📌 Day 3 Summary - Linux & Shell Scripting Commands

1️⃣ Directory & File Management

- `rmdir / rmdir -p a/b/c/d/e/f`
  - **Removes empty directories**. The `-p` flag removes parent directories if they become empty.
- `cat content.txt`
  - **Displays the content of a file** on the terminal.
- `wc -l name.txt`
  - Counts the **number of lines** in `name.txt`.
- `wc -w name.txt`
  - Counts the **number of words** in `name.txt`.
- `wc -c name.txt`
  - Counts the **number of characters** in `name.txt`.

2️⃣ System Information & Monitoring

- `uname`
  - Displays system information (kernel name, version, etc.).
- `uname -a`
  - Shows detailed system info including OS, kernel version, machine type, etc.
- `whoami`
  - Displays the **current logged-in user**.
- `ps aux`
  - Shows **all running processes** with details like user, PID, CPU usage, etc.
- `top`
  - Shows **real-time processes & system resource usage** (CPU, memory).
- `du -sh *`
  - Displays the **disk usage** of all files and directories in the current folder.
- `ncdu .`
  - **Interactive disk usage analyzer** (requires `sudo apt install ncdu`).

3️⃣ Package Management & Software Installation

- `sudo apt install ncal`
  - Installs the **`ncal` calendar utility**.
- `cal 2025` / `ncal 12 2025`
  - Displays a calendar for a specific year/month.
- `sudo apt update`
  - Updates the **package lists** for available updates.

- `sudo apt install`
  - Installs new software packages.

---

4 Searching & Finding Files

- `find . -name "*.txt"`
  - Finds all `.txt` files in the current directory and subdirectories.
- `find . -type d`
  - Lists all **directories** inside the current folder.
- `find . -name "*.tmp" -exec rm {} \;`
  - Finds and **deletes all `.tmp` files**.
- `whereis python`
  - Locates the binary, source, and manual pages for Python.
- `where python`
  - Finds **all paths where Python is installed**.

---

5 File Archiving & Compression

- `tar -cvf etc.tar Invoice-9.pdf`
  - Creates a **tar archive** (`etc.tar`) containing `Invoice-9.pdf`.
- `tar -cvf etc1.tar .`
  - Creates a tar archive of the **current directory**.
- `tar -xzvf etc1.tar`
  - Extracts the **tar archive**.
- `tar -cvf etc12.tar.gz .`
  - Creates a **compressed tar archive** (`.tar.gz`).
- `tar -xzvf etc12.tar.gz`
  - Extracts a **compressed tar.gz archive**.
- `tar -tzvf etc12.tar.gz`
  - Lists the **contents** of the archive without extracting.

---

6 Shell Scripting Basics

Variables

```
var_name="Hitesh"
var_age=23
echo "Name is $var_name and age is $var_age"
```

- Defines **variables** and prints their values.

Readonly Variables

```
var_blood_group="O-"
readonly var_blood_group
```

- **Marks a variable as readonly**, preventing modification.

Unset Variables

```
unset var_age
echo "Age is after unsetting: $var_age"
```

- **Removes the variable** from memory.

Time-Based Greetings

```
time=$(date +%H)
if [ $time -lt 12 ]; then
    message="Good morning user"
elif [ $time -lt 18 ]; then
    message="Good afternoon user"
else
    message="Good evening user"
fi
echo "$message $time"
```

- Uses `date +%H` to fetch **current hour** and print a **greeting** accordingly.

---

7 $hell Scripting - Conditional Statements & User Input Handling

Banking Transaction Script

```
balance=500
withdrawl=1200
daily_limit=1000
account_type="savings"

if [ $withdrawl -le $balance -a $withdrawl -le $daily_limit ]; then
  echo "Transaction approved"
else
  echo "Transaction not approved"
fi
```

User Input & Case Statements

```
read -p "Enter account number and password: " acn password
echo "Account Number: $acn"
echo "Password: $password"

read -p "Enter selection [1-3]: " selection
case $selection in
  1) accounttype="checking"; echo "You selected checking";;
  2) accounttype="saving"; echo "You selected saving";;
```

```
  3) accounttype="current"; echo "You selected current";;
  *) echo "Invalid selection";;
esac
```

Basic Calculator

```
while true;
do
    echo "Enter two numbers:"
    read a
    read b
    echo "Enter choice:"
    echo "1.Addition"
    echo "2.Subtraction"
    echo "3.Multiplication"
    echo "4.Division"
    echo "5.Exit"
    read ch
    case $ch in
        1) res=$((a + b)) ;;
        2) res=$((a - b))  ;;
        3) res=$((a * b))  ;;
        4) if [ $b -eq 0 ]; then
                echo "Cannot divide by zero"
           else
                res=$((a / b))
           fi ;;
        5) exit ;;
        *) echo "Invalid choice" ;;
    esac
    echo "Result: $res"
done
```

✅ Key Takeaways

1. **File & Directory Management:** Use `rmdir`, `cat`, `wc`, and `find` for organizing files.
2. **Shell Scripting:** Implement **banking scripts, user inputs, and case statements**.
3. **Process Management:** Use `kill`, `alias`, and `history` to manage processes.
4. **Text Processing with AWK:** Use `awk` to extract & format data from files.