Operating Systems Writeup

SETU - South East Technological University
Imbriani Paolo - W20114452
Professor Micheal McMahon

February 12, 2025

Contents

1 Practical 1 - Commands

Taking a selection of Windows CLI commands from those given below, use the online help to examine the various options and arguments, and try them out. You're required carefully to write two A4 pages (Times 12 point or equivalent size) detailing your experiments with different options for between six and ten different commands. To get the online help for a command, type command /?
e.g.
dir /?
prompt
mkdir
color
title
tree
type
ver
print
xcopy
Type help at the windows command line prompt to see some more instructions

- Prompt - The prompt command is used to customize the text that appears before the cursor in the command prompt.

```
prompt MyPrompt$G
```

  This changes the prompt to MyPrompt>. The $G represents the > symbol.

- Mkdir - The mkdir command is used to create a new directory.

```
mkdir MyDirectory
```

  This creates a new directory called MyDirectory. To create a folder inside another folder:

```
mkdir MyDirectory\MySubDirectory
```

- Color - The color command is used to change the color of the text and background in the command prompt.

```
color 0A
```

  This sets a black background (0) with green text (A). To reset to default:

```
color
```

  To see all the available colors:

```
color /?
```

- Title - The title command is used to change the title of the command prompt window.

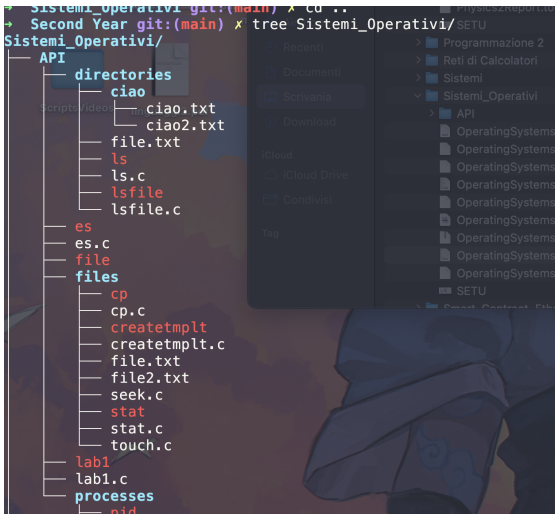
```
title MyTitle
```

  This changes the title of the command prompt window to MyTitle.

- Tree - The tree command is used to display a graphical representation of the directory structure.

```
tree
```

And it will output something like this:



Displays a simple tree structure of folders in the current directory. To include all files in the display:

```
tree /f
```

The /F option lists all files along with the folder structure.

- Type - The type command is used to display the contents of a text file.

```
type MyFile.txt
```

This displays the contents of the file MyFile.txt. Useful for quickly viewing small text files without opening them.

- Ver - The ver command is used to display the version of the operating system.

```
ver
```

This displays the version of the operating system.

What's the purpose of the first line - @ECHO OFF? Remove it and see the effect

```
@ECHO OFF
ECHO Please insert a USB memory stick
PAUSE
COPY *.txt I:\
ECHO BACKUP COMPLETE
```

- @ECHO OFF → Hides command execution lines for cleaner output.

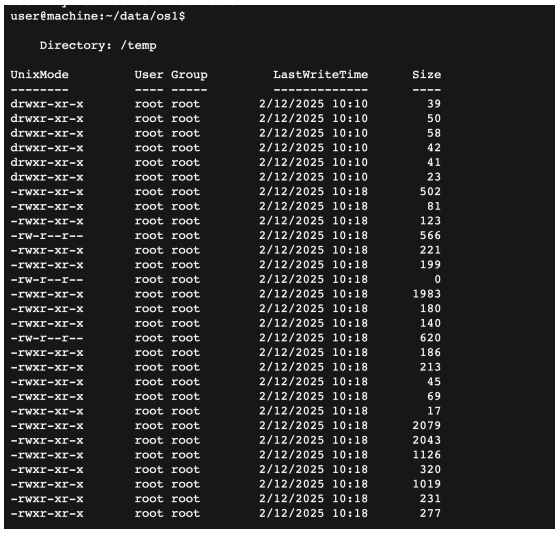
- ECHO → Displays messages on the screen.
- PAUSE → Waits for the user to press a key before continuing.
- COPY *.txt I: → Copies all .txt files from the current folder to the USB drive (assuming it's drive I:).
- ECHO BACKUP COMPLETE → Displays a completion message.

The first line, @ECHO OFF, is used to prevent the command prompt from displaying each command as it executes.

```
C:\Users\YourName\Desktop> ECHO Please insert a USB memory stick
Please insert a USB memory stick

C:\Users\YourName\Desktop> PAUSE
Press any key to continue . . .

C:\Users\YourName\Desktop> COPY *.txt I:\
3 file(s) copied.

C:\Users\YourName\Desktop> ECHO BACKUP COMPLETE
BACKUP COMPLETE
```

2 Practical 2 - Powershell

2.1 Test the above using a directory that you created last week.



```
user@machine:~/data/os1$

    Directory: /temp

UnixMode        User Group        LastWriteTime     Size
--------        ---- -----        -------------     ----
drwxr-xr-x      root root       2/12/2025 10:10       39
drwxr-xr-x      root root       2/12/2025 10:10       50
drwxr-xr-x      root root       2/12/2025 10:10       58
drwxr-xr-x      root root       2/12/2025 10:10       42
drwxr-xr-x      root root       2/12/2025 10:10       41
drwxr-xr-x      root root       2/12/2025 10:10       23
-rwxr-xr-x      root root       2/12/2025 10:18      502
-rwxr-xr-x      root root       2/12/2025 10:18       81
-rwxr-xr-x      root root       2/12/2025 10:18      123
-rw-r--r--      root root       2/12/2025 10:18      566
-rwxr-xr-x      root root       2/12/2025 10:18      221
-rwxr-xr-x      root root       2/12/2025 10:18      199
-rw-r--r--      root root       2/12/2025 10:18        0
-rwxr-xr-x      root root       2/12/2025 10:18     1983
-rwxr-xr-x      root root       2/12/2025 10:18      180
-rwxr-xr-x      root root       2/12/2025 10:18      140
-rw-r--r--      root root       2/12/2025 10:18      620
-rwxr-xr-x      root root       2/12/2025 10:18      186
-rwxr-xr-x      root root       2/12/2025 10:18      213
-rwxr-xr-x      root root       2/12/2025 10:18       45
-rwxr-xr-x      root root       2/12/2025 10:18       69
-rwxr-xr-x      root root       2/12/2025 10:18       17
-rwxr-xr-x      root root       2/12/2025 10:18     2079
-rwxr-xr-x      root root       2/12/2025 10:18     2043
-rwxr-xr-x      root root       2/12/2025 10:18     1126
-rwxr-xr-x      root root       2/12/2025 10:18      320
-rwxr-xr-x      root root       2/12/2025 10:18     1019
-rwxr-xr-x      root root       2/12/2025 10:18      231
-rwxr-xr-x      root root       2/12/2025 10:18      277
```

Using the directory was created last week, we can use the Get-ChildItem command to list the contents of the directory. using the following list of commands:

```
$mylist = dir
$mylist
$mylist[0].Name
$mylist[0].length
$mylist[0].Mode
$mylist[0].LastWriteTime
```

Exercise of bash scripting in Powershell:

- Simply record what happens when you run this script.
- Find out how to run scripts if they're in a directory other than the working directory.
- Study the following script and see if you can figure out what it'll do. Now type it into a file called game.ps1 (You can use copy/paste in places to reduce the labour.) Run it and see if your predictions are true.

```
#############################################################
#############################################################
##
## The three pathetic knock-knock jokes program!
## Date: 26/09/17
## For: BSc (Hons) Computer Forensics and Security
##
#############################################################
#############################################################

#############################
## initialisation section
#############################
$userReply = ""
#############################
## first question
#############################
Clear-Host

while($userReply -ne "Who is there?"){
    $userReply = read-host "Knock Knock!"
}
Clear-Host
while($userReply -ne "Orange who?"){
    $userReply = read-host "Orange"
}
Clear-Host
Write-Output "Orange you glad you created this PowerShell script?"
Start-Sleep -Seconds 5
#############################
## Second Question
#############################
Clear-Host
while($userReply -ne "Who is there?"){
    $userReply = read-host "Knock Knock!"
}
Clear-Host
while($userReply -ne "Orange who?"){
    $userReply = read-host "Orange"
}
```

```
Clear-Host
Write-Output "Oranges are oranges but this is PowerShellscripting!"
Start-Sleep -Seconds 5
############################
## Third Question
############################
Clear-Host
while($userReply -ne "Who is there?"){
    $userReply = read-host "Knock Knock!"
}
Clear-Host
while($userReply -ne "Banana who?"){
    $userReply = read-host "Banana"
}
Clear-Host
Write-Output "Orange you glad I didn't say orange?"
Start-Sleep -Seconds 5
############################
## Farewell Message
############################
Clear-Host
Write-Output "Goodbye!nn"
```

When you run game.ps1, here's the interactive sequence you'll experience:

1. The script will prompt you with "Knock Knock!"
2. You'll need to respond with "Who is there?"
3. The script will then prompt you with "Orange"
4. Until you respond with "Orange who?"
5. The script will then output "Orange you glad you created this PowerShell script?"
6. The second and third questions follow a similar pattern.
7. The script will then output "Goodbye!"

To do:

1. Create another file called beverage.txt identical to drink.txt. (Hint: Use what you learned in practical 1 to do this.)
2. Issue this command again
   Get-ChildItem — select-string coffee
   and see what happens.
3. Create a file called fruit.txt with the list apple, orange, banana in it.
4. Issue this command again
   Get-ChildItem — select-string coffee
   and see what happens.

```
PS C:\Users\User1\green> Copy-Item drink.txt beverage.txt
PS C:\Users\User1\green> Get-ChildItem *.txt
Directory: C:\Users\User1\green

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         2/10/2025   10:15 AM            123 drink.txt
-a----         2/10/2025   10:15 AM            123 beverage.txt
PS C:\Users\User1\green> Get-ChildItem | Select-String coffee
drink.txt:1:coffee.
beverage.txt:1:coffee.
PS C:\Users\User1\green> Set-Content fruit.txt -Value "apple`r`norange`r`nbanana"
PS C:\Users\User1\green> Get-Content fruit.txt
apple
orange
banana
PS C:\Users\User1\green> Get-ChildItem | Select-String coffee
drink.txt:1:coffee.
beverage.txt:1:coffee.
```

Find out and explain how to get help about any cmdlet.

- Find out and explain what F7 does in PowerShell.
- What is the purposes of (a) the –whatif switch and (b) the –confirm
- Write a note to explain how you can use tab to complete a command as soon as it's unambiguous.

In PowerShell, you can obtain help for any cmdlet by typing Get-Help followed by the name of the cmdlet.

```
Get-Help Get-Process
```

This will display detailed information about the Get-Process cmdlet, including a description, syntax, parameters, examples, and more. When you press F7 in a PowerShell console, it brings up a graphical popup window displaying your command history from the current session. You can use the arrow keys to navigate through the list of previously executed commands. This is a handy feature to quickly recall and reuse commands without retyping them.
The -WhatIf switch is used to simulate the execution of a command. It shows you what would happen if the command ran but does not make any actual changes. Use -WhatIf with potentially destructive or impactful commands to verify what actions would be performed.

```
Remove-Item C:\Temp\* -WhatIf
```

The -Confirm switch forces the command to prompt for your confirmation before executing each action. This extra safety measure helps prevent accidental changes.

```
Remove-Item C:\Temp\* -Confirm
```

PowerShell supports intelligent tab completion. As you start typing a command, cmdlet name, parameter, or even file path, you can press the Tab key to auto-complete the text. If the text you've entered uniquely identifies a command or parameter, pressing Tab will automatically complete it. If **multiple completions are possible repeatedly pressing Tab cycles through the available options until you reach the one you want.**