

Shell Scripting

PART 1: Shell Scripting Basics

1. What is a Shell Script?

A **shell script** is a file containing a sequence of Linux commands executed by the shell.

Why Shell Scripting?

- Automate repetitive tasks
- System administration
- File & process management
- Scheduling jobs

2. Writing Your First Shell Script

Step 1: Create a Script File

```
touch hello.sh
```

Step 2: Add Shebang Line

Open file:

```
nano hello.sh
```

Add:

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

```
echo "Hello, Welcome to Shell Scripting"
```

👉 `#!/bin/bash` tells the system which shell to use.

Step 3: Run the Script

```
bash hello.sh
```

Output:

```
Hello, Welcome to Shell Scripting
```

3. Variables in Shell Script

3.1 Creating Variables

```
name="Linux"
```

```
echo $name
```

⚠ No space before or after =

3.2 User Input

```
echo "Enter your name:"
```

```
read username
```

```
echo "Hello $username"
```

3.3 System Variables

```
echo $USER
```

```
echo $HOME
```

```
echo $PWD
```

4. Making Script Executable

Step 1: Give Execute Permission

```
chmod +x hello.sh
```

Step 2: Run Script

```
./hello.sh
```

5. Practice Exercises (Basic)

1. Create a script to print your name and date
2. Read two numbers and print sum
3. Print current directory and logged-in user

PART 2: Advanced Shell Scripting

6. Conditional Statements

6.1 if Statement

```
#!/bin/bash
echo "Enter a number:"
read num

if [ $num -gt 10 ]
then
    echo "Number is greater than 10"
fi
```

6.2 if-else Statement

```
if [ $num -ge 50 ]
then
    echo "Pass"
else
    echo "Fail"
fi
```

6.3 if-elif-else

```
if [ $num -ge 80 ]
then
    echo "Distinction"
elif [ $num -ge 60 ]
then
    echo "First Class"
else
    echo "Needs Improvement"
fi
```

6.4 case Statement

```
#!/bin/bash
echo "Enter a choice (start|stop|restart):"
read choice

case $choice in
start)
    echo "Service Started";;
stop)
    echo "Service Stopped";;
restart)
    echo "Service Restarted";;
*)
    echo "Invalid choice";;
esac
```

7. Loops

7.1 for Loop

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

7.2 for Loop with Range

```
for i in {1..5}
do
    echo $i
done
```

7.3 while Loop

```
count=1
while [ $count -le 5 ]
do
    echo $count
    count=$((count+1))
done
```

8. Command Line Arguments

```
#!/bin/bash
echo "Script name: $0"
echo "First argument: $1"
echo "Second argument: $2"
```

Run:

```
./args.sh Apple Banana
```

9. Functions in Shell Script

```
#!/bin/bash
add() {
    sum=$(( $1 + $2 ))
    echo "Sum = $sum"
}

add 10 20
```

PART 3: Practical System Tasks

10. File Operations Using Script

```
#!/bin/bash
mkdir demo
cd demo
touch a.txt b.txt
ls
```

11. Background Processes

Run command in background:

```
sleep 60 &
```

Check jobs:

```
jobs
```

Bring to foreground:

`fg`

12. Scheduling Jobs

12.1 Cron Job

Edit crontab:

`crontab -e`

Example (run every minute):

```
* * * * * echo "Hello" >> log.txt
```

Cron format:

```
* * * * *
```

```
| | | | |
```

```
| | | | +-- Day of week
```

```
| | | +---- Month
```

```
| | +----- Day of month
```

```
| +----- Hour
```

```
+----- Minute
```

12.2 at Command

```
echo "date" | at now + 1 minute
```

12.3 batch Command

`batch`

13. Networking Commands

```
ping google.com  
ifconfig  
netstat -tuln  
ss -tuln
```

PART 4: Automation Mini Project

14. Automation Project – Backup Script

Requirement:

- Take backup of a directory
- Compress it
- Store with date

Script:

```
#!/bin/bash  
  
src="/home/user/data"  
dest="/home/user/backup"  
date=$(date +%F)  
  
mkdir -p $dest  
tar -czvf $dest/backup_$(date +%F).tar.gz $src  
  
echo "Backup completed on $(date)"
```

15. Practice Assignments

1. Script to check file exists or not

2. Script to monitor disk usage
3. Script to kill a process by name
4. Script to automate cleanup of old files