

Terminal do Windows

Introdução Engenharia Informática

Mário Antunes

September 29, 2025

Exercícios

Exercício 1: A Orientar-se 🧭

Este exercício abrange comandos básicos de navegação e informação. Complete os passos tanto no **CMD** como no **PowerShell**.

1. Abra o seu terminal. Verifique a sua localização inicial (o seu diretório pessoal).
 - **CMD:** `$ cd`
 - **PowerShell:** `$ Get-Location` (ou o seu alias `pwd`)
 2. Liste o conteúdo do seu diretório pessoal. De seguida, liste-o novamente mostrando **todos** os ficheiros (incluindo os ocultos).
 - **CMD:** `$ dir` e depois `$ dir /a`
 - **PowerShell:** `$ ls` e depois `$ ls -Force`
 3. Navegue para o diretório principal do Windows.
 - **CMD & PowerShell:** `$ cd C:\Windows`
 4. Descubra o seu nome de utilizador e a data atual.
 - **CMD:** `$ whoami` e depois `$ date /t`
 - **PowerShell:** `$ whoami` e depois `$ Get-Date`
 5. Regresse ao seu diretório pessoal usando o atalho mais rápido.
 - **CMD:** `$ cd %USERPROFILE%`
 - **PowerShell:** `$ cd ~`
-

Exercício 2: A Explorar Diretórios Chave do Sistema 🗺️

Visite diretórios importantes do sistema para compreender a estrutura do Windows.

1. Navegue para o diretório `System32`, que contém a maioria dos executáveis centrais do sistema.
 - **CMD & PowerShell:** `$ cd C:\Windows\System32`
 2. Liste o seu conteúdo para ver o vasto número de ficheiros de sistema.
 - **CMD:** `$ dir`
 - **PowerShell:** `$ ls`
 3. Obtenha informação sobre a sua versão do Windows.
 - **CMD:** `$ systeminfo | findstr /B /C:"OS Name" /C:"OS Version"`
 - **PowerShell:** `$ Get-ComputerInfo | Select-Object OSName, OSVersion`
-

Exercício 3: A Criar e Gerir Ficheiros 📁

Crie, copie, mova e apague ficheiros e diretórios.

1. A partir do seu diretório pessoal, crie um novo diretório chamado `IEI`.
 - **CMD & PowerShell:** `$ mkdir IEI`
2. Navegue para dentro do seu novo diretório `IEI`.
 - **CMD & PowerShell:** `$ cd IEI`
3. Crie um ficheiro vazio chamado `notes.txt`.

- **CMD:** `$ echo. > notes.txt`
 - **PowerShell:** `$ New-Item notes.txt`
4. Adicione texto ao seu ficheiro e depois veja o seu conteúdo.
 - **CMD:** `$ echo A minha primeira linha. > notes.txt` e depois `$ type notes.txt`
 - **PowerShell:** `$ Set-Content -Path notes.txt -Value "A minha primeira linha."` e depois `$ Get-Content notes.txt`
 5. Faça uma cópia do ficheiro com o nome `notes_backup.txt`.
 - **CMD:** `$ copy notes.txt notes_backup.txt`
 - **PowerShell:** `$ Copy-Item notes.txt notes_backup.txt`
 6. Renomeie `notes.txt` para `important_notes.txt`.
 - **CMD:** `$ ren notes.txt important_notes.txt`
 - **PowerShell:** `$ Rename-Item notes.txt important_notes.txt`
 7. Faça a limpeza, apagando o ficheiro de *backup*.
 - **CMD:** `$ del notes_backup.txt`
 - **PowerShell:** `$ Remove-Item notes_backup.txt`
-

Exercício 4: A Compreender Atributos de Ficheiros

As permissões no Windows podem ser complexas. Este exercício foca-se num conceito mais simples: o **atributo de só de leitura**.

1. Dentro de `~/IEI`, crie um ficheiro chamado `report.docx`.
 - **CMD:** `$ echo. > report.docx`
 - **PowerShell:** `$ New-Item report.docx`
 2. Defina o ficheiro como só de leitura.
 - **CMD:** `$ attrib +r report.docx`
 - **PowerShell:** `$ Set-ItemProperty -Path report.docx -Name IsReadOnly -Value $true`
 3. Tente apagar o ficheiro. A operação deverá falhar ou pedir confirmação porque o ficheiro é só de leitura.
 - **CMD:** `$ del report.docx`
 - **PowerShell:** `$ Remove-Item report.docx`
 4. Remova o atributo de só de leitura para poder gerir o ficheiro novamente.
 - **CMD:** `$ attrib -r report.docx`
 - **PowerShell:** `$ Set-ItemProperty -Path report.docx -Name IsReadOnly -Value $false`
-

Exercício 5: A Encontrar Ficheiros e Conteúdo

Procure por ficheiros por nome e por texto dentro deles.

1. Crie um subdiretório e um novo ficheiro dentro dele.
 - **CMD:** `$ mkdir reports` e depois `$ echo Relatorio confidencial. > reports\report-2025.txt`
 - **PowerShell:** `$ mkdir reports` e depois `$ Set-Content reports\report-2025.txt "Relatorio confidencial."`
 2. Use o comando apropriado para procurar qualquer ficheiro que termine em `.txt` dentro do seu diretório `IEI` e dos seus subdiretórios.
 - **CMD:** `$ dir /s /b *.txt`
 - **PowerShell:** `$ Get-ChildItem -Recurse -Filter "*.txt"`
 3. Procure pela palavra "Confidencial" dentro do diretório `reports`.
 - **CMD:** `$ findstr /i "Confidencial" reports*`
 - **PowerShell:** `$ Select-String -Path reports* -Pattern "Confidencial"`
-

Exercício 6: A Gerir Processos

Aprenda a ver e a parar programas em execução.

1. Inicie um processo do Notepad a partir do terminal.

- **CMD & PowerShell:** \$ notepad
2. No mesmo terminal, encontre o ID do Processo (PID) do Notepad.
 - **CMD:** \$ tasklist | findstr /i "notepad"
 - **PowerShell:** \$ Get-Process -Name "notepad"
 3. Termine o processo do Notepad usando o seu PID. Substitua PID pelo número real do passo anterior.
 - **CMD:** \$ taskkill /PID PID
 - **PowerShell:** \$ Stop-Process -Id PID
-

Exercício 7: A Gerir Software com o Winget 📦

Instale e remova um programa usando o **Windows Package Manager**. Estes comandos funcionam tanto no CMD como no PowerShell.

1. Procure pelo popular utilitário 7zip.
 - \$ winget search 7zip
 2. Instale o pacote. Pode ser necessário concordar com os termos da fonte.
 - \$ winget install 7zip.7zip
 3. Liste todos os seus pacotes instalados geridos pelo Winget para verificar a instalação.
 - \$ winget list
 4. Faça a limpeza, removendo o pacote do seu sistema.
 - \$ winget uninstall 7zip.7zip
-

Exercício 8: A Combinar Comandos (Pipes & Redirecionamento) 🔗

Explore o poder do *pipe* (|) e do redirecionamento (>>).

1. Use o *pipe* para encontrar o seu próprio processo de terminal ("cmd.exe" ou "powershell.exe").
 - **CMD:** \$ tasklist | findstr "cmd.exe"
 - **PowerShell:** \$ Get-Process | Where-Object { \$_.Name -eq "powershell" }
 2. Crie um ficheiro de *log* com uma entrada usando >.
 - **CMD:** \$ echo %date% %time%: A iniciar trabalho. > activity.log
 - **PowerShell:** \$ Set-Content activity.log "\$(Get-Date): A iniciar trabalho."
 3. Use o operador de acréscimo (>>) para adicionar uma segunda linha sem apagar a primeira.
 - **CMD:** \$ echo %date% %time%: Exercício terminado. >> activity.log
 - **PowerShell:** \$ Add-Content activity.log "\$(Get-Date): Exercício terminado."
 4. Verifique se o seu ficheiro de *log* contém ambas as linhas.
 - **CMD:** \$ type activity.log
 - **PowerShell:** \$ Get-Content activity.log
-

Exercício 9: A Personalizar o Seu Ambiente ✨

Crie um atalho útil (um *alias*).

• **CMD (Alias Temporário):**

1. Crie um *alias* ll para o comando dir /a usando o doskey. \$ doskey ll=dir /a
2. Teste o seu *alias*: \$ ll (Nota: Este *alias* desaparece quando fecha a janela do CMD.)

• **PowerShell (Alias Permanente):**

1. Abra o seu *script* de perfil do PowerShell no Notepad.

```
if (!(Test-Path -Path $PROFILE)) { New-Item -ItemType File -Path $PROFILE -Force }
$ notepad $PROFILE
```

2. Adicione a seguinte linha ao ficheiro, depois guarde e feche-o. Set-Alias -Name ll -Value Get-ChildItem -Force
 3. Feche e reabra o PowerShell, e depois teste o seu novo *alias*: \$ ll
-

Exercício 10: A Compreender a Variável \$PATH 🏡

Descubra como a *shell* encontra os comandos.

1. Veja a variável \$PATH atual.
 - **CMD:** `$ echo %PATH%`
 - **PowerShell:** `$ echo $env:Path`
 2. Crie um ficheiro *Batch* simples no seu diretório ~/IEI chamado `hello.bat` contendo uma linha: `@echo Ola do meu script personalizado!`
 3. Tente executar o *script* pelo nome. Irá falhar porque IEI não está na \$PATH.
 - **CMD & PowerShell:** `$ hello`
 4. Execute-o usando o seu caminho relativo. Isto funciona.
 - **CMD & PowerShell:** `$.\hello.bat`
 5. Adicione temporariamente o seu diretório ~/IEI à \$PATH. Agora tente executá-lo pelo nome novamente.
 - **CMD:** `$ set PATH=%USERPROFILE%\IEI;%PATH% e depois $ hello.bat`
 - **PowerShell:** `$ $env:Path = "$HOME\IEI;" + $env:Path e depois $ hello.bat`
-

Exercício 11: Desafio de Scripting 🚀

Crie um *script* que automatiza a criação de uma estrutura de projeto.

• CMD (Script Batch):

1. Crie um ficheiro chamado `setup_project.bat` em ~/IEI.
2. Adicione o seguinte código e guarde-o.

```
@echo off
set PROJECT_DIR=%USERPROFILE%\IEI\my_project
if exist %PROJECT_DIR% (
    echo Erro: O directorio ja existe.
    exit /b
)
mkdir %PROJECT_DIR%
mkdir %PROJECT_DIR%\assets
mkdir %PROJECT_DIR%\source
mkdir %PROJECT_DIR%\docs
echo Configuracao do projeto concluida!
```

3. Execute o *script*: `$.\setup_project.bat`

• Script PowerShell:

1. Crie um ficheiro chamado `setup_project.ps1` em ~/IEI.
2. Adicione o seguinte código e guarde-o.

```
$ProjectDir = "$HOME\IEI\my_project"
if (Test-Path $ProjectDir) {
    Write-Error "O directorio ja existe."
    return
}
mkdir $ProjectDir
foreach ($folder in "assets", "source", "docs") {
    mkdir (Join-Path $ProjectDir $folder)
}
Write-Host "Configuracao do projeto concluida!"
```

3. Execute o *script*: `$.\setup_project.ps1`
-

Exercício 12: A Agendar uma Tarefa

Crie um *script* simples e agende a sua execução automática.

1. **Crie o Script:** Em ~/IEI, crie o ficheiro `log_time.bat` com o seguinte conteúdo:

```
@echo off
@echo %date% %time:~0,5% >> %USERPROFILE%\IEI\cron_log.txt
```

2. **Agende a Tarefa (CMD):**

- Este comando agenda a execução do *script* para daqui a um minuto (periodico a um minuto):

```
$ schtasks /create /sc minute /tn "My Logger" ^
/tr "%USERPROFILE%\IEI\log_time.bat" /st %time:~0,5%
```

3. **Agende a Tarefa (PowerShell):**

- Este comando agenda a execução do *script* para daqui a um minuto (periodico a um minuto):

```
$action = New-ScheduledTaskAction -Execute "$env:USERPROFILE\IEI\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. **Verifique:** Após um minuto, verifique o ficheiro de *output*.

- **CMD:** \$ type %USERPROFILE%\IEI\cron_log.txt
- **PowerShell:** \$ Get-Content \$env:USERPROFILE\IEI\cron_log.txt

5. **Limpeza:** É importante remover a tarefa para que não permaneça no sistema.

- **CMD:** \$ schtasks /delete /tn "My Logger" /f
- **PowerShell:** \$ Unregister-ScheduledTask "My Logger"