🧪 LAB SETUP STRUCTURE

```
mkdir vim_file_lab
cd vim_file_lab
```

You will create the following files:

```
vim_file_lab/
│
├── app.log
├── users.txt
├── config_v1.txt
├── config_v2.txt
├── yesterday.log
├── today.log
├── dirA/
│   ├── app.conf
│   └── database.conf
└── dirB/
    ├── app.conf
    └── cache.conf
```

📄1️⃣app.log (For head, tail, less, wc, live monitoring)

Create:

```
vim app.log
```

Paste this content:

```
INFO: Server started at 08:00
INFO: Database connected
```

```
INFO: User admin logged in
WARNING: CPU usage high
INFO: Backup started
ERROR: Failed to connect to payment gateway
INFO: Retrying connection
INFO: Connection successful
INFO: User john logged in
WARNING: Disk space 80% full
INFO: File upload completed
ERROR: Invalid user input detected
INFO: Cache cleared
INFO: Scheduled job executed
INFO: Email service started
WARNING: Memory usage high
INFO: User mary logged in
ERROR: Timeout while calling API
INFO: Restarting API service
INFO: API service started
INFO: Health check passed
WARNING: High network latency
INFO: Cleanup completed
ERROR: Database connection lost
INFO: Database reconnected
INFO: System stable
```

Save and exit (`:wq`).

📄2️⃣users.txt (For head, wc, less)

```
vim users.txt
```

Content:

```
ID Name Role
1 John Admin
```

```
2 Mary Developer
3 Alex Tester
4 Sara DevOps
5 David Manager
6 Priya Developer
7 Amit Support
8 Neha HR
9 Rahul Developer
10 Tina Tester
11 Arjun DevOps
12 Kavya Developer
13 Rohan Support
14 Meera Analyst
15 Vikram Developer
```

📄3️⃣config_v1.txt (Old Configuration)

```
vim config_v1.txt
```

Content:

```
server_name=production_server
port=8080
max_clients=100
timeout=30
log_level=INFO
cache_enabled=true
database_host=localhost
database_port=5432
```

📄4️⃣config_v2.txt (New Configuration)

```
vim config_v2.txt
```

Content:

```
server_name=production_server
port=9090
max_clients=200
timeout=60
log_level=DEBUG
cache_enabled=false
database_host=db.production.local
database_port=5432
enable_ssl=true
```

👉 This will help students practice:

- diff
- vimdiff
- merging
- added lines
- changed values

📄5️⃣yesterday.log

```
vim yesterday.log
```

Content:

```
INFO: Server started
INFO: User login success
INFO: Backup completed
ERROR: Payment API failed
INFO: Retry successful
INFO: Health check passed
```

📄6️⃣today.log

```
vim today.log
```

Content:

```
INFO: Server started
INFO: User login success
WARNING: Disk space low
ERROR: Payment API failed
INFO: Retry successful
INFO: Health check passed
INFO: New deployment completed
```

👉 Used for:

- diff comparison
- wc line comparison
- log analysis

📁7️⃣Directory Comparison Files

Create directories:

```
mkdir dirA dirB
```

---

dirA/app.conf

```
vim dirA/app.conf
```

Content:

```
app_name=InventorySystem
version=1.0
port=8080
environment=production
```

---

dirA/database.conf

```
vim dirA/database.conf
```

Content:

```
db_name=inventory
db_user=admin
db_password=admin123
db_host=localhost
```

---

dirB/app.conf

```
vim dirB/app.conf
```

Content:

```
app_name=InventorySystem
version=2.0
port=9090
environment=production
```

---

dirB/cache.conf

```
vim dirB/cache.conf
```

Content:

```
cache_enabled=true
```

```
cache_size=256
cache_type=redis
```

🎯 What Students Can Practice With These Files

These instructions assume they are inside:

```
cd vim_file_lab
```

🔹 1️⃣ HEAD – Step-by-Step Instructions

🎯 Objective: View beginning of file

✅ Task 1: View first 10 lines (default)

✅ Task 2: View first 5 lines

✅ Task 3: View first line of all text files

🔹 2️⃣ TAIL – Step-by-Step Instructions

🎯 Objective: View end of file & monitor logs

✅ Task 1: View last 10 lines

✅ Task 2: View last 5 lines

✅ Task 3: Live Log Monitoring

🔹 Terminal 1:

```
tail -f app.log
```

🔹 Terminal 2:

```
vim app.log
```

Add new line at bottom:

```
ERROR: New payment failure detected
```

Save file:

```
:w
```

👉 Observe real-time update in Terminal 1.

Stop monitoring:

```
Ctrl + C
```

🔹 3️⃣ LESS – Step-by-Step Instructions

🎯 Objective: Scroll & Search inside file

✅ Task 1: Open file

```
less app.log
```

Inside less:

Key	Action
Space	Next page
b	Previous page
g	Top of file
G	Bottom of file
/ERROR	Search word
n	Next match
N	Previous match
q	Quit

👉 Ask students to:

- Find all ERROR lines
- Jump to bottom
- Return to top

🔹 4️⃣ MORE – Step-by-Step Instructions

🎯 Objective: Basic page viewing

```
more users.txt
```

Try:

- Space → next page
- Try going backward (Not possible)

- Try searching

🔹 5️⃣ WC – Step-by-Step Instructions

🎯 Objective: Count lines, words, characters

✅ Task 1: Full count of characters, words, and lines of `users.txt`

✅ Task 2: Count only lines of `users.txt`

✅ Task 3: Count words in all text files

🔹 6️⃣ DIFF – Step-by-Step Instructions

🎯 Objective: Compare files

✅ Task 1: Compare configs `config_v1.txt & config_v2.txt`

👉 Identify:

- Changed lines (c)
- Added lines (a)
- Deleted lines (d)

✅ Task 2: Side-by-side comparison

```
diff -y config_v1.txt config_v2.txt
```

✅ Task 3: Compare logs

```
diff yesterday.log today.log
```

🔹 7️⃣ Directory Comparison – Step-by-Step

🎯 Compare dirA and dirB

```
diff -r dirA dirB
```

👉 Observe:

- Files only in dirA
- Files only in dirB
- Modified files

🔹 8️⃣ VIMDIFF – Step-by-Step Instructions

🎯 Objective: Interactive comparison & merging

✅ Step 1: Open comparison

```
vimdiff config_v1.txt config_v2.txt
```

Inside vimdiff:

Command	Action
]c	Next difference
[c	Previous difference
Ctrl + w w	Switch window
do	Copy change from other file
dp	Put change to other file

🔥 REAL-WORLD PRACTICE FLOW (Give Students This Scenario)

You are a DevOps engineer.

1. Check first 5 lines of app.log
2. Check last 10 lines
3. Monitor live logs
4. Search for ERROR
5. Count total lines
6. Compare yesterday and today logs
7. Compare project directories