**Name – Jyoti Sahebrao Jagtap**

**Email** [**–jyotishinge60@gmail.com**](mailto:–jyotishinge60@gmail.com)

**1. Cmdlets** – Cmdlets are small, built-in PowerShell commands that follow a Verb-Noun pattern (e.g., Get-Process). They perform a single task and return rich .NET objects instead of plain text.

**2. The PowerShell Pipeline** – Lets you pass the output of one command directly to another, allowing step-by-step processing without temporary storage.

**3. Key Cmdlets** – Core commands like Get-Help, Get-Command, Get-Service, and Get-Process form the base toolkit for system management in PowerShell.

**4. WMI & PowerShell** – WMI (Windows Management Instrumentation) allows querying and managing OS, hardware, and application data. Accessed via cmdlets like Get-CimInstance.

**5. Pipeline Filtering & Operators** – Use filters like Where-Object and comparison operators (-eq, -gt, -like) to refine results from the pipeline.

**6. Input, Output & Formatting** – Handle user input (Read-Host), output results to screen/files, and format data using Format-Table, Format-List, or Out-File.

**7. Scripting Overview** – Write .ps1 scripts to automate repetitive tasks. Include variables, loops, conditions, and functions for flexibility.

**8. Objects, Arrays, Variables** – PowerShell works with objects (data + properties), arrays (lists of items), and variables ($varName) for storing data during execution.

**9. Scope** – Defines where variables, functions, and scripts are accessible. Scopes include Global, Script, Local, and Private.

**10. More Operators** – In addition to comparison operators, PowerShell has logical (-and, -or), assignment (=, +=), redirection (>), and type (-is, -as) operators.

**11. Scripting Constructs** – Control flow structures like if-else, switch, for, foreach, and while allow complex logic in scripts.

**12. Modularization** – Breaking scripts into reusable modules (.psm1 files) makes them easier to maintain, share, and load with Import-Module.

**13. Understand Basic Platform Value, Concepts, and Usage** – Recognize PowerShell as a cross-platform automation tool built on .NET, designed for configuration, management, and data handling in Windows, Linux, and macOS environments.

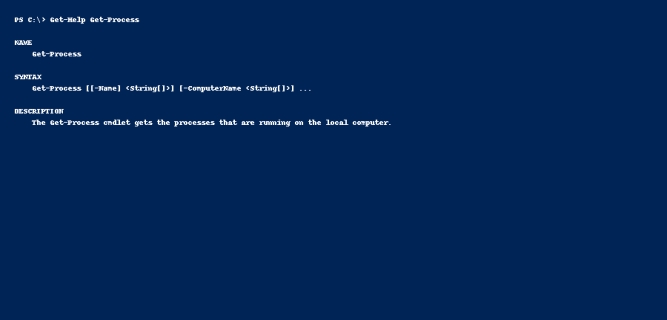
Cmdlets Example

|  |
| --- |
|  |
|  | C:\Users\LENOVO\AppData\Local\Temp\ksohtml3152\wps22.jpg |

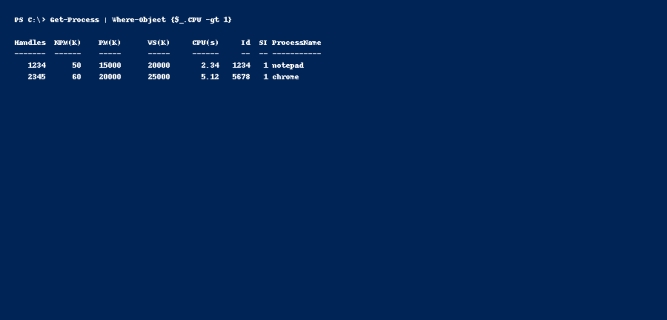
Pipeline Example

|  |
| --- |
|  |
|  | C:\Users\LENOVO\AppData\Local\Temp\ksohtml3152\wps23.jpg |

Key Cmdlets

WMI & PowerShell

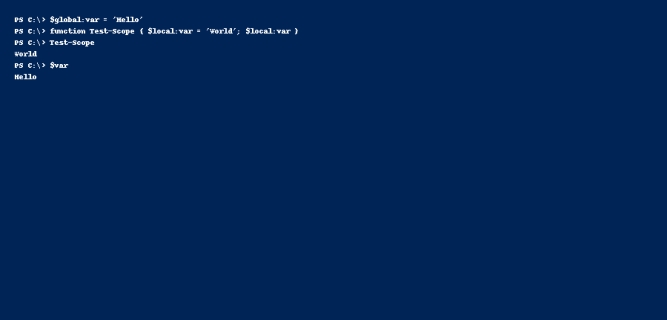
|  |
| --- |
|  |
|  | C:\Users\LENOVO\AppData\Local\Temp\ksohtml3152\wps25.jpg |

Input, Output & Formatting

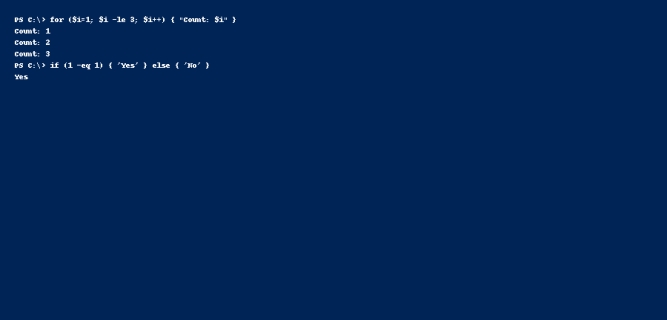
|  |
| --- |
|  |
|  | C:\Users\LENOVO\AppData\Local\Temp\ksohtml3152\wps27.jpg |

Objects, Arrays, Variables

|  |
| --- |
|  |
|  | C:\Users\LENOVO\AppData\Local\Temp\ksohtml3152\wps29.jpg |

More Operators

|  |
| --- |
|  |
|  | C:\Users\LENOVO\AppData\Local\Temp\ksohtml3152\wps31.jpg |

Modularization

|  |
| --- |
|  |
|  | C:\Users\LENOVO\AppData\Local\Temp\ksohtml3152\wps33.jpg |

