🛠️ Git Exercise: Navigating a Real-World Repository 🚀

📌 Scenario

Welcome to your first day on the job! Your team has been working on a Python project, but things have gone wrong:

- **Colleagues have deleted critical files.**
- **Some files were renamed incorrectly.**
- **Your boss still expects results!**

Your task is to investigate the repository's history, recover missing files, and get the project running again.

📥 Step 1: Create the repository from the script

Activate the provided script for exercise 2.

```
chmod +x E2_repository.sh
./E2_repository.sh
```

🔍 Step 2: Investigate the Git History

Before making changes, analyze the history to see what went wrong.

```
git log
```

You should see a commit history where files were deleted or renamed. Identify the commits where `prime_number.py` and `fibonacci_number.py` last existed.

To see the history of a specific file:

```
USE CHATGPT!
```

🔄 Step 3: Restore Deleted Files

Once you find the commit where the files last existed, restore them:

```
git checkout <commit-hash> -- prime_number.py
git checkout <commit-hash> -- fibonacci_number.py
```

Verify that they are now back in the repository:

```
ls
```

If everything looks good, **stage and commit the restored files**:

```
git add prime_number.py fibonacci_number.py
git commit -m "Restore missing files"
```

🛠 Step 4: Fix `main.py`

The main script (`main.py`) was modified to use the wrong filenames. Open `main.py` and **replace this incorrect import**:

```
from primes import get_prime_numbers  # Incorrect file name
```

With:

```
from prime_number import get_prime_numbers  # Corrected
```

Check if everything works now:

```
python main.py
```

🚀 Step 5: Commit the changes

Save the file and commit your fix:

```
git add main.py
git commit -m "Fix incorrect import in main.py"
```

If it runs successfully, you should see **two lists printed**:
✅ **Prime Numbers** found in `numbers.csv`
✅ **Fibonacci Numbers** found in `numbers.csv`

If you still get errors, **double-check your commits and file history**!

📚 Additional Git Commands You Might Need

- **Check the status of your working directory:**

```
git status
```

- **View all commits in a detailed log:**

```
git log --graph --all --decorate --oneline
```

- **Restore a deleted file from the last commit:**

```
git restore prime_number.py
```

- **Undo the last commit (before pushing):**

```
git reset --soft HEAD~1
```

💡 What You Learned

✅ How to investigate a Git history.
✅ How to recover deleted files using previous commits.
✅ How to fix incorrect file imports.
✅ How to work in a collaborative Git environment.

🚀 **Well done! You're now more prepared to handle real-world Git issues.**