Linux terminal

Introdução Engenharia Informática

Mário Antunes

September 22, 2025

Exercises

Exercise 1: Finding Your Way Around 🧭

This exercise covers `pwd`, `ls`, `cd`, and basic information commands.

1. Open your terminal. Verify your starting location (your home directory) by printing the working directory.

   ```
   $ pwd
   ```

2. List the contents of your home directory. Then, list them again showing **all** files in the **long** list format.

   ```
   $ ls
   $ ls -la
   ```

3. Navigate to the system log directory at `/var/log` and list its contents.

   ```
   $ cd /var/log
   $ ls
   ```

4. Get some information: find out your username and the current date.

   ```
   $ whoami
   $ date
   ```

5. Return to your home directory using the quickest shortcut.

   ```
   $ cd ~
   ```

Exercise 2: Exploring Key System Directories 🗺️

Reinforce your knowledge of the filesystem layout by visiting important system directories.

1. Navigate to the `/etc` directory, which holds system-wide configuration files.

   ```
   $ cd /etc
   ```

2. List its contents. You'll see many configuration files.

   ```
   $ ls
   ```

3. View the contents of the `os-release` file to see information about your Linux distribution.

   ```
   $ cat os-release
   ```

4. Now, navigate to the `/bin` directory to see where many essential command programs are stored. List its contents and see if you recognize any.

   ```
   $ cd /bin
   $ ls
   ```

Exercise 3: Creating and Managing Files 📂

In this exercise, you'll create, copy, move, and delete files and directories.

1. From your home directory, create a new directory called IEI.

   ```
   $ cd ~
   $ mkdir IEI
   ```

2. Navigate inside your new IEI directory.

   ```
   $ cd IEI
   ```

3. Create an empty file called notes.txt.

   ```
   $ touch notes.txt
   ```

4. Add text to your file and then view its contents.

   ```
   $ echo "My first line of text." > notes.txt
   $ cat notes.txt
   ```

5. Make a copy of your file named notes_backup.txt.

   ```
   $ cp notes.txt notes_backup.txt
   ```

6. Rename notes.txt to important_notes.txt.

   ```
   $ mv notes.txt important_notes.txt
   ```

7. Clean up by deleting the backup file.

   ```
   $ rm notes_backup.txt
   ```

Exercise 4: Understanding Permissions 🔐

This exercise focuses on reading and changing file permissions with chmod.

1. Inside your ~/IEI directory, create a new file called secret_data.txt.

   ```
   $ touch secret_data.txt
   ```

2. View the file's default permissions.

   ```
   $ ls -l secret_data.txt
   ```

3. Remove all permissions for everyone.

   ```
   $ chmod 000 secret_data.txt
   ```

4. Try to view the file's contents. You should get a **"Permission denied"** error.

   ```
   $ cat secret_data.txt
   ```

5. Restore read and write permission for **only yourself**.

   ```
   $ chmod u+rw secret_data.txt
   ```

6. Create an empty script file my_script.sh and make it executable for yourself. Check the permissions afterward to see the change.

   ```
   $ touch my_script.sh
   $ chmod u+x my_script.sh
   $ ls -l my_script.sh
   ```

Exercise 5: Finding Files and Content with `find` and `grep` 🔎

Learn to locate files by name and search for text within them.

1. Inside ~/IEI, create a subdirectory and a new file within it.

   ```
   $ mkdir -p ~/IEI/reports
   $ echo "This is a confidential report." > ~/IEI/reports/report-2025.txt
   ```

2. Use the `find` command to search for any file ending with `.txt` inside your IEI directory.

   ```
   $ find ~/IEI -name "*.txt"
   ```

3. Use `grep` to search for the word "confidential" in your new report file. The `-i` flag makes the search case-insensitive.

   ```
   $ grep -i "confidential" ~/IEI/reports/report-2025.txt
   ```

Exercise 6: Managing Processes ⚙️

Learn how to view and stop running programs from the command line.

1. Start a process that will run in the background. The `sleep` command waits for a specified number of seconds, and the & sends it to the background.

   ```
   $ sleep 120 &
   ```

2. Find the Process ID (PID) of the `sleep` command. You can use `pgrep` for this.

   ```
   $ pgrep sleep
   ```

3. Now, terminate the process using the `kill` command and the PID you just found. Replace PID with the actual number from the previous step.

   ```
   $ kill PID
   ```

4. Verify that the process is no longer running. The `pgrep sleep` command should now return nothing.

   ```
   $ pgrep sleep
   ```

Exercise 7: Managing Software with APT 📦

Let's install and remove a program using the **APT** package manager.

1. First, synchronize your system's package list with the software repositories.

   ```
   $ sudo apt update
   ```

2. Search for a useful command-line tool called `htop`.

   ```
   $ apt search htop
   ```

3. Now, install `htop`. You will need to confirm the installation when prompted.

   ```
   $ sudo apt install htop
   ```

4. Run the program you just installed. Press q to quit.

   ```
   $ htop
   ```

5. Finally, clean up by removing the package from your system.

   ```
   $ sudo apt remove htop
   ```

Exercise 8: Combining Commands 🔗

Let's explore the power of the **pipe (|)** and **redirection (>>)**.

1. The command `ps aux` lists all running processes. Use the pipe (|) to send this output to `grep` to find your own "bash" process.

   ```
   $ ps aux | grep "bash"
   ```

2. Create a log file with one entry.

   ```
   $ echo "$(date): Starting my work." > ~/IEI/activity.log
   ```

3. Use the append operator (>>) to add a second line to the file without deleting the first one.

   ```
   $ echo "$(date): Finished exercise 8." >> ~/IEI/activity.log
   ```

4. Verify that your log file contains both lines.

   ```
   $ cat ~/IEI/activity.log
   ```

Exercise 9: Customizing Your Environment ✨

Time to edit your `.bashrc` file to create a handy shortcut (an alias).

1. Open your `~/.bashrc` file using the `nano` editor.

   ```
   $ nano ~/.bashrc
   ```

2. Scroll to the very bottom and add the following line to create a shortcut `ll` for the command `ls -alF`.

   ```
   alias ll='ls -alF'
   ```

3. Save the file and exit `nano` (Ctrl+X, then Y, then `Enter`).

4. Load the changes into your current session.

   ```
   $ source ~/.bashrc
   ```

5. Test your new alias.

   ```
   $ ll
   ```

Exercise 10: Understanding the $PATH Variable 🛣️

Discover how the shell finds the commands you run.

1. View the current $PATH variable. It's a colon-separated list of directories.

   ```
   $ echo $PATH
   ```

2. Create a simple one-line script in your `~/IEI` directory and make it executable.

   ```
   $ echo '#!/bin/bash' > ~/IEI/hello
   $ echo 'echo "Hello from my custom script!"' >> ~/IEI/hello
   $ chmod +x ~/IEI/hello
   ```

3. Try to run the script by name. It will fail because it's not in a directory listed in $PATH.

   ```
   $ hello
   ```

4. Now run it using its relative path. This works.

   ```
   $ ./hello
   ```

5. Temporarily add your `~/IEI` directory to the $PATH. Now try running the script by name again.

   ```
   $ export PATH="$HOME/IEI:$PATH"
   $ hello
   ```

This change only lasts for your current terminal session.

Exercise 11: Scripting Challenge 🚀

Let's create a script that automates creating a project structure.

1. Create and open a new file named `setup_project.sh` in your ~/IEI directory. Add the following code, then save and close the file.

```bash
#!/bin/bash
PROJECT_DIR="$HOME/IEI/my_project"

if [ -d "$PROJECT_DIR" ]; then
  echo "Error: Directory '$PROJECT_DIR' already exists."
  exit 1
fi

mkdir "$PROJECT_DIR"
echo "Directory '$PROJECT_DIR' created."

for folder in assets source docs
do
  mkdir "$PROJECT_DIR/$folder"
  echo "→ Created subfolder: $folder"
done

echo "Project setup complete!"
```

2. Make the script executable and then run it.

```
$ chmod +x ~/IEI/setup_project.sh
$ ~/IEI/setup_project.sh
```

3. Verify that the directory and its subdirectories were created.

```
$ ls -R ~/IEI/my_project
```

Exercise 12: Scheduling a Task with `cron` 🕐

Let's create a simple script and schedule it to run automatically every minute.

1. **Create the Script:** In your ~/IEI directory, create a script named `log_time.sh` with the following content.

```bash
#!/bin/bash
date >> $HOME/IEI/cron_log.txt
```

2. **Make it Executable:**

```
$ chmod +x ~/IEI/log_time.sh
```

3. **Open your Crontab:** This will open a text editor.

```
$ crontab -e
```

4. **Add the Cron Job:** Go to the bottom of the file and add the following line. You must use the full, absolute path to your script.

```
* * * * * /home/student/IEI/log_time.sh
```

5. **Save and Verify:** Save and exit the editor. Wait two minutes, then check your log file. You should see two timestamp entries.

```
$ cat ~/IEI/cron_log.txt
```

6. **Clean Up:** It's very important to remove the cron job so it doesn't run forever. This command removes your entire crontab file.

```
$ crontab -r
```