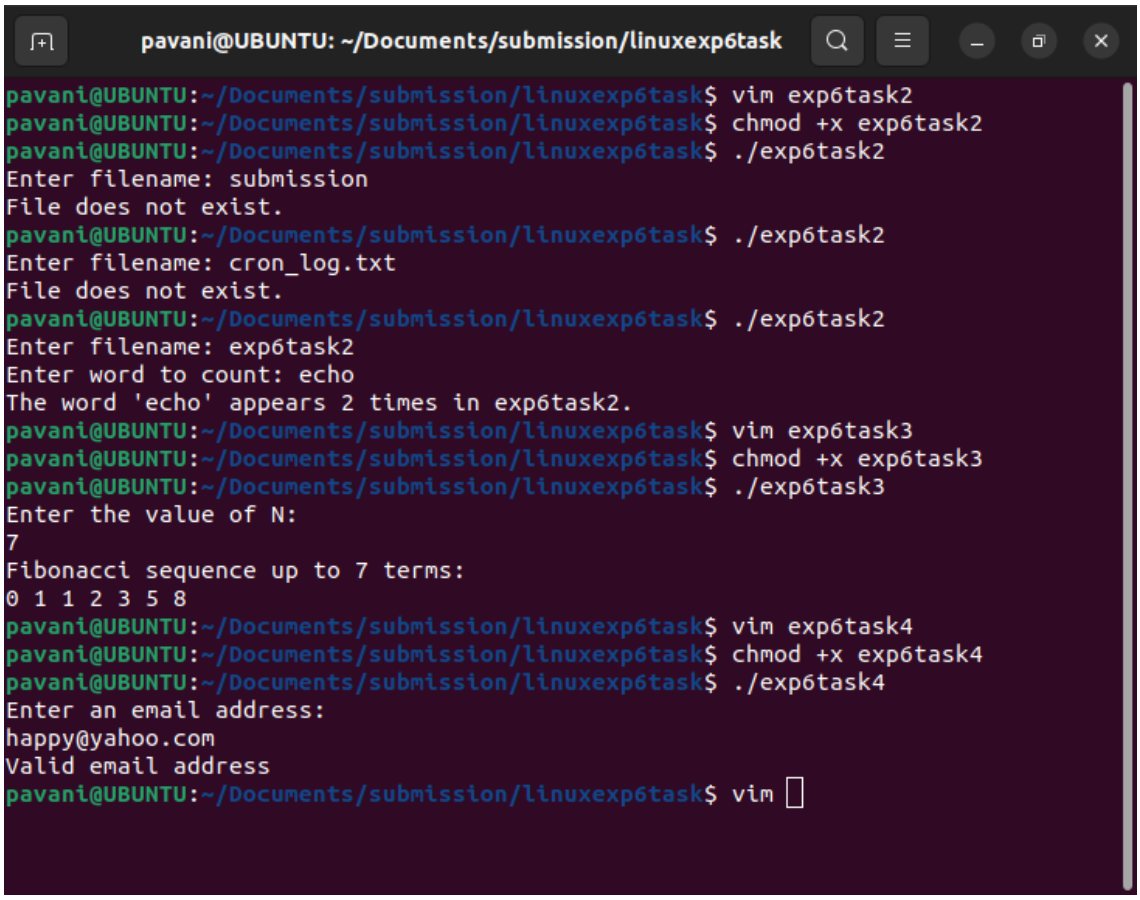
Experiment 6: Shell Programming (Advanced Concepts)

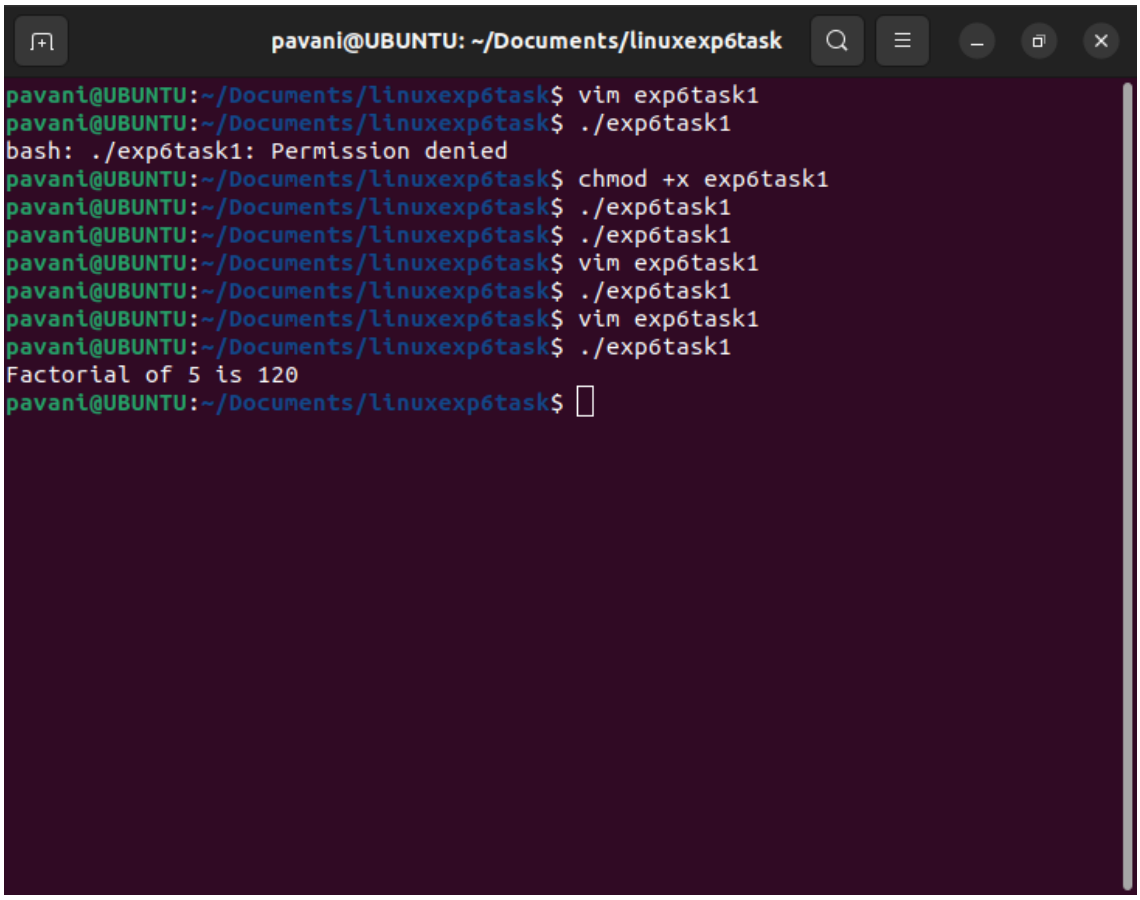
Aim

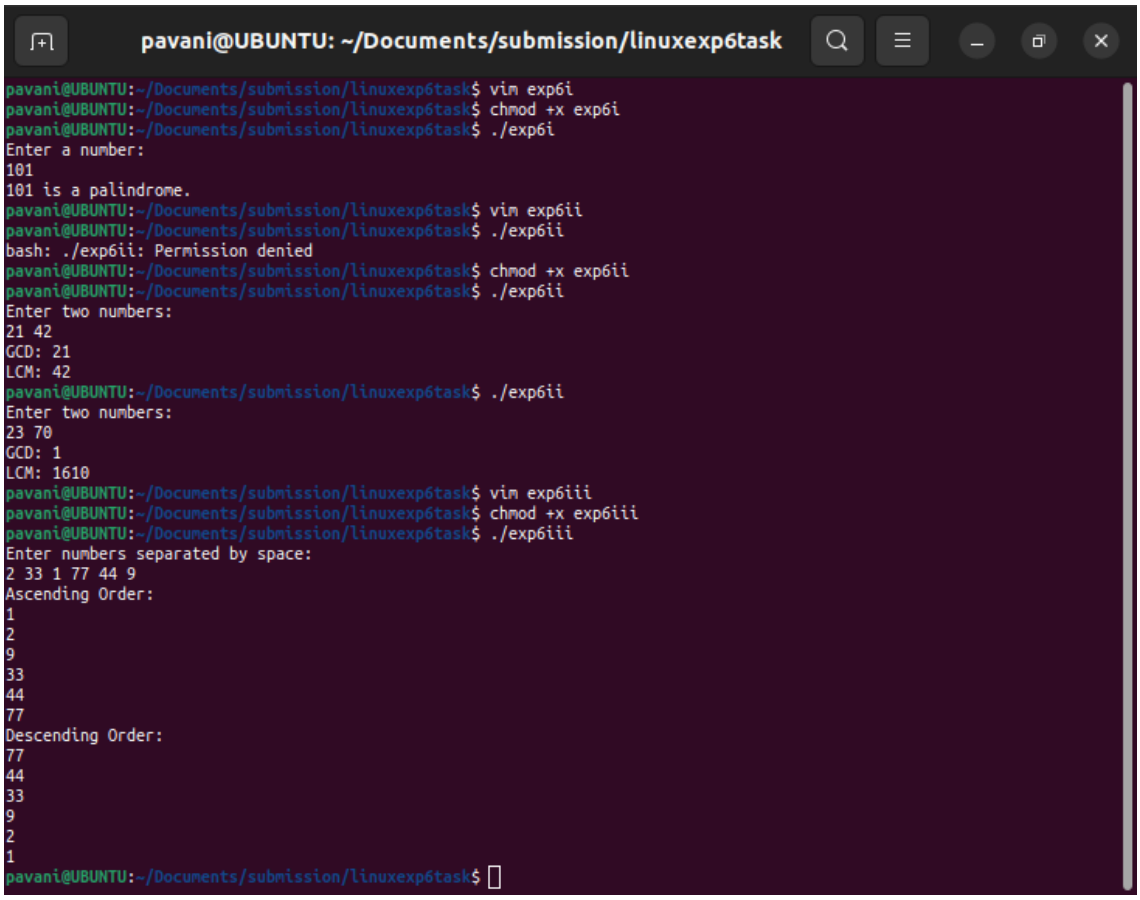
The aim of this experiment is to understand and implement advanced concepts in shell programming such as loops, loop control, input/output redirection, shell functions, use of regular expressions, and debugging scripts.

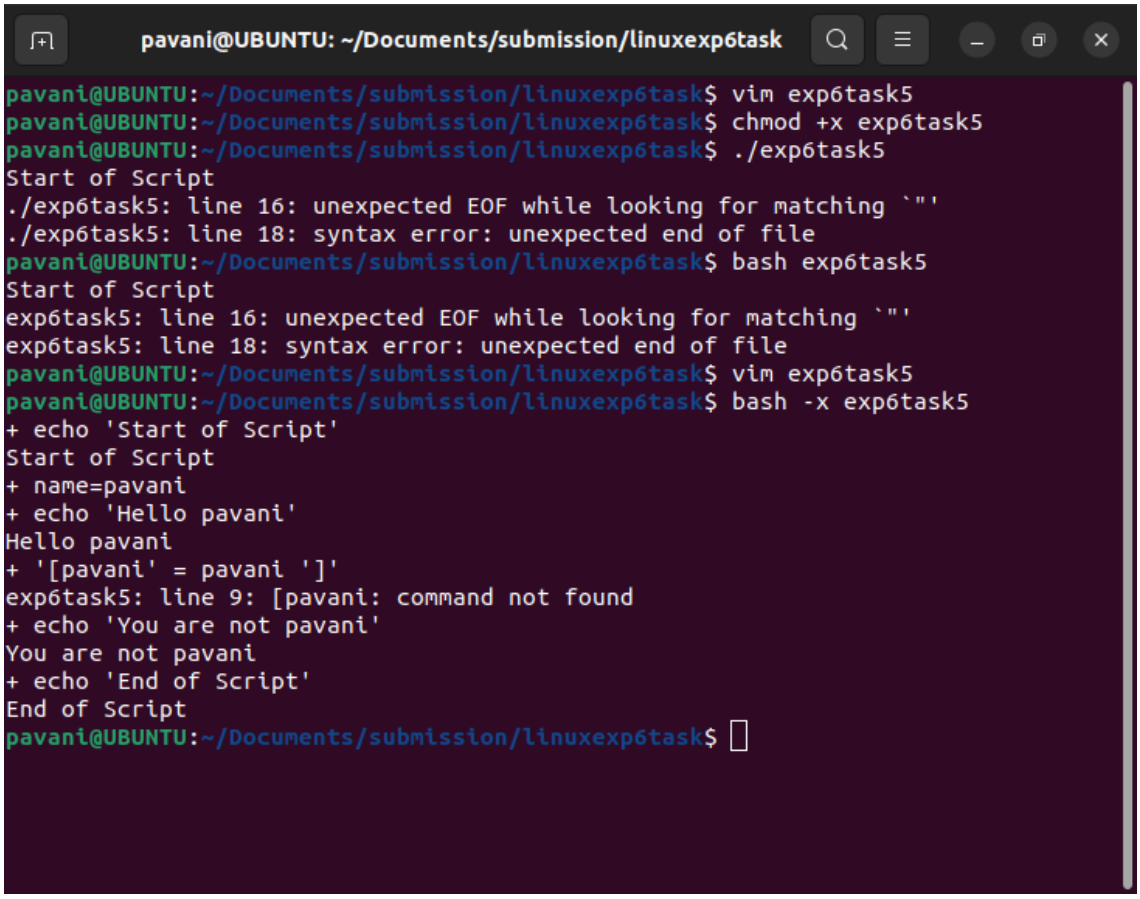
Requirements

1. Linux operating system (Ubuntu, Fedora, etc.)
2. Access to a terminal
3. Text editor (nano, gedit, or vim) to write scripts
4. Basic knowledge of shell programming5. Bash shell installed


pavani@UBUNTU: ~/Documents/submission/linuxexp6task
pavani@UBUNTU:~/Documents/submission/linuxexp6task$ vim exp6task2
pavani@UBUNTU:~/Documents/submission/linuxexp6task$ chmod +x exp6task2
pavani@UBUNTU:~/Documents/submission/linuxexp6task$ ./exp6task2
Enter filename: submission
File does not exist.
pavani@UBUNTU:~/Documents/submission/linuxexp6task$ ./exp6task2
Enter filename: cron_log.txt
File does not exist.
pavani@UBUNTU:~/Documents/submission/linuxexp6task$ ./exp6task2
Enter filename: exp6task2
Enter word to count: echo
The word 'echo' appears 2 times in exp6task2.
pavani@UBUNTU:~/Documents/submission/linuxexp6task$ vim exp6task3
pavani@UBUNTU:~/Documents/submission/linuxexp6task$ chmod +x exp6task3
pavani@UBUNTU:~/Documents/submission/linuxexp6task$ ./exp6task3
Enter the value of N:
7
Fibonacci sequence up to 7 terms:
0 1 1 2 3 5 8
pavani@UBUNTU:~/Documents/submission/linuxexp6task$ vim exp6task4
pavani@UBUNTU:~/Documents/submission/linuxexp6task$ chmod +x exp6task4
pavani@UBUNTU:~/Documents/submission/linuxexp6task$ ./exp6task4
Enter an email address:
happy@yahoo.com
Valid email address
pavani@UBUNTU:~/Documents/submission/linuxexp6task$ vim


pavani@UBUNTU: ~/Documents/linuxexp6task
pavani@UBUNTU:~/Documents/linuxexp6task$ vim exp6task1
pavani@UBUNTU:~/Documents/linuxexp6task$ ./exp6task1
bash: ./exp6task1: Permission denied
pavani@UBUNTU:~/Documents/linuxexp6task$ chmod +x exp6task1
pavani@UBUNTU:~/Documents/linuxexp6task$ ./exp6task1
pavani@UBUNTU:~/Documents/linuxexp6task$ ./exp6task1
pavani@UBUNTU:~/Documents/linuxexp6task$ vim exp6task1
pavani@UBUNTU:~/Documents/linuxexp6task$ ./exp6task1
pavani@UBUNTU:~/Documents/linuxexp6task$ vim exp6task1
pavani@UBUNTU:~/Documents/linuxexp6task$ ./exp6task1
Factorial of 5 is 120
pavani@UBUNTU:~/Documents/linuxexp6task$


pavani@UBUNTU: ~/Documents/submission/linuxexp6task
pavani@UBUNTU:~/Documents/submission/linuxexp6task$ vim exp6i
pavani@UBUNTU:~/Documents/submission/linuxexp6task$ chmod +x exp6i
pavani@UBUNTU:~/Documents/submission/linuxexp6task$ ./exp6i
Enter a number:
101
101 is a palindrome.
pavani@UBUNTU:~/Documents/submission/linuxexp6task$ vim exp6ii
pavani@UBUNTU:~/Documents/submission/linuxexp6task$ ./exp6ii
bash: ./exp6ii: Permission denied
pavani@UBUNTU:~/Documents/submission/linuxexp6task$ chmod +x exp6ii
pavani@UBUNTU:~/Documents/submission/linuxexp6task$ ./exp6ii
Enter two numbers:
21 42
GCD: 21
LCM: 42
pavani@UBUNTU:~/Documents/submission/linuxexp6task$ ./exp6ii
Enter two numbers:
23 70
GCD: 1
LCM: 1610
pavani@UBUNTU:~/Documents/submission/linuxexp6task$ vim exp6iii
pavani@UBUNTU:~/Documents/submission/linuxexp6task$ chmod +x exp6iii
pavani@UBUNTU:~/Documents/submission/linuxexp6task$ ./exp6iii
Enter numbers separated by space:
2 33 1 77 44 9
Ascending Order:
1
2
9
33
44
77
Descending Order:
77
44
33
9
2
1
pavani@UBUNTU:~/Documents/submission/linuxexp6task$



Observation

Loops in shell help to repeat commands.

Loop control statements like break and continue control execution flow.

I/O redirection is powerful for handling files and error messages.

Functions allow code reusability.

Regular expressions help in text searching and pattern matching.

Debugging tools like bash -x help in finding errors in scripts.

Conclusion

From this experiment, we conclude that advanced shell programming concepts make scripts more powerful and flexible. Loops and loop control help manage repeated tasks,

I/O redirection manages inputs and outputs, functions improve modularity, and regular expressions enhance text handling. Debugging techniques are essential for error-free execution of scripts.