

Windows terminal

Tópicos de Informática para Automação

Mário Antunes

September 29, 2025

Exercises

Exercise 1: Finding Your Way Around □

This exercise covers basic navigation and information commands. Complete the steps in both **CMD** and **PowerShell**.

1. Open your terminal. Verify your starting location (your home directory).
 - **CMD:** \$ cd
 - **PowerShell:** \$ Get-Location (or its alias pwd)
 2. List the contents of your home directory. Then, list them again showing **all** files (including hidden ones).
 - **CMD:** \$ dir then \$ dir /a
 - **PowerShell:** \$ ls then \$ ls -Force
 3. Navigate to the main Windows directory.
 - **CMD & PowerShell:** \$ cd C:\Windows
 4. Find out your username and the current date.
 - **CMD:** \$ whoami and then \$ date /t
 - **PowerShell:** \$ whoami and then \$ Get-Date
 5. Return to your home directory using the quickest shortcut.
 - **CMD:** \$ cd %USERPROFILE%
 - **PowerShell:** \$ cd ~
-

Exercise 2: Exploring Key System Directories □

Visit important system directories to understand the Windows layout.

1. Navigate to the System32 directory, which holds most of the core system executables.
 - **CMD & PowerShell:** \$ cd C:\Windows\System32
 2. List its contents to see the vast number of system files.
 - **CMD:** \$ dir
 - **PowerShell:** \$ ls
 3. Get information about your Windows version.
 - **CMD:** \$ systeminfo | findstr /B /C:"OS Name" /C:"OS Version"
 - **PowerShell:** \$ Get-ComputerInfo | Select-Object OSName, OSVersion
-

Exercise 3: Creating and Managing Files □

Create, copy, move, and delete files and directories.

1. From your home directory, create a new directory called TIA.
 - **CMD & PowerShell:** \$ mkdir TIA
2. Navigate inside your new TIA directory.
 - **CMD & PowerShell:** \$ cd TIA
3. Create an empty file called notes.txt.

- **CMD:** \$ echo. > notes.txt
 - **PowerShell:** \$ New-Item notes.txt
4. Add text to your file and then view its contents.
 - **CMD:** \$ echo My first line. > notes.txt then \$ type notes.txt
 - **PowerShell:** \$ Set-Content -Path notes.txt -Value "My first line." then \$ Get-Content notes.txt
 5. Make a copy of the file named notes_backup.txt.
 - **CMD:** \$ copy notes.txt notes_backup.txt
 - **PowerShell:** \$ Copy-Item notes.txt notes_backup.txt
 6. Rename notes.txt to important_notes.txt.
 - **CMD:** \$ ren notes.txt important_notes.txt
 - **PowerShell:** \$ Rename-Item notes.txt important_notes.txt
 7. Clean up by deleting the backup file.
 - **CMD:** \$ del notes_backup.txt
 - **PowerShell:** \$ Remove-Item notes_backup.txt
-

Exercise 4: Understanding File Attributes □

Windows permissions can be complex. This exercise focuses on a simpler concept: the **read-only attribute**.

1. Inside ~/TIA, create a file named report.docx.
 - **CMD:** \$ echo. > report.docx
 - **PowerShell:** \$ New-Item report.docx
 2. Set the file to be read-only.
 - **CMD:** \$ attrib +r report.docx
 - **PowerShell:** \$ Set-ItemProperty -Path report.docx -Name IsReadOnly -Value \$true
 3. Attempt to delete the file. The operation should fail or ask for confirmation because the file is read-only.
 - **CMD:** \$ del report.docx
 - **PowerShell:** \$ Remove-Item report.docx
 4. Remove the read-only attribute so you can manage the file again.
 - **CMD:** \$ attrib -r report.docx
 - **PowerShell:** \$ Set-ItemProperty -Path report.docx -Name IsReadOnly -Value \$false
-

Exercise 5: Finding Files and Content □

Search for files by name and for text within them.

1. Create a subdirectory and a new file within it.
 - **CMD:** \$ mkdir reports and then \$ echo Confidential report. > reports\report-2025.txt
 - **PowerShell:** \$ mkdir reports and then \$ Set-Content reports\report-2025.txt "Confidential report."
 2. Use the appropriate command to search for any file ending with .txt inside your TIA directory and its subdirectories.
 - **CMD:** \$ dir /s /b *.txt
 - **PowerShell:** \$ Get-ChildItem -Recurse -Filter "*.txt"
 3. Search for the word "Confidential" inside the reports directory.
 - **CMD:** \$ findstr /i "Confidential" reports*
 - **PowerShell:** \$ Select-String -Path reports* -Pattern "Confidential"
-

Exercise 6: Managing Processes □

Learn to view and stop running programs.

1. Start a Notepad process from the terminal.
 - **CMD & PowerShell:** `$ notepad`
 2. In the same terminal, find the Process ID (PID) of Notepad.
 - **CMD:** `$ tasklist | findstr /i "notepad"`
 - **PowerShell:** `$ Get-Process -Name "notepad"`
 3. Terminate the Notepad process using its PID. Replace PID with the actual number from the previous step.
 - **CMD:** `$ taskkill /PID PID`
 - **PowerShell:** `$ Stop-Process -Id PID`
-

Exercise 7: Managing Software with Winget

Install and remove a program using the **Windows Package Manager**. These commands work in both CMD and PowerShell.

1. Search for the popular 7zip utility.
 - `$ winget search 7zip`
 2. Install the package. You may need to agree to the source terms.
 - `$ winget install 7zip.7zip`
 3. List all your installed packages managed by Winget to verify the installation.
 - `$ winget list`
 4. Clean up by removing the package from your system.
 - `$ winget uninstall 7zip.7zip`
-

Exercise 8: Combining Commands (Pipes & Redirection)

Explore the power of the pipe (|) and redirection (>>).

1. Use the pipe to find your own terminal process ("cmd.exe" or "powershell.exe").
 - **CMD:** `$ tasklist | findstr "cmd.exe"`
 - **PowerShell:** `$ Get-Process | Where-Object { $_.Name -eq "powershell" }`
 2. Create a log file with one entry using >.
 - **CMD:** `$ echo %date% %time%: Starting work. > activity.log`
 - **PowerShell:** `$ Set-Content activity.log "$(Get-Date): Starting work."`
 3. Use the append operator (>>) to add a second line without deleting the first.
 - **CMD:** `$ echo %date% %time%: Finished exercise. >> activity.log`
 - **PowerShell:** `$ Add-Content activity.log "$(Get-Date): Finished exercise."`
 4. Verify that your log file contains both lines.
 - **CMD:** `$ type activity.log`
 - **PowerShell:** `$ Get-Content activity.log`
-

Exercise 9: Customizing Your Environment

Create a handy shortcut (an alias).

• CMD (Temporary Alias):

1. Create an alias `ll` for the `dir /a` command using `doskey`. `$ doskey ll=dir /a`
2. Test your alias: `$ ll` (Note: This alias disappears when you close the CMD window.)

• PowerShell (Permanent Alias):

1. Open your PowerShell profile script in Notepad. `if (!(Test-Path -Path $PROFILE)) { New-Item -ItemType File -Path $PROFILE -Force } $ notepad $PROFILE`
 2. Add the following line to the file, then save and close it. `Set-Alias -Name ll -Value Get-ChildItem -Force`
 3. Close and reopen PowerShell, then test your new alias: `$ ll`
-

Exercise 10: Understanding the \$PATH Variable

Discover how the shell finds commands.

1. View the current \$PATH variable.
 - **CMD:** `$ echo %PATH%`
 - **PowerShell:** `$ echo $env:Path`
 2. Create a simple Batch file in your ~/TIA directory named hello.bat containing one line: @echo Hello from my custom script!
 3. Try to run the script by name. It will fail because TIA is not in the \$PATH.
 - **CMD & PowerShell:** `$ hello`
 4. Run it using its relative path. This works.
 - **CMD & PowerShell:** `$.\hello.bat`
 5. Temporarily add your ~/TIA directory to the \$PATH. Now try running it by name again.
 - **CMD:** `$ set PATH=%USERPROFILE%\TIA;%PATH% then $ hello.bat`
 - **PowerShell:** `$ $env:Path = "$HOME\TIA;" + $env:Path then $ hello.bat`
-

Exercise 11: Scripting Challenge

Create a script that automates setting up a project structure.

• CMD (Batch Script):

1. Create a file named setup_project.bat in ~/TIA.
2. Add the following code, then save it. cmd @echo off set PROJECT_DIR=%USERPROFILE%\TIA if exist %PROJECT_DIR% (echo Error: Directory already exists. exit /b) mkdir %PROJECT_DIR% mkdir %PROJECT_DIR%\assets mkdir %PROJECT_DIR%\source mkdir %PROJECT_DIR%\docs echo Project setup complete!
3. Run the script: `$.\setup_project.bat`

• PowerShell Script:

1. Create a file named setup_project.ps1 in ~/TIA.
 2. Add the following code, then save it. powershell \$ProjectDir = "\$HOME\TIA\my_project" if (Test-Path \$ProjectDir) { Write-Error "Directory already exists." return } mkdir \$ProjectDir foreach (\$folder in "assets", "source", "docs") { mkdir (Join-Path \$ProjectDir \$folder) } Write-Host "Project setup complete!"
 3. Run the script: `$.\setup_project.ps1`
-

Exercise 12: Scheduling a Task

Create a simple script and schedule it to run automatically.

1. **Create the Script:** In ~/TIA, create log_time.bat with the content: cmd @echo off @echo %date% %time:~0,5% >> %USERPROFILE%\TIA\cron_log.txt
2. **Schedule the Task (CMD):**
 - This command schedules the script to run in one minute from now (and periodically at one minute):
`$ schtasks /create /sc minute /tn "My Logger" ^
/tr "%USERPROFILE%\TIA\log_time.bat" /st %time:~0,5%`
3. **Schedule the Task (PowerShell):**
 - This command schedules the script to run in one minute from now (and periodically at one minute):
`$action = New-ScheduledTaskAction -Execute "$env:USERPROFILE\TIA\log_time.bat"
$trigger = New-ScheduledTaskTrigger -At $(Get-Date -Format HH:mm) -Once `~
-RepetitionInterval (New-TimeSpan -Minutes 1)
Register-ScheduledTask -Action $action -Trigger $trigger -TaskName "My Logger"`
4. **Verify:** After a minute, check for the output file (should have repeated lines).

- **CMD:** \$ type %USERPROFILE%\TIA\cron_log.txt
 - **PowerShell:** \$ Get-Content \$env:USERPROFILE\TIA\cron_log.txt
5. **Clean Up:** It's important to remove the task so it doesn't remain in the system.
- **CMD:** \$ schtasks /delete /tn "My Logger" /f
 - **PowerShell:** \$ Unregister-ScheduledTask "My Logger"