Experiment 6: Shell Loops

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

Aim:

- To understand and implement shell loops (`for` , `while` , `until`) in Bash.
- To practice loop control constructs (`break` , `continue`) and loop-based file processing.

Requirements

- A Linux system with bash shell.
- A text editor (nano, vim) and permission to create and execute shell scripts.

Theory

Loops allow repeated execution of commands until a condition is met. Common loop constructs in Bash include `for` (iterate over items), `while` (repeat while condition true), and `until` (repeat until condition becomes true). Loop control statements like `break` and `continue` change the flow inside loops. Loops are essential for automating repetitive tasks such as processing multiple files, generating sequences, and collecting user input.

Procedure & Observations

Exercise 1: Simple `for` loop

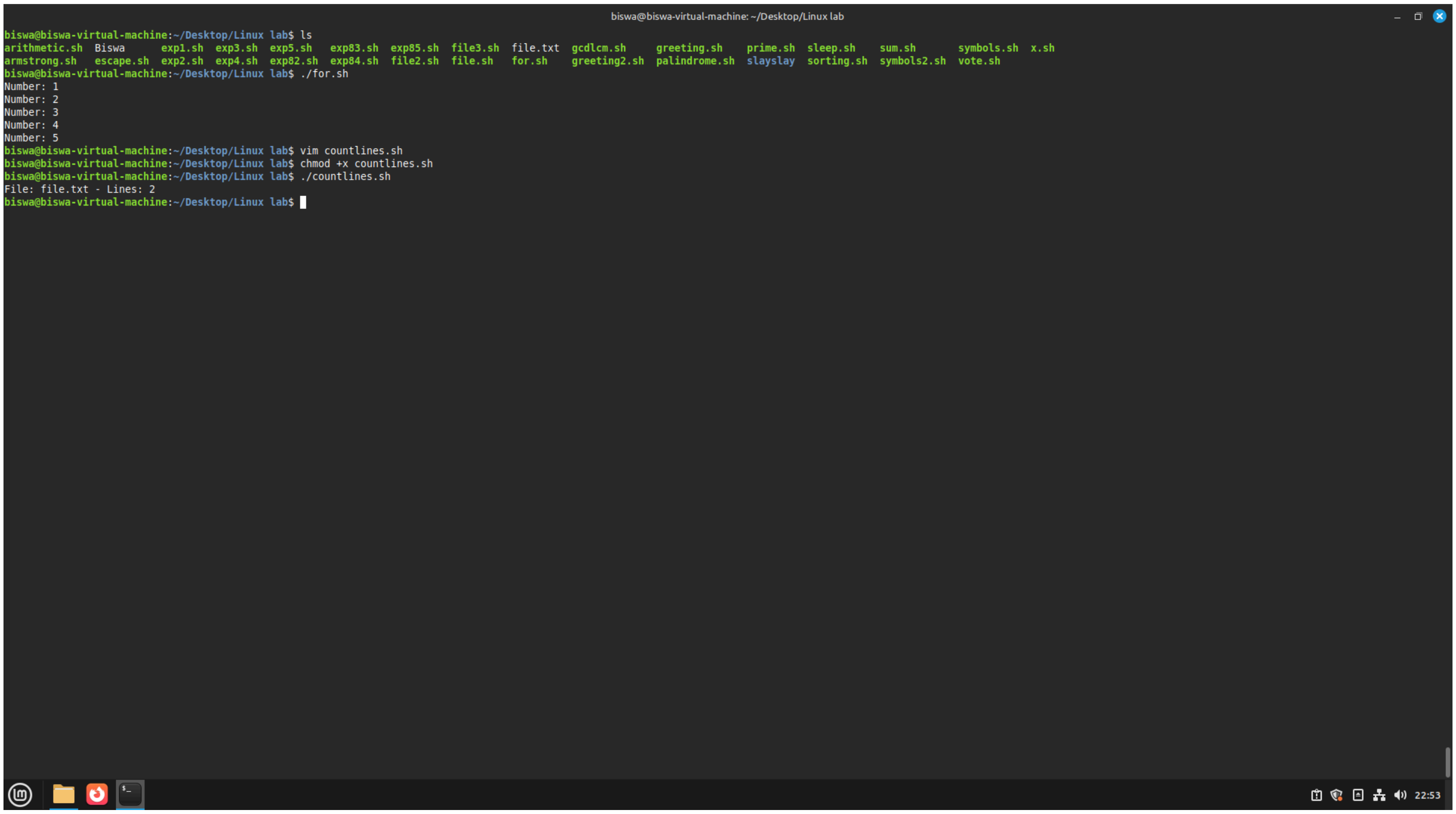
Task Statement:

Write a `for` loop that prints numbers 1 to 5.

Command(s):

```
for i in 1 2 3 4 5; do
  echo "Number: $i"
done
```

Output:



```
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ls
arithmetic.sh  Biswa      exp1.sh  exp3.sh  exp5.sh   exp83.sh  exp85.sh  file3.sh  file.txt  gcdlcm.sh     greeting.sh    prime.sh  sleep.sh    sum.sh       symbols.sh  x.sh
armstrong.sh   escape.sh  exp2.sh  exp4.sh  exp82.sh  exp84.sh  file2.sh  file.sh   for.sh    greeting2.sh  palindrome.sh  slayslay  sorting.sh  symbols2.sh  vote.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ./for.sh
Number: 1
Number: 2
Number: 3
Number: 4
Number: 5
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ vim countlines.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ chmod +x countlines.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ./countlines.sh
File: file.txt - Lines: 2
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ 
```

Exercise 2: `for` loop over files

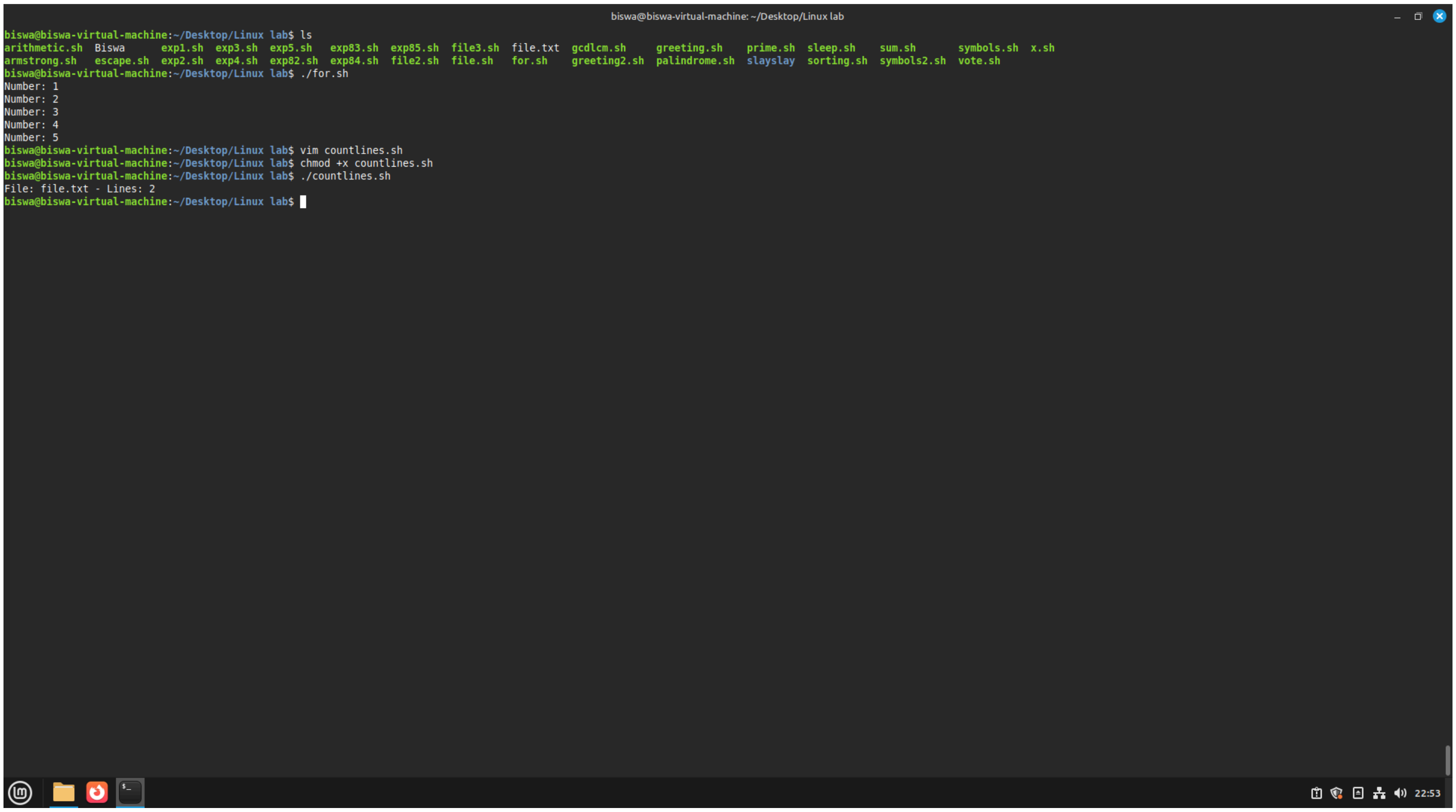
Task Statement:

Process all `.txt` files in a directory and count lines in each.

Command(s):

```
for f in *.txt; do
  echo "File: $f - Lines: $(wc -l < "$f")"
done
```

Output:



```
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ls
arithmetic.sh  Biswa      exp1.sh  exp3.sh  exp5.sh   exp83.sh  exp85.sh  file3.sh  file.txt  gcdlcm.sh     greeting.sh    prime.sh  sleep.sh    sum.sh       symbols.sh  x.sh
armstrong.sh   escape.sh  exp2.sh  exp4.sh  exp82.sh  exp84.sh  file2.sh  file.sh   for.sh    greeting2.sh  palindrome.sh  slayslay  sorting.sh  symbols2.sh  vote.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ./for.sh
Number: 1
Number: 2
Number: 3
Number: 4
Number: 5
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ vim countlines.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ chmod +x countlines.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ./countlines.sh
File: file.txt - Lines: 2
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ 
```

Exercise 3: C-style `for` loop

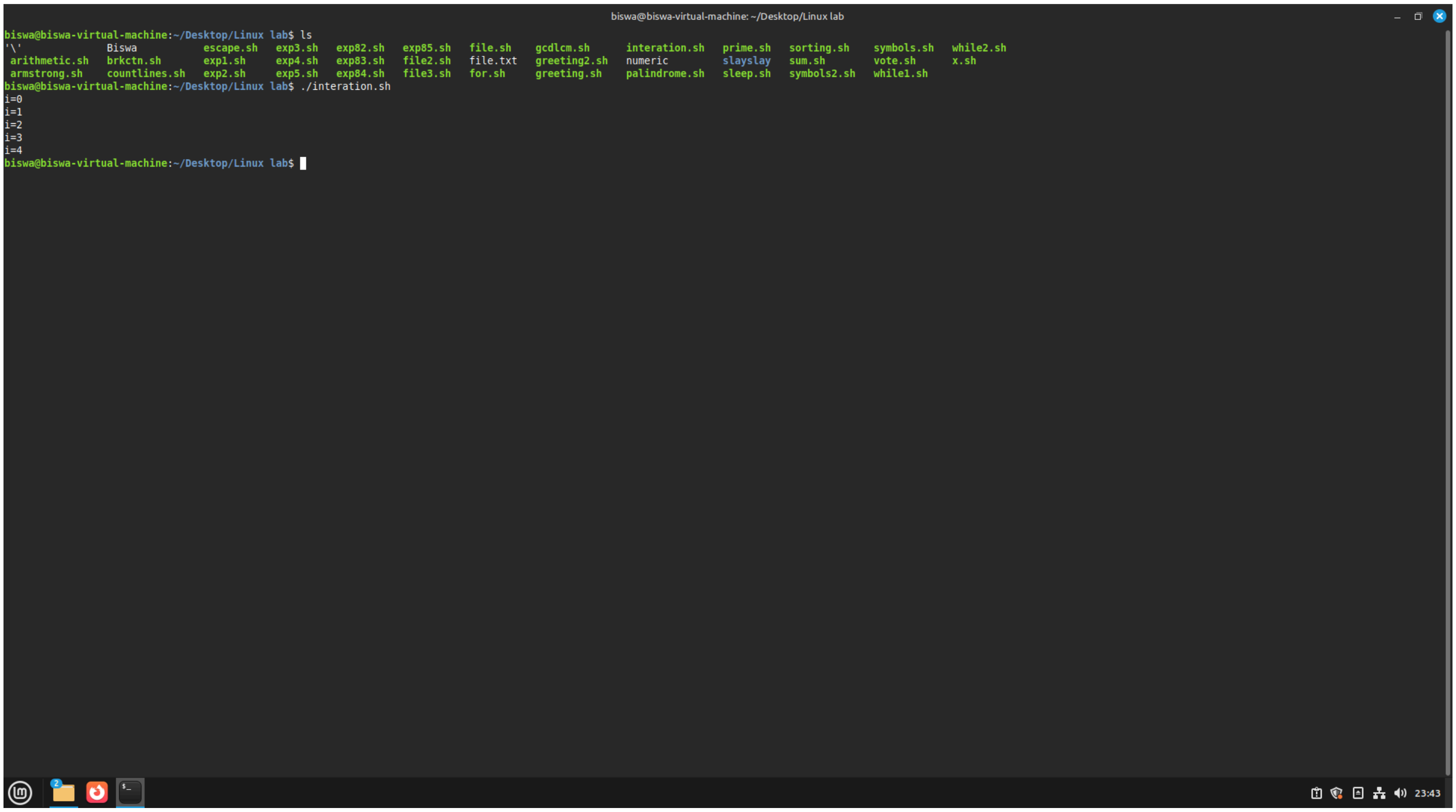
Task Statement:

Use arithmetic C-style loop for numeric iteration.

Command(s):

```
for ((i=0;i<5;i++)); do
  echo "i=$i"
done
```

Output:



```
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ls
'\'             Biswa          escape.sh  exp3.sh  exp82.sh  exp85.sh  file.sh   gcdlcm.sh     interation.sh  prime.sh  sorting.sh   symbols.sh   while2.sh
 arithmetic.sh  brkctn.sh      exp1.sh    exp4.sh  exp83.sh  file2.sh  file.txt  greeting2.sh  numeric        slayslay  sum.sh       vote.sh      x.sh
 armstrong.sh   countlines.sh  exp2.sh    exp5.sh  exp84.sh  file3.sh  for.sh    greeting.sh   palindrome.sh  sleep.sh  symbols2.sh  while1.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ./interation.sh
i=0
i=1
i=2
i=3
i=4
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ 
```

Exercise 4: `while` loop and reading input

Task Statement:

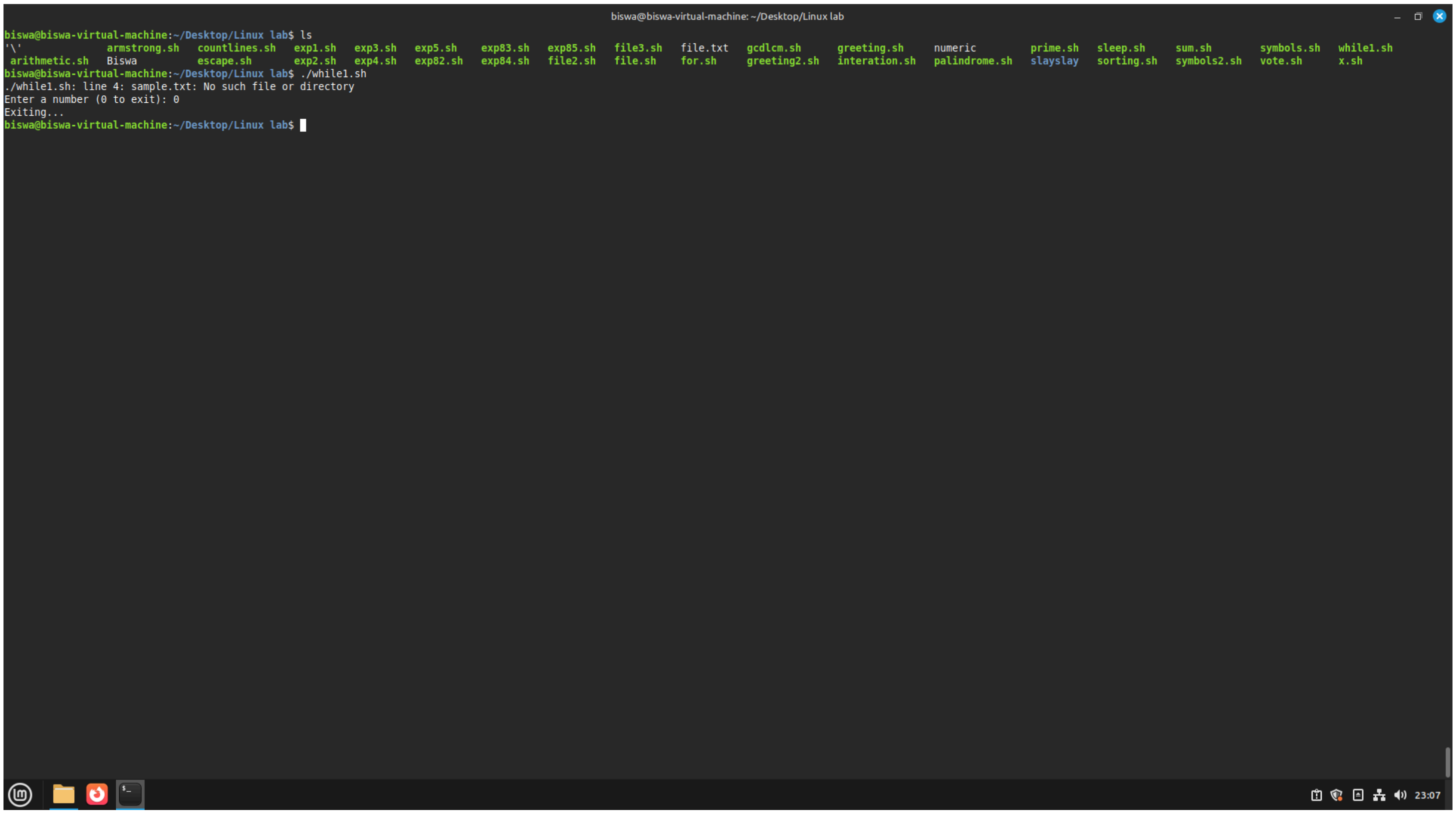
Write a `while` loop that reads lines from a file or from user input.

Command(s):

```
while read -r line; do
  echo "Line: $line"
done < sample.txt

while true; do
  read -p "Enter a number (0 to exit): " n
  if [[ $n -eq 0 ]]; then
    echo "Exiting..."; break
  fi
  echo "You entered: $n"
done
```

Output:



biswa@biswa-virtual-machine: ~/Desktop/Linux lab

```
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ls
'\'              armstrong.sh   countlines.sh   exp1.sh   exp3.sh   exp5.sh    exp83.sh   exp85.sh   file3.sh   file.txt   gcdlcm.sh      greeting.sh     numeric         prime.sh   sleep.sh     sum.sh        symbols.sh   while1.sh
 arithmetic.sh   Biswa          escape.sh       exp2.sh   exp4.sh   exp82.sh   exp84.sh   file2.sh   file.sh    for.sh     greeting2.sh   interation.sh   palindrome.sh   slayslay   sorting.sh   symbols2.sh   vote.sh      x.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ./while1.sh
./while1.sh: line 4: sample.txt: No such file or directory
Enter a number (0 to exit): 0
Exiting...
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ 
```

Exercise 5: `until` loop

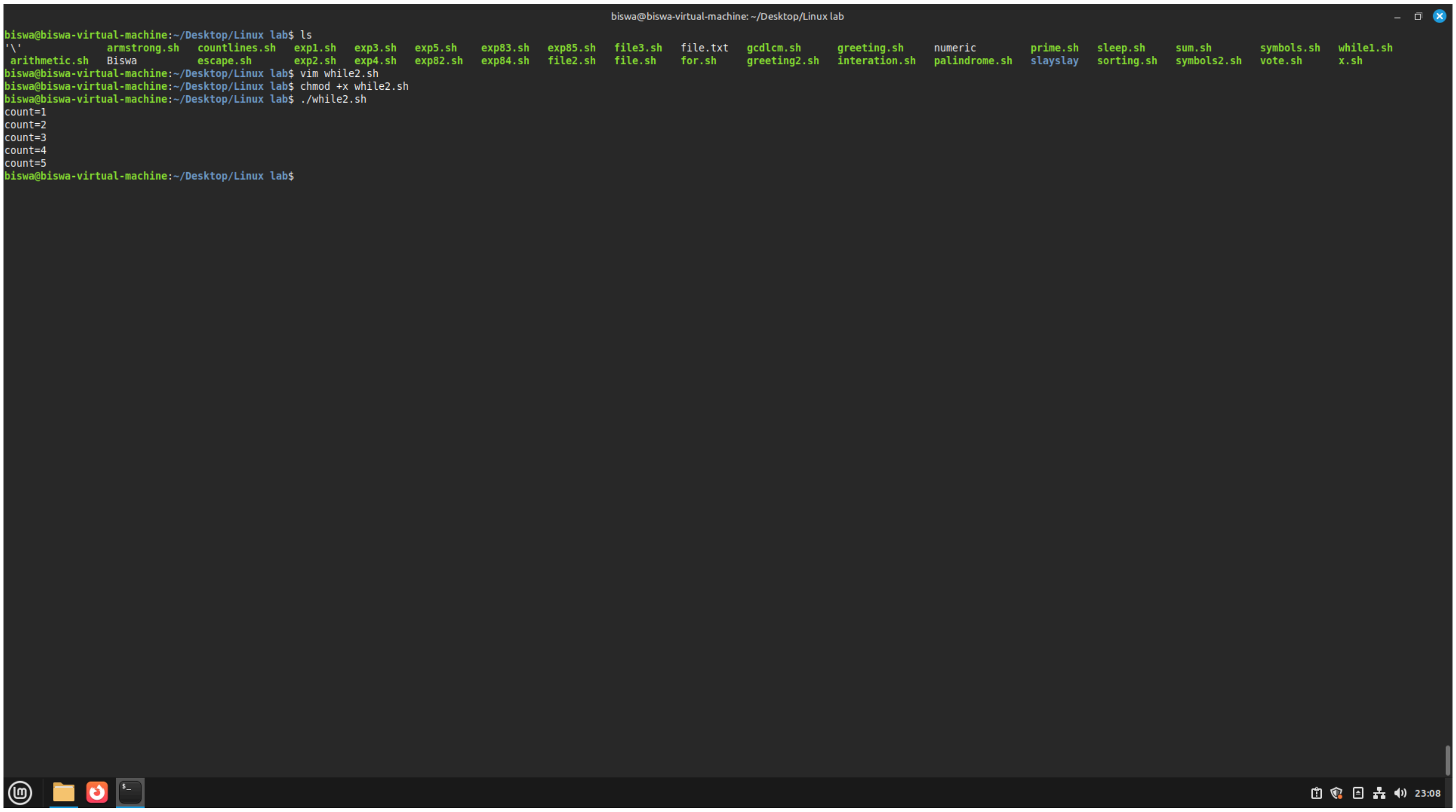
Task Statement:

Use an `until` loop to run until a condition becomes true.

Command(s):

```
count=1
until [ $count -gt 5 ]; do
  echo "count=$count"
  ((count++))
done
```

Output:



biswa@biswa-virtual-machine: ~/Desktop/Linux lab

```
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ls
'\'              armstrong.sh   countlines.sh   exp1.sh   exp3.sh   exp5.sh    exp83.sh   exp85.sh   file3.sh   file.txt   gcdlcm.sh      greeting.sh      numeric         prime.sh   sleep.sh     sum.sh        symbols.sh   while1.sh
 arithmetic.sh   Biswa          escape.sh       exp2.sh   exp4.sh   exp82.sh   exp84.sh   file2.sh   file.sh    for.sh     greeting2.sh   interation.sh    palindrome.sh   slayslay   sorting.sh   symbols2.sh   vote.sh      x.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ vim while2.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ chmod +x while2.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ./while2.sh
count=1
count=2
count=3
count=4
count=5
biswa@biswa-virtual-machine:~/Desktop/Linux lab$
```

Exercise 6: `break` and `continue`

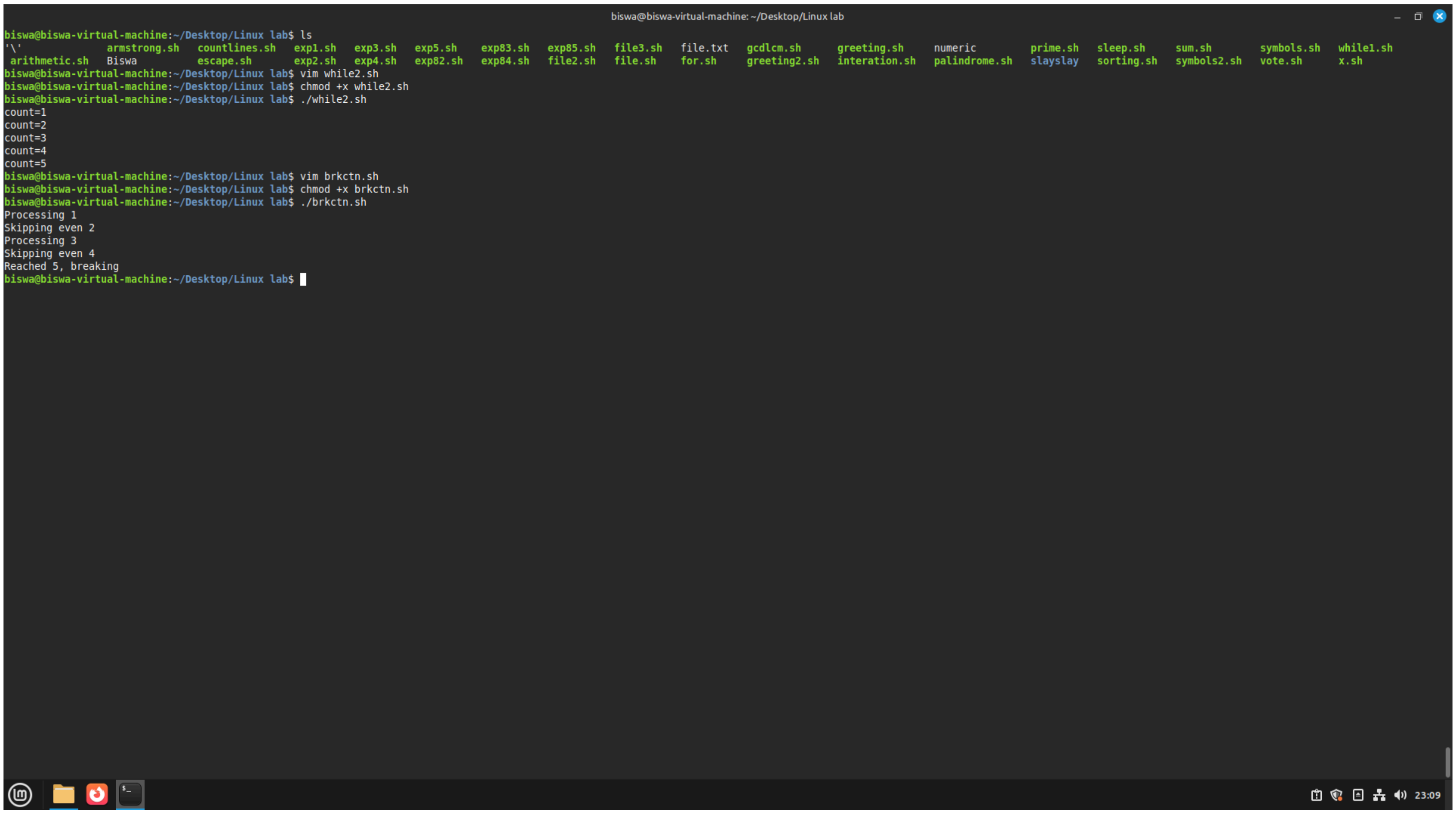
Task Statement:

Demonstrate `break` and `continue` inside a loop.

Command(s):

```
for i in {1..10}; do
  if [[ $i -eq 5 ]]; then
    echo "Reached 5, breaking"; break
  fi
  if (( i % 2 == 0 )); then
    echo "Skipping even $i"; continue
  fi
  echo "Processing $i"
done
```

Output:



```
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ls
'\'             armstrong.sh   countlines.sh   exp1.sh   exp3.sh   exp5.sh    exp83.sh   exp85.sh   file3.sh   file.txt   gcdlcm.sh      greeting.sh     numeric         prime.sh   sleep.sh     sum.sh         symbols.sh   while1.sh
 arithmetic.sh   Biswa          escape.sh       exp2.sh   exp4.sh   exp82.sh   exp84.sh   file2.sh   file.sh    for.sh     greeting2.sh   interation.sh   palindrome.sh   slayslay   sorting.sh   symbols2.sh   vote.sh      x.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ vim while2.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ chmod +x while2.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ./while2.sh
count=1
count=2
count=3
count=4
count=5
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ vim brkctn.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ chmod +x brkctn.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ./brkctn.sh
Processing 1
Skipping even 2
Processing 3
Skipping even 4
Reached 5, breaking
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ 
```

Exercise 7: Nested loops

Task Statement:

Create nested loops to generate a multiplication table.

Command(s):

```
for i in {1..3}; do
  for j in {1..3}; do
    echo -n "$((i*j)) "
  done
  echo
done
```

Output:

exp6_nested

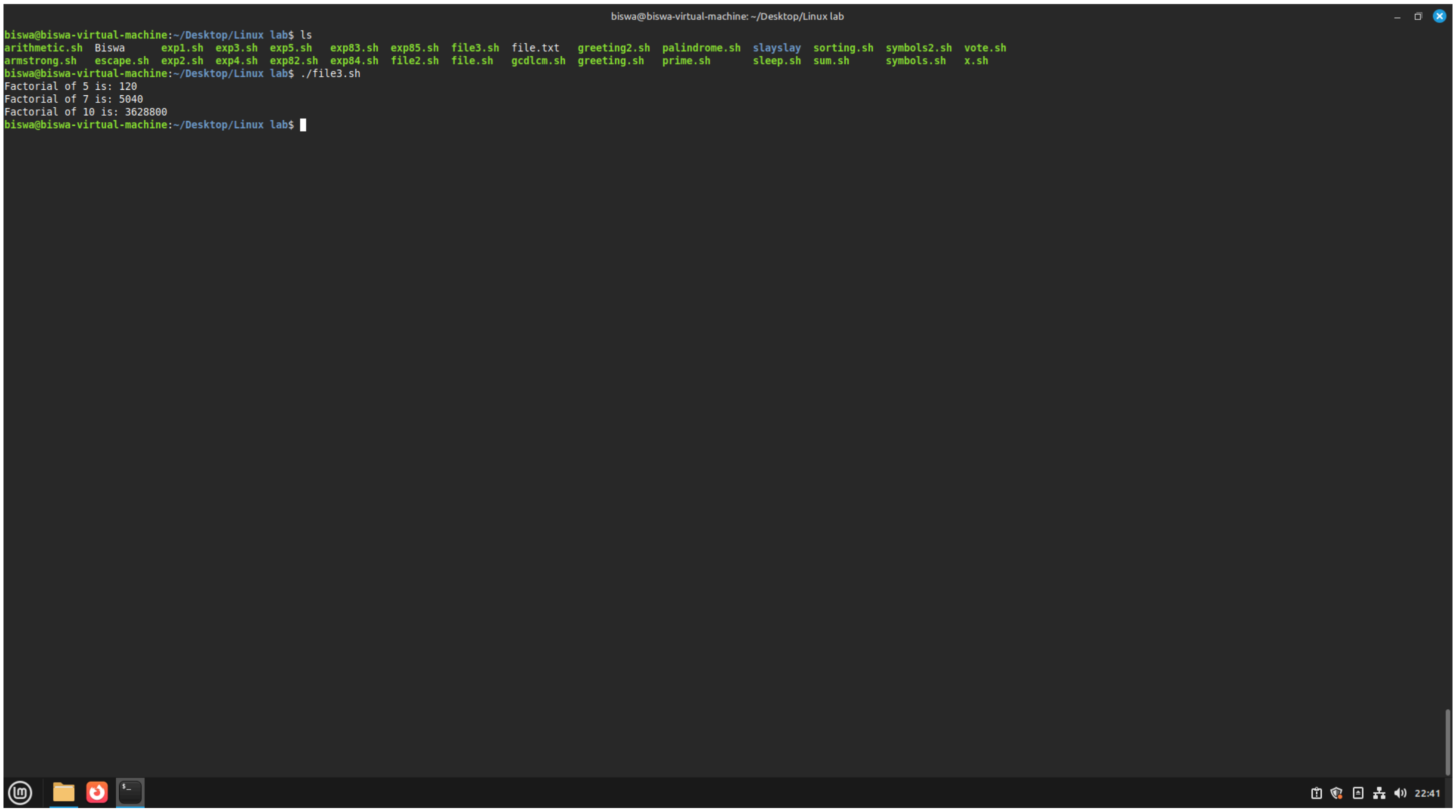
Assignments

Assignment 1: Factorial of a Number

Command(s):

```
#!/bin/bash
echo -n "Enter a number: "
read num
fact=1
for ((i=1;i<=num;i++)); do
  fact=$((fact*i))
done
echo "Factorial of $num is $fact"
```

Output:



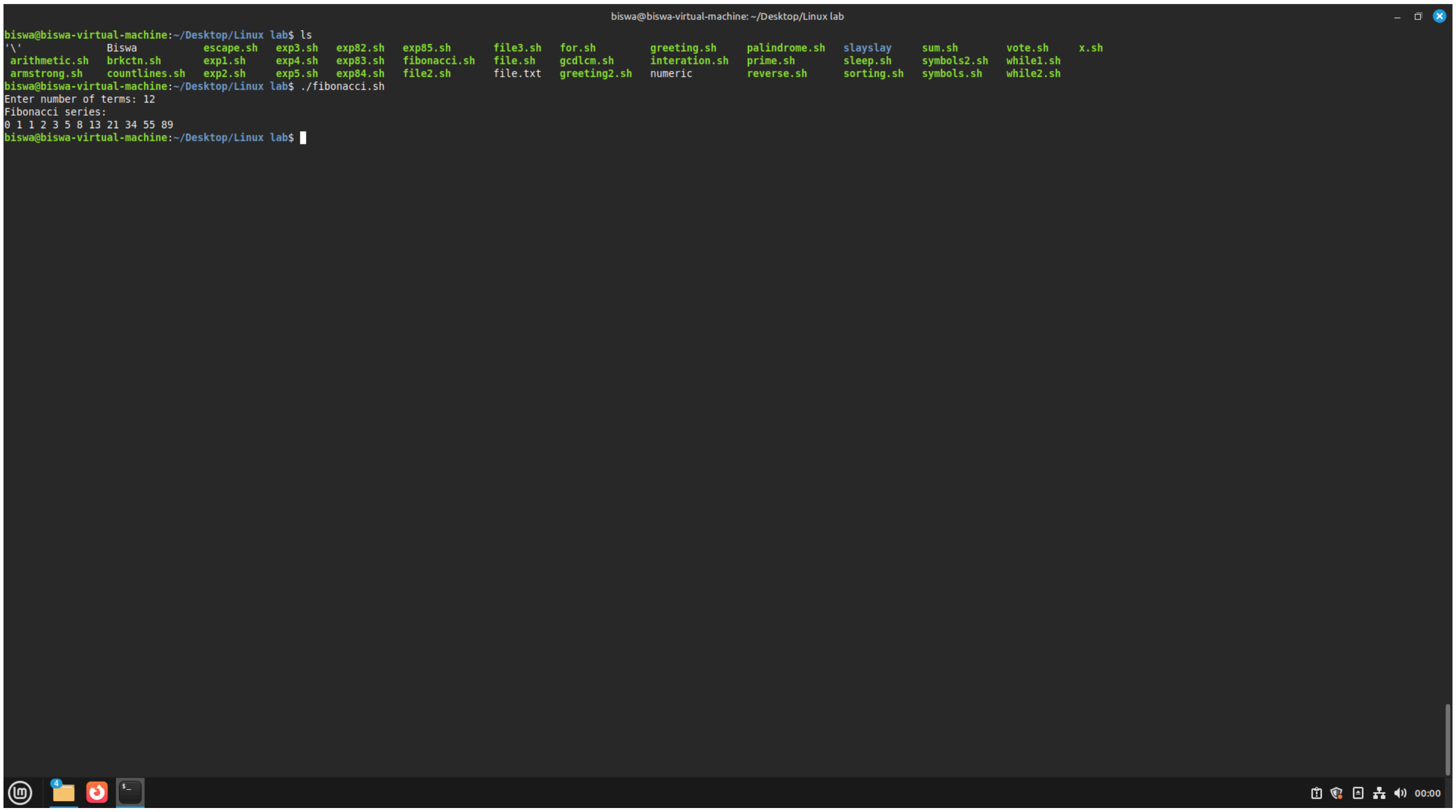
```
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ls
arithmetic.sh  Biswa      exp1.sh  exp3.sh  exp5.sh   exp83.sh  exp85.sh  file3.sh  file.txt   greeting2.sh  palindrome.sh  slayslay  sorting.sh  symbols2.sh  vote.sh
armstrong.sh   escape.sh  exp2.sh  exp4.sh  exp82.sh  exp84.sh  file2.sh  file.sh   gcdlcm.sh  greeting.sh   prime.sh       sleep.sh  sum.sh      symbols.sh   x.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ./file3.sh
Factorial of 5 is: 120
Factorial of 7 is: 5040
Factorial of 10 is: 3628800
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ 
```

Assignment 2: Fibonacci Series

Command(s):

```
#!/bin/bash
echo -n "Enter number of terms: "
read n
a=0
b=1
echo "Fibonacci series:"
for ((i=0;i<n;i++)); do
  echo -n "$a "
  fn=$((a+b))
  a=$b
  b=$fn
done
echo
```

Output:



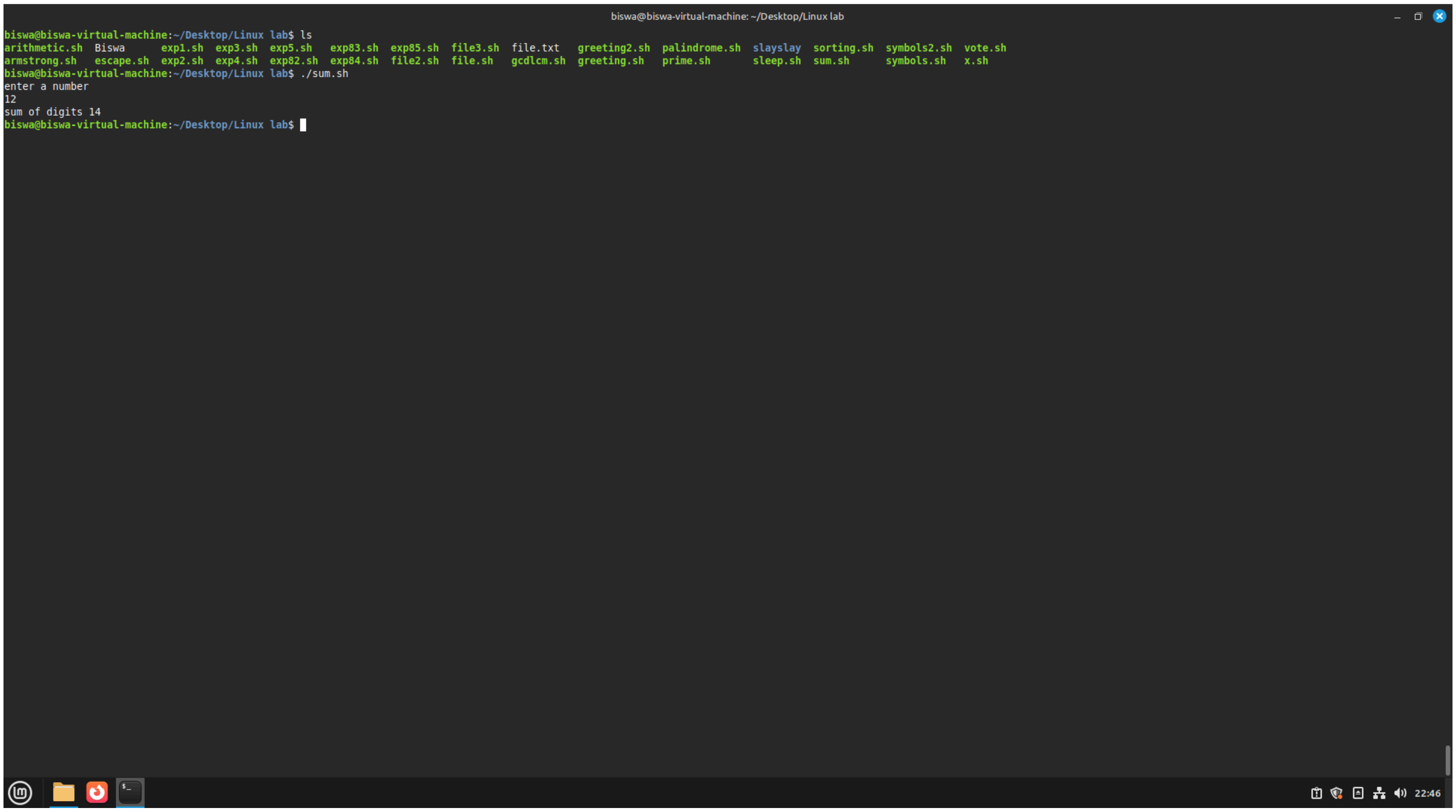
```
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ls
'\'              Biswa           escape.sh   exp3.sh   exp82.sh   exp85.sh       file3.sh   for.sh         greeting.sh     palindrome.sh   slayslay     sum.sh        vote.sh     x.sh
 arithmetic.sh   brkctn.sh       exp1.sh     exp4.sh   exp83.sh   fibonacci.sh   file.sh    gcdlcm.sh      interation.sh   prime.sh        sleep.sh     symbols2.sh   while1.sh
 armstrong.sh    countlines.sh   exp2.sh     exp5.sh   exp84.sh   file2.sh       file.txt   greeting2.sh   numeric         reverse.sh      sorting.sh   symbols.sh    while2.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ./fibonacci.sh
Enter number of terms: 12
Fibonacci series:
0 1 1 2 3 5 8 13 21 34 55 89
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ 
```

Assignment 3: Sum of Digits

Command(s):

```
#!/bin/bash
echo -n "Enter a number: "
read num
sum=0
temp=$num
while [ $temp -gt 0 ]; do
  digit=$((temp % 10))
  sum=$((sum + digit))
  temp=$((temp / 10))
done
echo "Sum of digits of $num is $sum"
```

Output:



biswa@biswa-virtual-machine: ~/Desktop/Linux lab

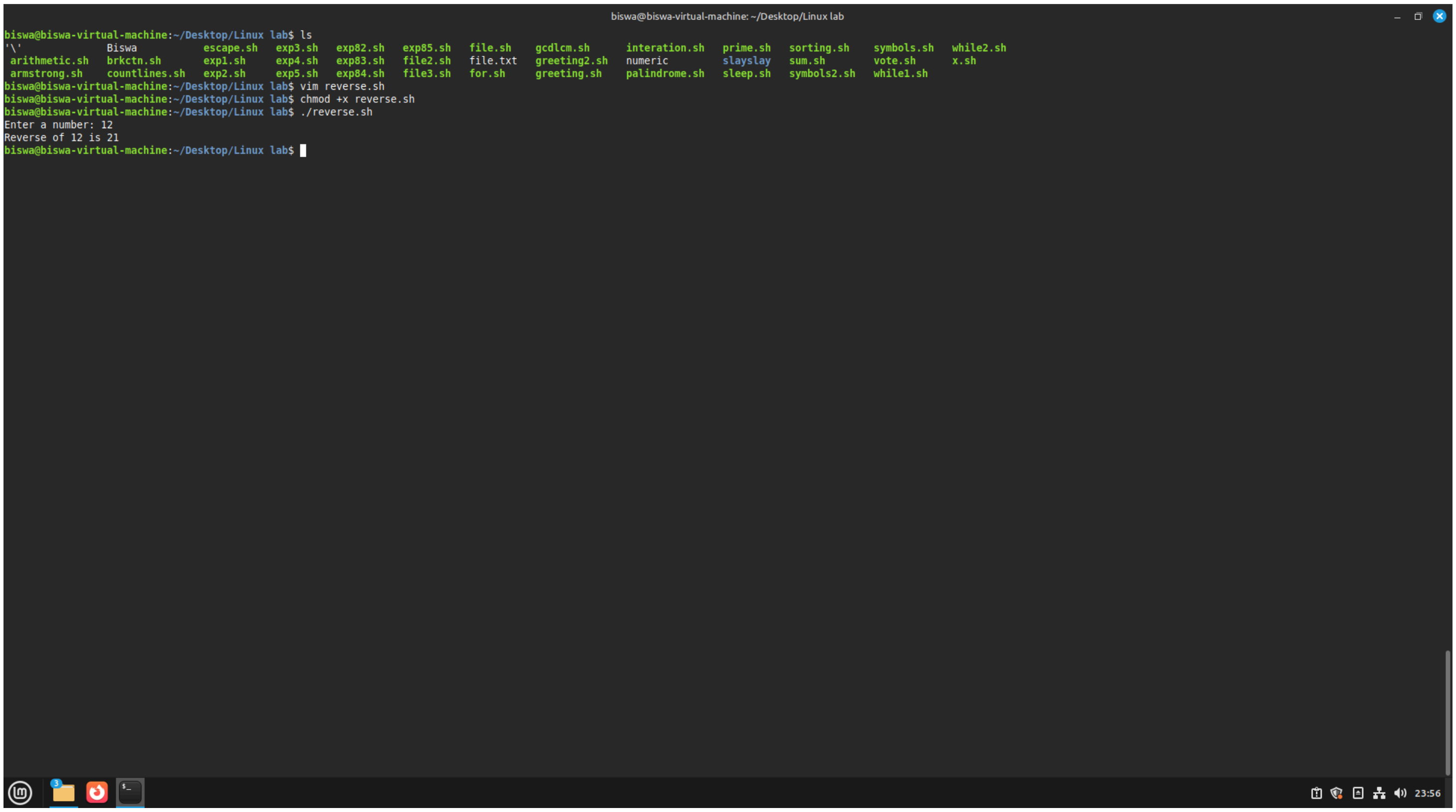
```
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ls
arithmetic.sh  Biswa      exp1.sh  exp3.sh  exp5.sh   exp83.sh  exp85.sh  file3.sh  file.txt   greeting2.sh  palindrome.sh  slayslay  sorting.sh  symbols2.sh  vote.sh
armstrong.sh   escape.sh  exp2.sh  exp4.sh  exp82.sh  exp84.sh  file2.sh  file.sh   gcdlcm.sh  greeting.sh   prime.sh       sleep.sh  sum.sh      symbols.sh   x.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ./sum.sh
enter a number
12
sum of digits 14
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ 
```

Assignment 4: Reverse a Number

Command(s):

```
#!/bin/bash
echo -n "Enter a number: "
read num
rev=0
temp=$num
while [ $temp -gt 0 ]; do
  digit=$((temp % 10))
  rev=$((rev * 10 + digit))
  temp=$((temp / 10))
done
echo "Reverse of $num is $rev"
```

Output:



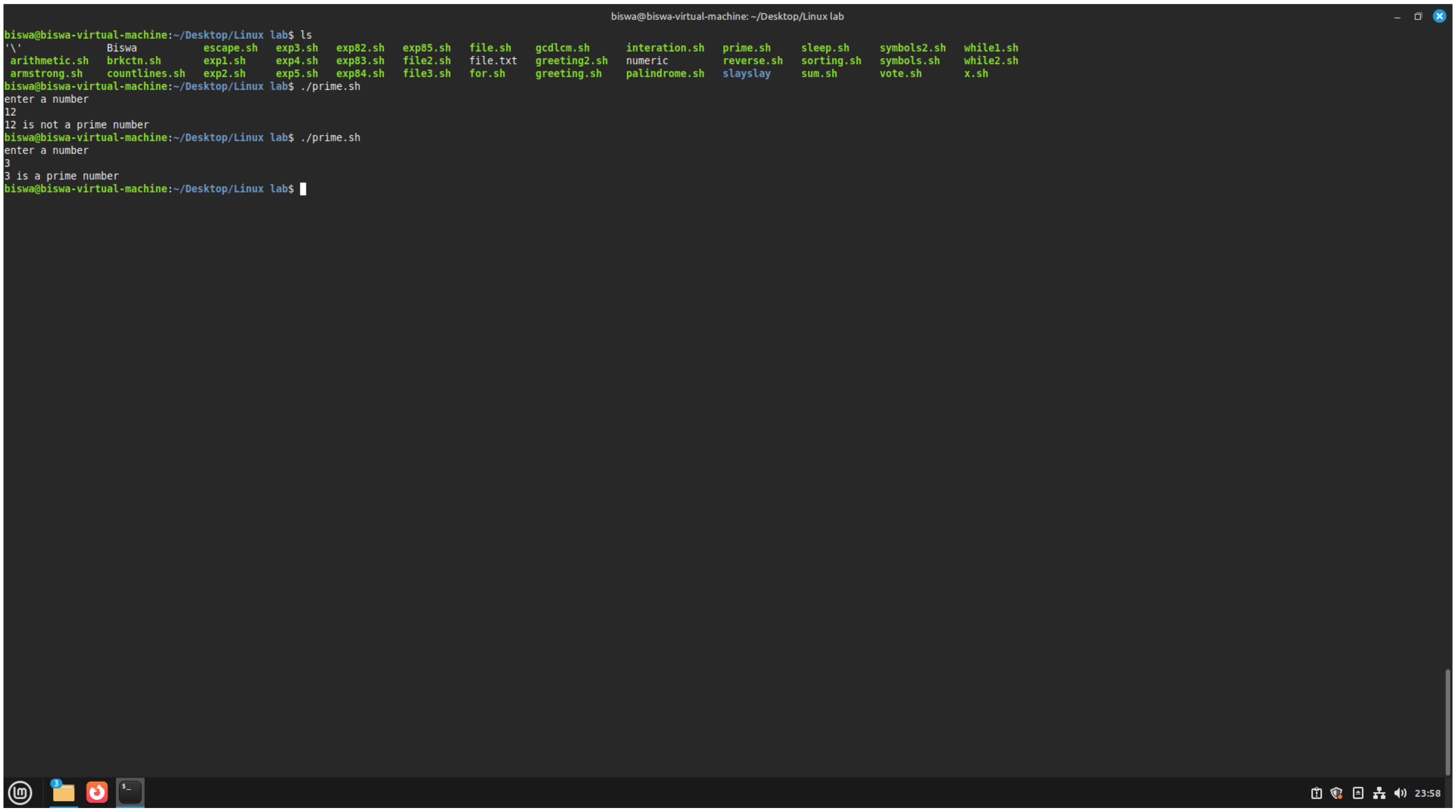
```
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ls
'\'             Biswa           escape.sh   exp3.sh   exp82.sh   exp85.sh   file.sh    gcdlcm.sh      interation.sh   prime.sh   sorting.sh    symbols.sh   while2.sh
 arithmetic.sh   brkctn.sh       exp1.sh     exp4.sh   exp83.sh   file2.sh   file.txt   greeting2.sh   numeric         slayslay   sum.sh        vote.sh      x.sh
 armstrong.sh    countlines.sh   exp2.sh     exp5.sh   exp84.sh   file3.sh   for.sh     greeting.sh    palindrome.sh   sleep.sh   symbols2.sh   while1.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ vim reverse.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ chmod +x reverse.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ./reverse.sh
Enter a number: 12
Reverse of 12 is 21
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ 
```

Assignment 5: Prime Number Check

Command(s):

```
#!/bin/bash
echo -n "Enter a number: "
read num
is_prime=1
for ((i=2;i<=num/2;i++)); do
  if (( num % i == 0 )); then
    is_prime=0
    break
  fi
done
if (( num <= 1 )); then
  echo "$num is not a prime number"
elif (( is_prime == 1 )); then
  echo "$num is a prime number"
else
  echo "$num is not a prime number"
fi
```

Output:



```
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ls
'\'             Biswa          escape.sh   exp3.sh   exp82.sh   exp85.sh   file.sh    gcdlcm.sh      interation.sh   prime.sh     sleep.sh     symbols2.sh   while1.sh
 arithmetic.sh  brkctn.sh      exp1.sh     exp4.sh   exp83.sh   file2.sh   file.txt   greeting2.sh   numeric         reverse.sh   sorting.sh   symbols.sh    while2.sh
 armstrong.sh   countlines.sh  exp2.sh     exp5.sh   exp84.sh   file3.sh   for.sh     greeting.sh    palindrome.sh   slayslay     sum.sh       vote.sh       x.sh
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ./prime.sh
enter a number
12
12 is not a prime number
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ ./prime.sh
enter a number
3
3 is a prime number
biswa@biswa-virtual-machine:~/Desktop/Linux lab$ 
```

Result

- Implemented `for` , `while` , and `until` loops and used loop control statements.
- Practiced reading input, processing files, nested iteration, and completed assignments like factorial, Fibonacci, sum of digits, reverse number, and prime check.

Challenges Faced & Learning Outcomes

- Challenge 1: Handling user input validation.
- Challenge 2: Managing arithmetic operations in loops.

Learning:

- Loops are powerful for automation in shell scripting.
- Implementing small programs like factorial and Fibonacci builds confidence in shell scripting.

Conclusion

The lab demonstrated practical loop constructs in Bash for automating repetitive tasks, and the assignments extended learning by applying loops to solve mathematical problems.