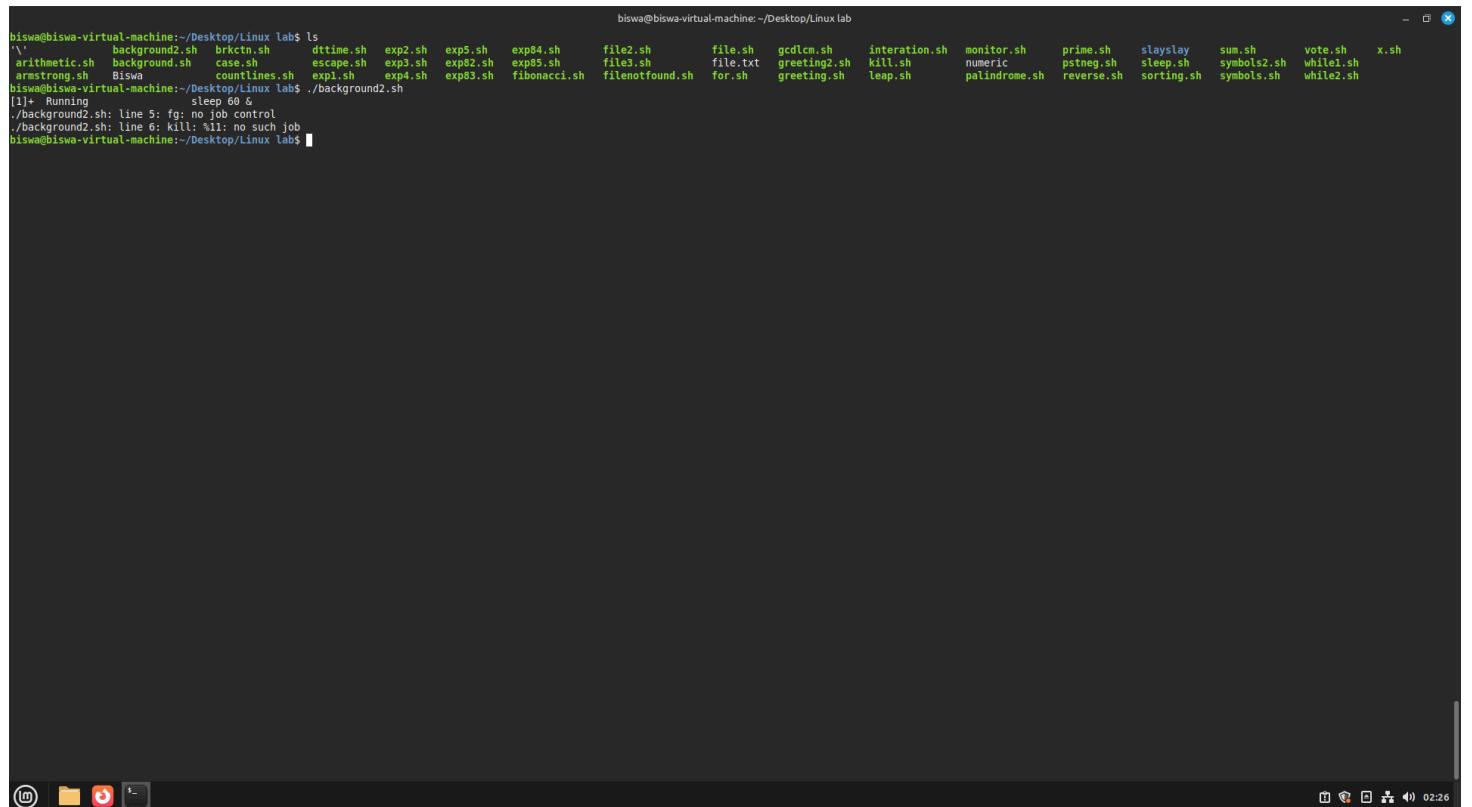
Write a script that starts a background job (e.g., `sleep 60`), lists all jobs, brings the job to the foreground, and then terminates it.

## Command(s):

```
#!/bin/bash
```

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

## Output:



The screenshot shows a terminal window with the following session:

```
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ls
'-'      background2.sh  brkctn.sh    dtime.sh   exp2.sh   exp5.sh   exp84.sh   file2.sh     file.sh   gcdlcm.sh   interation.sh   monitor.sh   prime.sh   slayslay   sum.sh     vote.sh   x.sh
arithmetic.sh  background.sh  case.sh    escape.sh  exp3.sh   exp82.sh  exp85.sh   file3.sh     file.txt  greeting2.sh  kill.sh    numeric    pstneg.sh  sleep.sh   symbols2.sh  white1.sh
armstrong.sh  Biswa       countlines.sh expl1.sh  exp4.sh   exp83.sh  fibonacci.sh filenotfound.sh for.sh   greeting.sh  leap.sh   palindrome.sh reverse.sh  sorting.sh  symbols.sh   while2.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ./background2.sh
[1]+  Running                 sleep 60 &
./background2.sh: line 5: fg: no job control
./background2.sh: line 6: kill: %1: no such job
biswa@biswa-virtual-machine:~/Desktop/Linux lab$
```

The terminal window has a dark theme. The bottom status bar shows icons for file operations and system status, with the time 02:26 displayed.

# Task 2

## Task Statement:

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

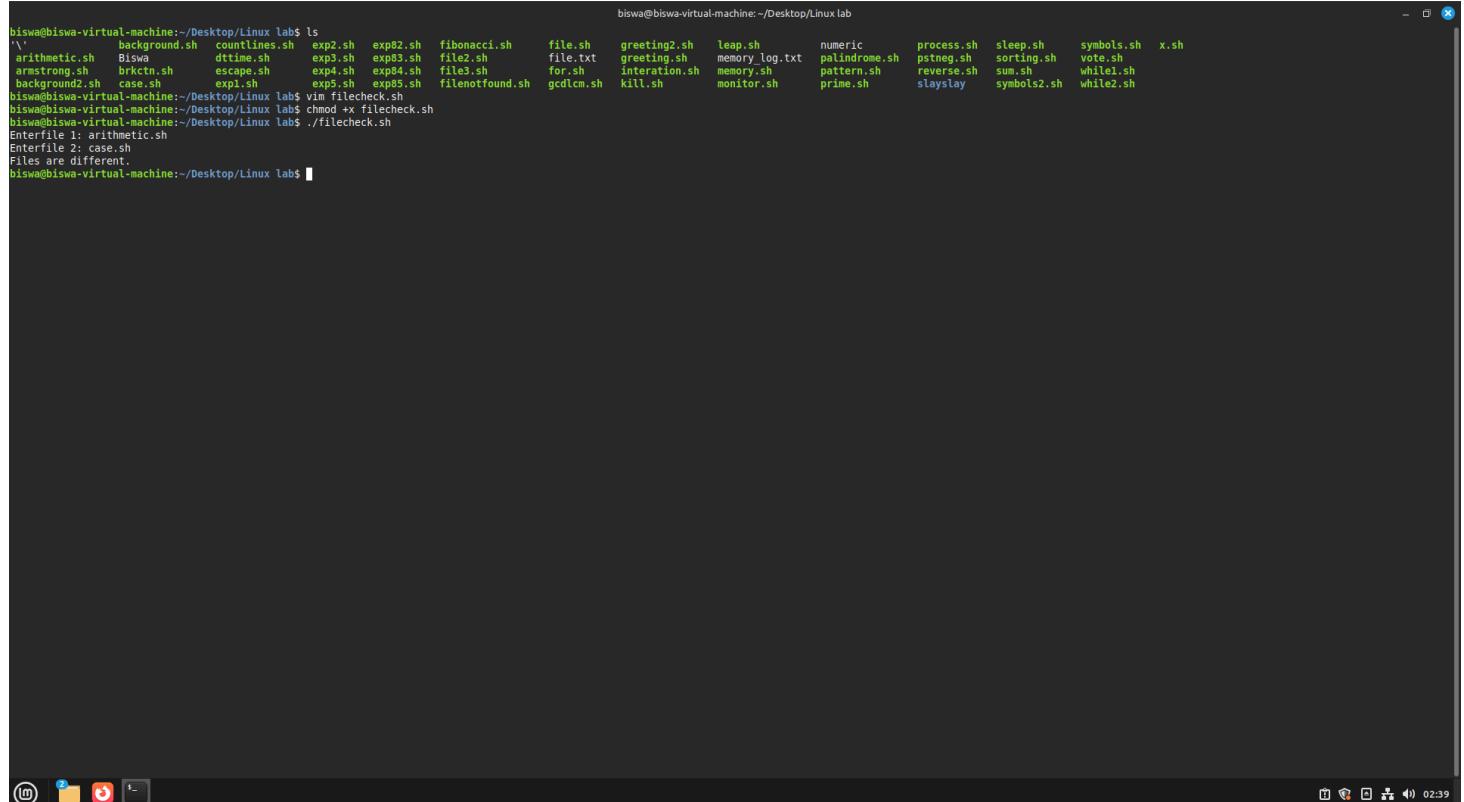
## Command(s):

```
#!/bin/bash

read -p "Enter file 1: " file1
read -p "Enter file 2: " file2

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

## Output:



The screenshot shows a terminal window titled 'biswa@biswa-virtual-machine:~/Desktop/Linux lab'. The user has run the command 'ls' to list files in the directory. Then, they run 'chmod +x filecheck.sh' to make the script executable. Finally, they run the script with './filecheck.sh' and provide two arguments: 'arithmetic.sh' and 'case.sh'. The output shows that the files are different.

```
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ls
background.sh  countlines.sh  exp2.sh  fibonacci.sh  file.sh  greeting2.sh  leap.sh  numeric.sh  process.sh  sleep.sh  symbols.sh  x.sh
arithmetic.sh  Biswa  dtime.sh  exp3.sh  file2.sh  file.txt  greeting.sh  memory_log.txt  palindrome.sh  pstneg.sh  sorting.sh  vote.sh
armstrong.sh  brkcnt.sh  escape.sh  exp4.sh  exp84.sh  file1.sh  for.sh  interation.sh  memory.sh  pattern.sh  reverse.sh  sum.sh  white1.sh
background2.sh  case.sh  exp1.sh  exp5.sh  exp85.sh  filenotfound.sh  gcdlcm.sh  kill.sh  monitor.sh  prime.sh  slayslay  symbols2.sh  white2.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ chmod +x filecheck.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ./filecheck.sh
Enter file 1: arithmetic.sh
Enter file 2: case.sh
Files are different.
biswa@biswa-virtual-machine:~/Desktop/Linux lab$
```

# Task 3

## Task Statement:

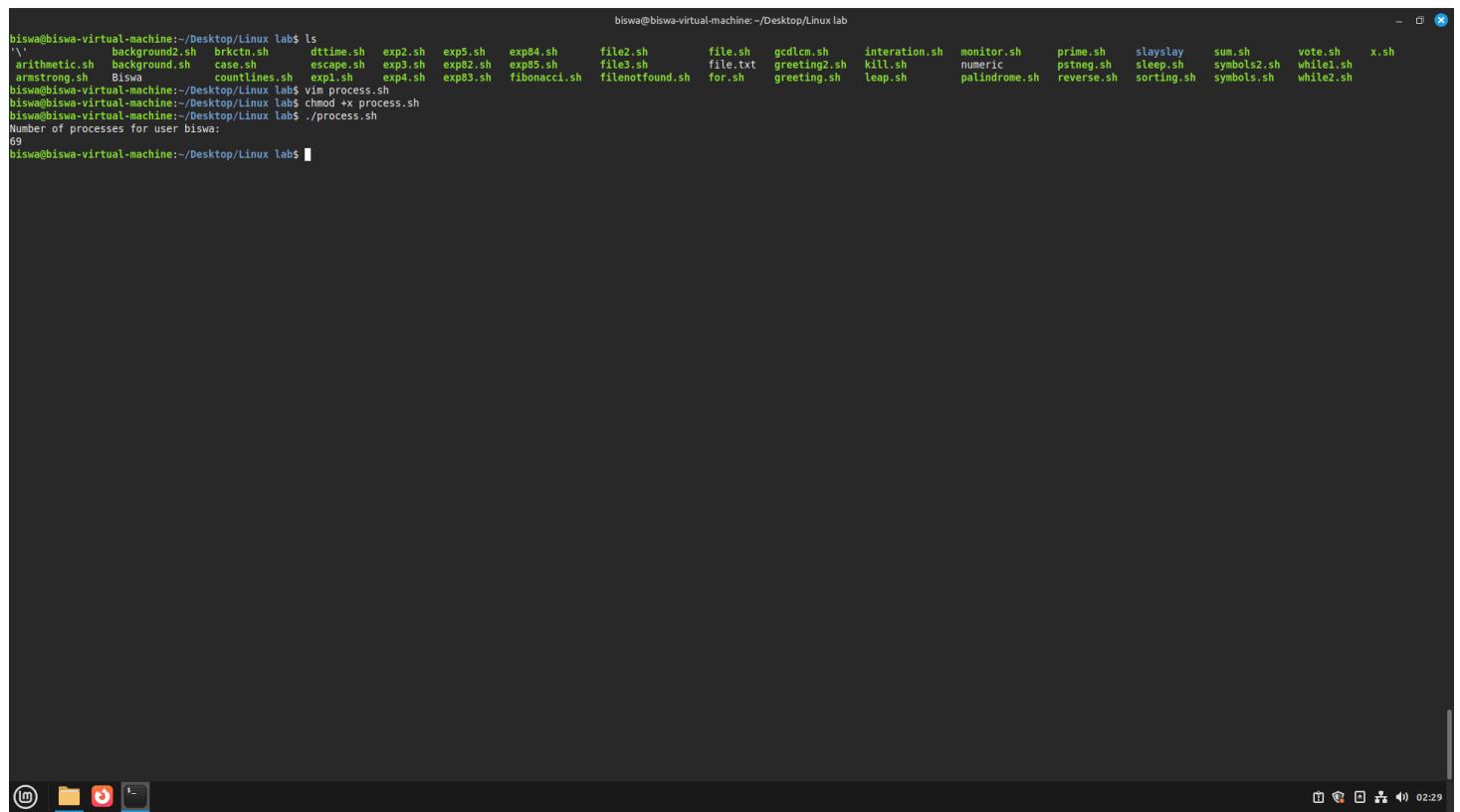
Write a script that counts the number of processes currently being run by your user.

## Command(s):

```
#!/bin/bash

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

## Output:



The screenshot shows a terminal window titled 'biswa@biswa-virtual-machine:~/Desktop/Linux lab'. The user has run the command 'ls' to list files in the current directory, which contains numerous shell scripts. Then, they ran the script 'process.sh' (which presumably contains the command 'ps -u \$USER | wc -l'). The output of this script is 'Number of processes for user biswa: 69', indicating that there are 69 processes currently running under the 'biswa' user. The terminal window also shows standard Linux desktop icons at the bottom.

```
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ls
''                                background2.sh  brkctn.sh    dttime.sh  exp2.sh  exp5.sh  exp84.sh   file2.sh      file.sh    gcdlcm.sh  interation.sh  monitor.sh  prime.sh  slayslay  sum.sh     vote.sh   x.sh
arithmetic.sh  background.sh  case.sh    escape.sh  exp3.sh  exp82.sh  exp85.sh   file3.sh      file.txt  greeting2.sh kill.sh    numeric   palindrome.sh pstneg.sh  sleep.sh  symbols.sh while1.sh
armstrong.sh   Biswa       countlines.sh  exp1.sh   exp4.sh  exp83.sh  fibonacci.sh filenotfound.sh for.sh   greeting.sh  leap.sh   numeric   reverse.sh sorting.sh symbols2.sh while2.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ vim process.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ chmod +x process.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ./process.sh
Number of processes for user biswa:
69
biswa@biswa-virtual-machine:~/Desktop/Linux lab$
```

# Task 4

## Task Statement:

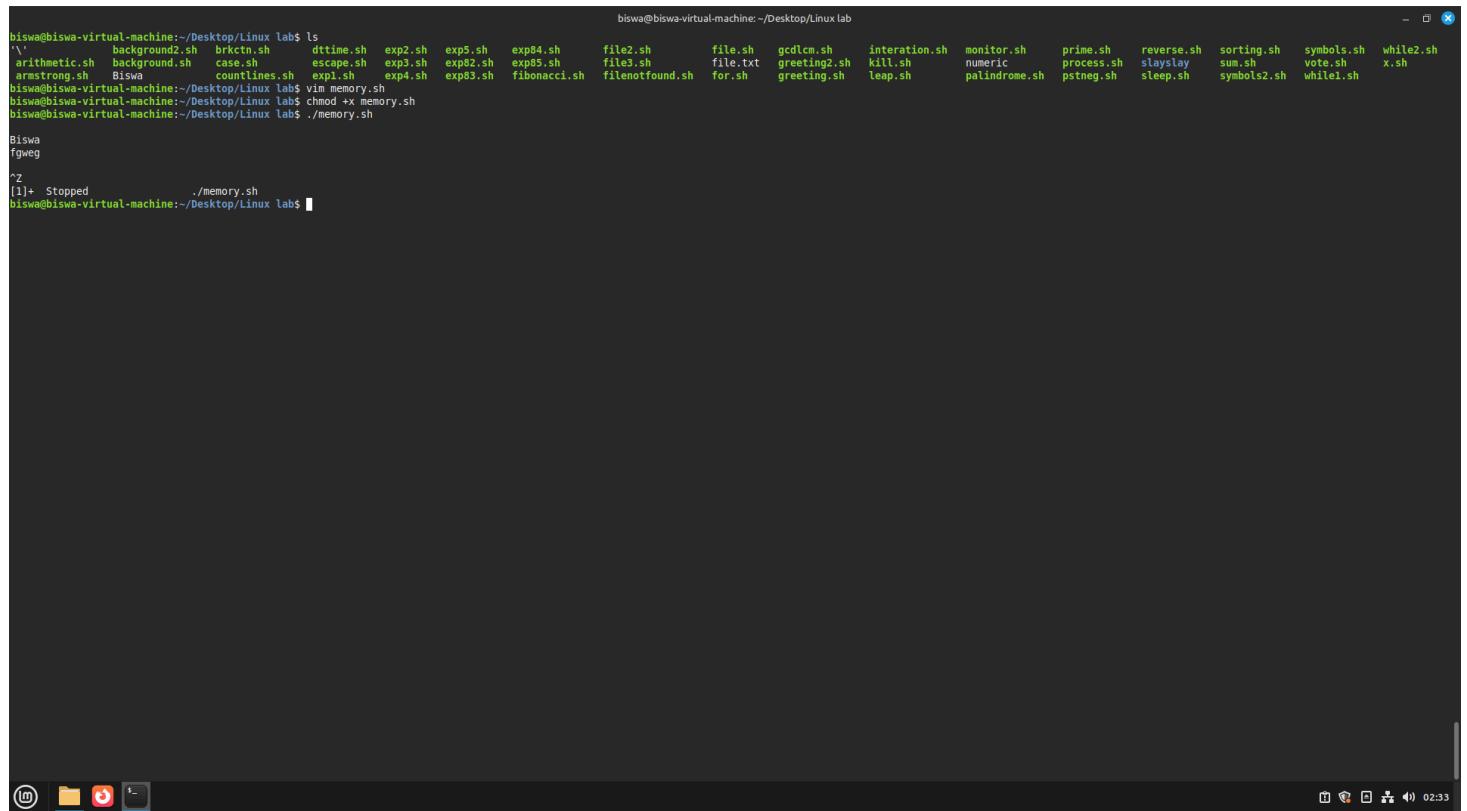
Develop a script that monitors memory usage every 5 seconds and logs it into a file.

## 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:



The screenshot shows a terminal window with the following session:

```
biswa@biswa-virtual-machine:~/Desktop/Linux labs$ ls
`-' background2.sh brktn.sh dtime.sh exp2.sh exp5.sh exp84.sh file2.sh file.sh gcdlcm.sh interation.sh monitor.sh prime.sh reverse.sh sorting.sh symbols.sh while2.sh
arithmetic.sh background.sh case.sh escape.sh exp3.sh exp82.sh exp85.sh file3.sh file.txt greeting2.sh kill.sh numeric process.sh slayslay sum.sh symbols2.sh vote.sh x.sh
armstrong.sh Biswa countlines.sh expl1.sh exp4.sh exp83.sh fibonacci.sh filenotfound.sh for.sh greeting.sh leap.sh palindrome.sh pstneg.sh sleep.sh symbols2.sh while1.sh
biswa@biswa-virtual-machine:~/Desktop/Linux labs$ vim memory.sh
biswa@biswa-virtual-machine:~/Desktop/Linux labs$ chmod +x memory.sh
biswa@biswa-virtual-machine:~/Desktop/Linux labs$ ./memory.sh
Biswa
fgweg
^Z
[1]+  Stopped                  ./memory.sh
biswa@biswa-virtual-machine:~/Desktop/Linux labs$
```

The terminal shows the user navigating through a directory of scripts, creating a new file 'memory.sh', giving it execute permissions, and then running it. The script itself is not visible in the terminal window, but its execution is indicated by the command and its output.

# Task 5

## Task Statement:

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

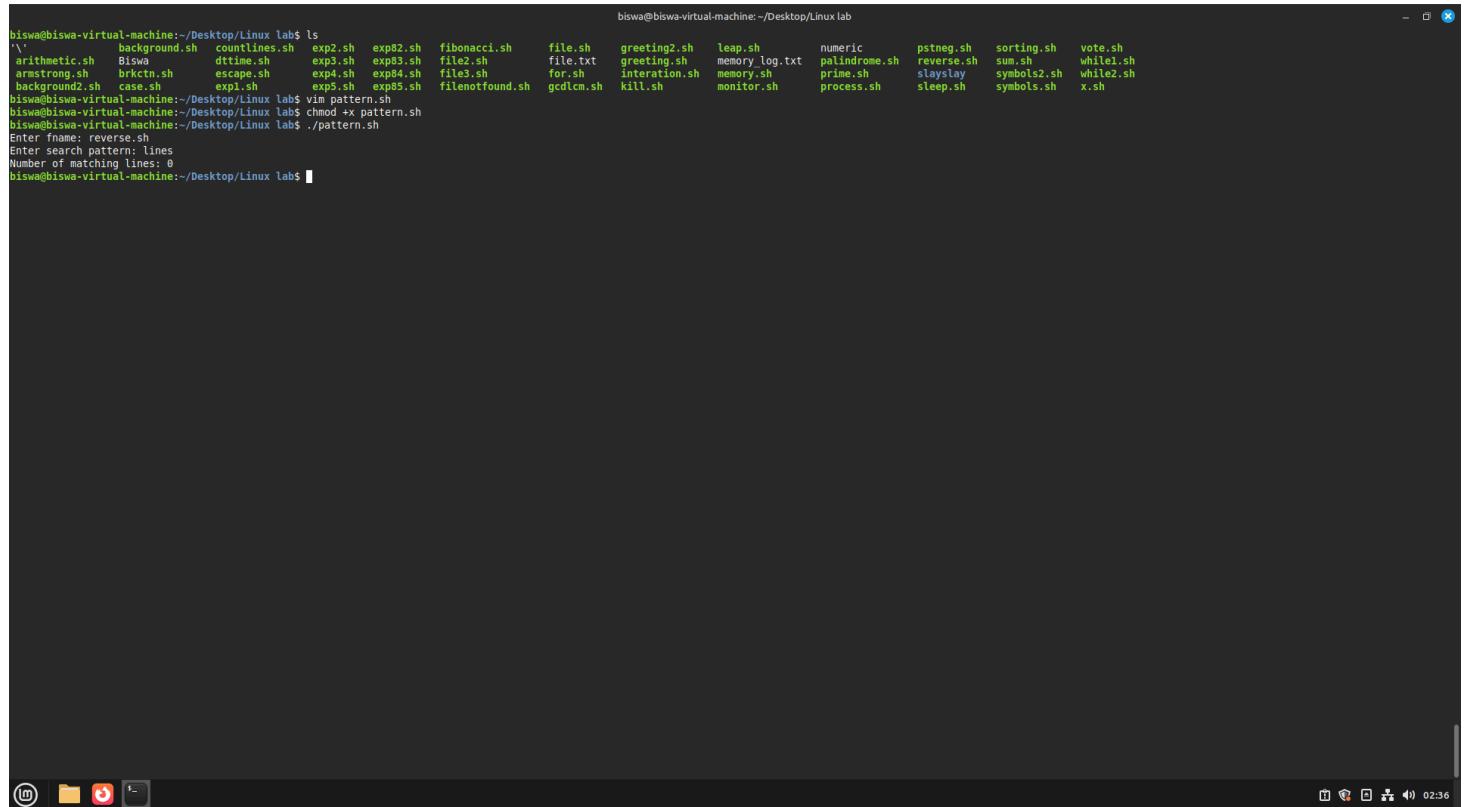
## Command(s):

```
#!/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:



The screenshot shows a terminal window titled 'biswa@biswa-virtual-machine:~/Desktop/Linux lab'. The user runs 'ls' to list files, then 'vim pattern.sh' to edit a search pattern. They enter 'reverse.sh' as the file name and 'lines' as the search pattern. The script outputs 'Number of matching lines: 0'.

```
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ls
backgroun...
arithmetic.sh  Biswa  countlines.sh  exp2.sh  fibonacci.sh  file.sh  greeting2.sh  leap.sh  numeric  pstneg.sh  sorting.sh  vote.sh
arithmeti...
Biswa        dttime.sh  exp3.sh  exp82.sh  fibonacci.sh  file.txt  greeting.sh  leap.sh  numeric  pstneg.sh  sorting.sh  vote.sh
armstrong.sh  brkctn.sh  escape.sh  exp4.sh  exp83.sh  file2.sh  for.sh  interation.sh  memory.log.txt  palindrome.sh  reverse.sh  sum.sh  while1.sh
background2.sh case.sh  expl.sh  exp5.sh  exp84.sh  file3.sh  for3.sh  interation.sh  memory.log.txt  palindrome.sh  reverse.sh  sum.sh  while1.sh
background2...
case.sh      expl...
exp5.sh     exp85.sh  filenotfound.sh  gcdlcm.sh  kill.sh  monitor.sh  process.sh  sleep.sh  symbols2.sh  while2.sh
background2...
case...
exp5...
filenotfound...
gcdlcm...
kill...
monitor...
process...
sleep...
symbols2...
while2...
x.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ vim pattern.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ chmod +x pattern.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ./pattern.sh
Enter fname: reverse.sh
Enter search pattern: lines
Number of matching lines: 0
biswa@biswa-virtual-machine:~/Desktop/Linux lab$
```

# **Result**

- Implemented conditional statements ( `if-else` , `case` ) in shell scripts.
- Practiced handling command-line arguments and nested conditions.
- Wrote reusable and flexible shell scripts.

## **Challenges Faced & Learning Outcomes**

- Challenge 1: Forgetting to quote variables in conditions — resolved by using `"$var"` to avoid word splitting.
- Challenge 2: Pattern matching in `case` — practiced with multiple examples.
- challenge 3: Could not run experiment 2 perfectly.

## **Learning:**

- Learned practical use of branching and decision-making in shell scripting.
- Understood command-line argument handling for automation.

## **Conclusion**

This experiment extended shell programming by introducing decision-making and parameter handling. The scripts demonstrate the flexibility of shell programming for different use cases.