D Y PATIL INTERNATIONAL UNIVERSITY
AKURDI PUNE

School of Computer Science, Engineering and Applications(SCSEA)

Academic Year: 2024-25 ***B.Tech S. Y. (Sem IV)***

Subject : System Software (CSE2103)

Name of the Student: Sakshi Biradar **PRN: 20230802152**

Title of Practical : **Linux Shell Scripting Automation Basics & Loops**

AIM – Lay the groundwork for shell programming**.**

- Shell scripting is a powerful tool for automating repetitive tasks, managing system operations, and performing complex computations in a Unix/Linux environment. Below is a detailed explanation of the topics mentioned:

---

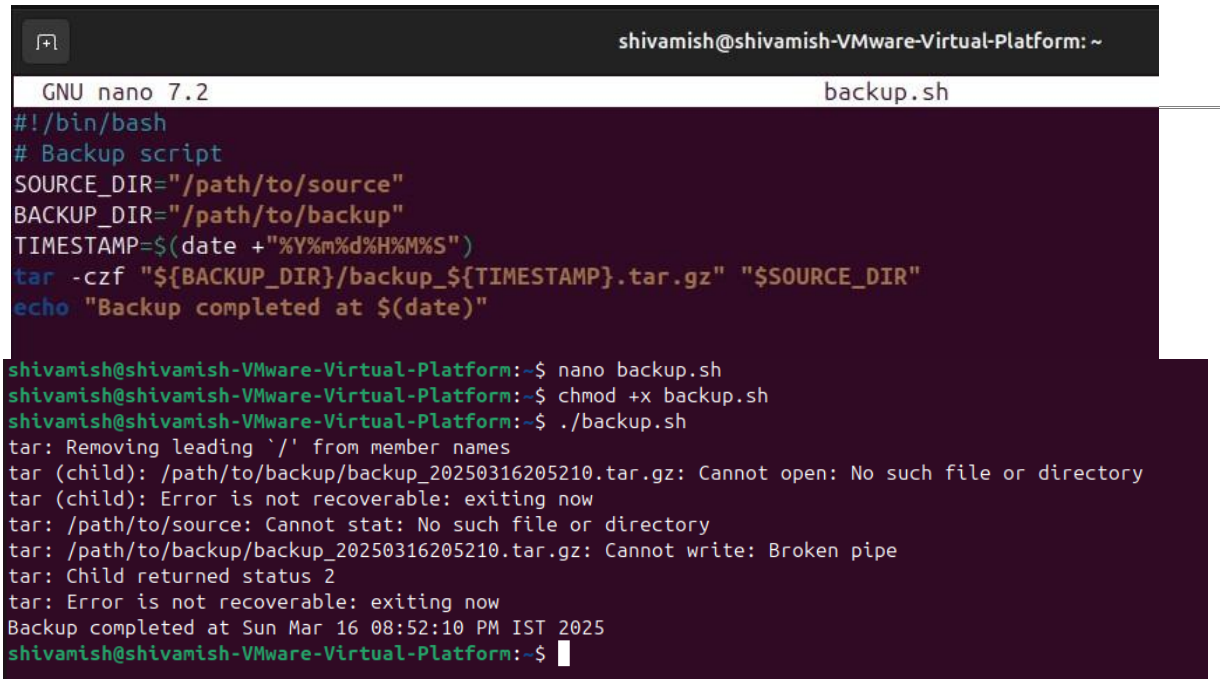
1. Writing Scripts for Simple Tasks (e.g., Automated File Backups)

- Shell scripts are text files containing a sequence of commands that are executed by the shell (e.g., Bash).
- **Automated File Backups**: A script can be written to copy files or directories to a backup location at regular intervals using commands like `cp`, `rsync`, or `tar`.
- Example:

bash
Copy

```
#!/bin/bash
# Backup script
SOURCE_DIR="/path/to/source"
BACKUP_DIR="/path/to/backup"
TIMESTAMP=$(date +"%Y%m%d%H%M%S")
tar -czf "${BACKUP_DIR}/backup_${TIMESTAMP}.tar.gz" "$SOURCE_DIR"
echo "Backup completed at $(date)"
```

- This script creates a compressed archive of the source directory and saves it with a timestamp.



2. Calculating the Sum of Even Numbers Within a Given Range

- Loops like `for` or `while` can be used to iterate through a range of numbers and perform calculations.
- Example:

bash
Copy

```
#!/bin/bash
# Sum of even numbers between 1 and 100
SUM=0
for ((i=2; i<=100; i+=2)); do
    SUM=$((SUM + i))
done
echo "Sum of even numbers from 1 to 100: $SUM"
```

- This script initializes a sum variable and adds even numbers to it using a `for` loop.

---

3. Calculating the Factorial of a Number Using a While Loop

- Factorial is the product of all positive integers up to a given number.
- Example:

bash
Copy

```
#!/bin/bash
# Factorial of a number
read -p "Enter a number: " num
FACTORIAL=1
COUNTER=1
while [ $COUNTER -le $num ]; do
    FACTORIAL=$((FACTORIAL * COUNTER))
    COUNTER=$((COUNTER + 1))
done
echo "Factorial of $num is $FACTORIAL"
```

- This script uses a `while` loop to calculate the factorial of a user-input number.

shivamish@shivamish-VMware-Virtual-Platform: ~

GNU nano 7.2 factoial.sh

```
#!/bin/bash
# Factorial of a number
read -p "Enter a number: " num
FACTORIAL=1
COUNTER=1
while [ $COUNTER -le $num ]; do
    FACTORIAL=$((FACTORIAL * COUNTER))
    COUNTER=$((COUNTER + 1))
done
echo "Factorial of $num is $FACTORIAL"
```

```
shivamish@shivamish-VMware-Virtual-Platform:~$ nano factoial.sh
shivamish@shivamish-VMware-Virtual-Platform:~$ chmod +x factorial.sh
chmod: cannot access 'factorial.sh': No such file or directory
shivamish@shivamish-VMware-Virtual-Platform:~$ chmod +x factoial.sh
shivamish@shivamish-VMware-Virtual-Platform:~$ ./factoial.sh
Enter a number: 5
Factorial of 5 is 120
shivamish@shivamish-VMware-Virtual-Platform:~$
```

---

4. Counting the Frequency of Words in a Text File

- Shell scripting can process text files using tools like `grep`, `awk`, and `sort`.
- Example:

bash
Copy

```
#!/bin/bash
# Word frequency counter
read -p "Enter the file name: " FILE
if [ -f "$FILE" ]; then
    tr '[:upper:]' '[:lower:]' < "$FILE" | tr -cs '[:alnum:]' '\n' | sort |
uniq -c | sort -nr
else
    echo "File not found!"
fi
```

- This script converts text to lowercase, splits it into words, and counts the frequency of each word.

shivamish@shivamish-VMware-Virtual-Platform: ~

GNU nano 7.2 words.sh

```
#!/bin/bash
# Word frequency counter
read -p "Enter the file name: " FILE
if [ -f "$FILE" ]; then
    tr '[:upper:]' '[:lower:]' < "$FILE" | tr -cs '[:alnum:]' '\n' | sort | uniq -c | sort -nr
else
    echo "File not found!"
fi
```

```
shivamish@shivamish-VMware-Virtual-Platform:~$ nano words.sh
shivamish@shivamish-VMware-Virtual-Platform:~$ chmod +x words.sh
shivamish@shivamish-VMware-Virtual-Platform:~$ ./words.sh
Enter the file name: backup.sh
      6 backup
      4 dir
      3 source
      2 to
      2 timestamp
      2 tar
      2 path
      2 m
      2 date
      1 y
      1 script
      1 s
      1 h
      1 gz
      1 echo
      1 d
      1 czf
      1 completed
      1 bin
      1 bash
      1 at
      1
shivamish@shivamish-VMware-Virtual-Platform:~$
```

5. Checking if a Number is Prime Using a Loop

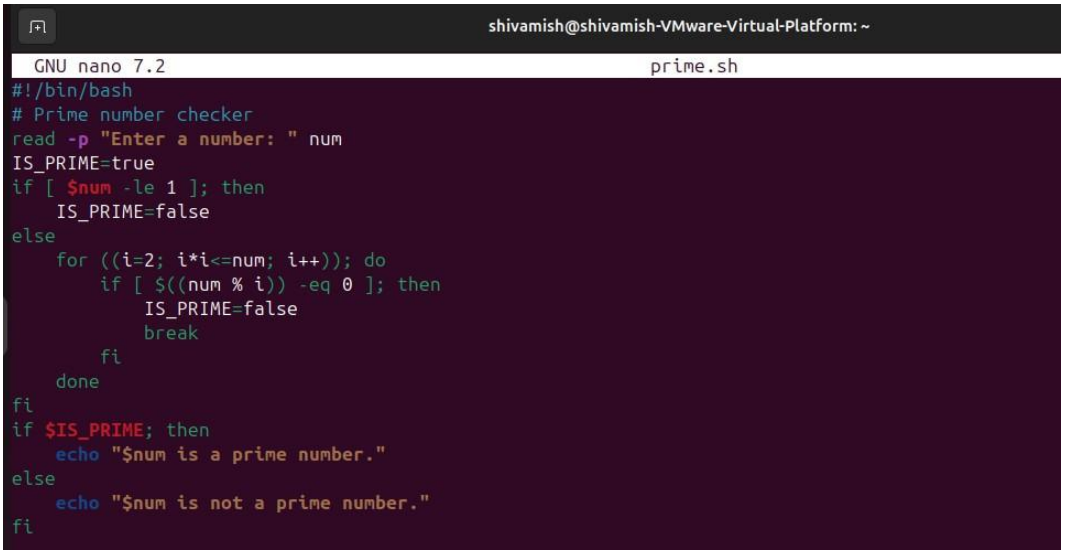
- A prime number is divisible only by 1 and itself. A loop can be used to check divisibility.
- Example:

bash
Copy

```
#!/bin/bash
# Prime number checker
read -p "Enter a number: " num
IS_PRIME=true
if [ $num -le 1 ]; then
    IS_PRIME=false
else
    for ((i=2; i*i<=num; i++)); do
        if [ $((num % i)) -eq 0 ]; then
            IS_PRIME=false
            break
        fi
    done
fi
if $IS_PRIME; then
    echo "$num is a prime number."
else
    echo "$num is not a prime number."
fi
```

- This script checks if a number is prime by testing divisibility up to the square root of the number.

```
shivamish@shivamish-VMware-Virtual-Platform:~$ nano prime.sh
shivamish@shivamish-VMware-Virtual-Platform:~$ chmod +x prime.sh
shivamish@shivamish-VMware-Virtual-Platform:~$ ./prime.sh
Enter a number: 8
8 is not a prime number.
shivamish@shivamish-VMware-Virtual-Platform:~$ chmod +x prime.sh
shivamish@shivamish-VMware-Virtual-Platform:~$ ./prime.sh
Enter a number: 2
2 is a prime number.
shivamish@shivamish-VMware-Virtual-Platform:~$
```



Conclusion:

Shell scripting is an essential skill for automating tasks, performing calculations, and managing system operations in a Unix/Linux environment. By mastering basic concepts like loops, conditionals, and file operations, you can create efficient scripts to solve real-world problems. The examples provided demonstrate how to automate backups, perform mathematical computations, process text files, and implement logic using loops. With practice, you can extend these concepts to build more complex and powerful scripts.

Course TA

(Mr. Rajkumar Sir)