```
# Day 4 Task: Basic Linux Shell Scripting for DevOps Engineers.

## What is Kernel

The kernel is a computer program that is the core of a computer's operating
system, with complete control over everything in the system.

## What is Shell

A shell is special user program which provide an interface to user to use
operating system services. Shell accept human readable commands from user and
convert them into something which kernel can understand. It is a command
language interpreter that execute commands read from input devices such as
keyboards or from files. The shell gets started when the user logs in or start
the terminal.

## What is Linux Shell Scripting?

A shell script is a computer program designed to be run by a linux shell, a
command-line interpreter. The various dialects of shell scripts are considered
to be scripting languages. Typical operations performed by shell scripts
include file manipulation, program execution, and printing text.

**Tasks**

- Explain in your own words and examples, what is Shell Scripting for DevOps.
→ Shell scripting in DevOps involves writing scripts to automate tasks in Unix/Linux environments. These scripts, written in shell languages like Bash, help streamline repetitive tasks, such as deploying code, managing servers, and configuring environments. For example, a shell script can automate the deployment process by pulling the latest code from a repository, building the application, and restarting the server. Another example is creating a script to monitor system health, checking CPU and memory usage, and sending alerts if thresholds are exceeded. Shell scripting enhances efficiency, consistency, and reliability in DevOps workflows.
- What is `#!/bin/bash?` can we write `#!/bin/sh` as well?
```

→ `#!/bin/bash` is a shebang line used at the beginning of a shell script to indicate that the script should be executed using the Bash shell. The `#!` is known as a shebang, and it specifies the path to the interpreter that should be used to run the script.

Yes, you can write `#!/bin/sh` as well. `#!/bin/sh` specifies that the script should be executed using the Bourne shell, which is typically linked to a shell that provides a POSIX-compliant environment. On many systems, `/bin/sh` is a symbolic link to `/bin/bash` or another shell.

The choice between `#!/bin/bash` and `#!/bin/sh` depends on the script's requirements:

- Use `#!/bin/bash` if the script uses Bash-specific features.
- Use `#!/bin/sh` for portability and POSIX compliance, ensuring the script runs on various Unix-like systems.

- Write a Shell Script which prints `I will complete #90DaysOofDevOps challenge`

→

```
#!/bin/bash
echo "I will complete #90DaysOfDevOps challenge"
```

- Write a Shell Script to take user input, input from arguments and print the variables.

```
#!/bin/bash
echo "Enter the name:"
read username
echo "you entered $username"
echo "the chareacters in $0 are $1 $2"
```

→

```
ubuntu@ip-172-31-33-37:~/shell-scripting-for-devops$ ./script.sh iyer bhide
Enter the name:
Tarak mehat ka ulta chshma:
you entered Tarak mehat ka ulta chshma:
the chareacters in ./script.sh are iyer bhide
```

- Write an Example of If else in Shell Scripting by comparing 2 numbers

→

```
#!/bin/bash

# Prompt the user to enter two numbers
read -p "Enter the first number: " num1
read -p "Enter the second number: " num2

# Compare the two numbers
if [ "$num1" -gt "$num2" ]; then
  echo "$num1 is greater than $num2"
elif [ "$num1" -lt "$num2" ]; then
  echo "$num1 is less than $num2"
else
  echo "$num1 is equal to $num2"
fi
```

```
ubuntu@ip-172-31-33-37:~/shell-scripting-for-devops$ vim script.sh
ubuntu@ip-172-31-33-37:~/shell-scripting-for-devops$ ./script.sh
Enter the first number: 10
Enter the second number: 25
10 is less than 25
```

Was it difficult?

→No

- Post about it on LinkedIn and Let me know :)

→Done