# LAB: PowerShell Core Review

In this lab, you’ll practice many of the techniques that you learned in the preceding Nuggets. “LAB A: PowerShell Core Review,” covers the solutions to these challenges. See how many you can complete before watching the Nugget.

1. PowerShell ships with two different hosting applications. Name them.

Console, ISE

2. In PowerShell v2, how can you display a list of locally installed modules that might not be loaded into memory?

Get-module -ListAvailable?

3. In PowerShell v3, why is it possible to search for help on commands that aren’t even loaded into memory, yet?

Module auto loading feature

4. If you were reading the help for a command, and saw a parameter listed as follows, what would you know about the parameter?

[[-ComputerName] <string[]>] It only takes strings

[[-Name] <string[]>] all brackets mean optional. Square around parameter means positional.

5. How can you discover the command that the alias **gsv** points to?

Help gsv (Get-service) or get-alias gsv

6. Fix the following command:

Get EventLog –Security –Newest 100

Get-EventLog -Logname Security -Newest 100

7. What command can be used to change a property value for an item? For example, how would you change the Read Only property of a file?

Set-itemproperty -path [C:\item.txt](../../../../../../../item.txt) -name property -value value

8. In PowerShell v2, can New-PSDrive be used to map a drive in Explorer? What about in PowerShell v3?

V2 won’t appear in explorer

new-psdrive -name test -root C:\windows\ -psprovider FileSystem

9. Create a simple hashtable that has two key/value pairs.

@{‘key1’=’valiue1’;’key2’=’value2’}

10. What is a here-string?

$herestring = @’

anything

‘@ # needs to be at beginning of line.

11. What is the difference between enclosing a string in single quotes or double quotes?

In doubles “$x” and in double quotes

$a = “x contains $(get-service | select -property name)”

12. Is **${variable name}** a valid variable name in PowerShell?

Yes. Though not good.

13. How would you permanently delete a variable from memory? List two possible techniques for doing so.

Close powershell

or cd Variable:

del variable:\x

get-command -noun variable

14. Write a command that can take a string like **first.last@company.pri** and change it to **last.first@company.pri.** Use a regular expression, and test it with strings like **don.jones@lab.pri** and [**greg.shields@remote.pri**](mailto:greg.shields@remote.pri).

$email = ‘[don.jones@lab.pri](mailto:don.jones@lab.pri)’

$regex = “([a-z]+)\.([a-z]+)@”

$email -match $regex

15. What’s the difference between Import-CSV and Get-Content when reading a comma-separated values file?

Get-Content = Raw Text Uninterpreted

Import-CSV = Interpreted

16. Can you use Export-CSV to export a data file whose fields are delimited with a pipe character instead of a comma?

Help export-csv

-delimiter ‘|’

17. How can you display a list of an object’s properties, methods, and events?

| gm (get-member)

18. Write a command that will display the current day of week (like Monday).

(get-date).dayofweek

19. Will the following command work to retrieve BIOS information from every computer in the domain? Why or why not?

Get-ADComputer –filter \* |  
Select @{n='computername';e={$\_.name}} |  
Get-WmiObject –class Win32\_BIOS

Get-WMIOBJECT Does not accept pipeline on computername.

20. Will the following command work to retrieve hotfix information from every computer in the domain? Why or why not? If the command will not work, rewrite it so that it will work.

Get-ADComputer –filter \* |  
Select @{n='computername';e={$\_.name}} |  
Get-Hotfix

21. Write a command that will display all running processes. The output should be a five-column table that includes each process name, ID, virtual memory, paged memory, and working set. Display all memory values in megabytes, with two decimal places.

Get-Process | ft -Property ID,Name,@{n='VM(GB)';e={$\_.VM / 1GB  };formatstring='N2'},@{n='PM(MB)';e={$\_.PM / 1MB  };formatstring='N2'},@{n='WS(MB)';e={$\_.WS / 1MB  };formatstring='N2'} -AutoSize

22. Read about the command **Get-History**. It displays a list of recently run commands. Write a command that displays a list of recent commands, including the ID, command text, and the amount of time it took the command to complete (display that information as ElapsedTime). Display the list with the longest-running commands first. This one is tricky – take some time to think about it.

get-history | Select-Object -property ID,CommandLine,@{n='ElapsedTime';e={$\_.endexecutiontime - $\_.startexecutiontime}} | Sort-Object -Property ElapsedTime `

-Descending | ft -AutoSize -Property Id,ElapsedTime,CommandLine -Wrap

23. Write at least four different commands that will terminate all running instances of Notepad.

Stop-process -name notepad

get-process -name notepad | stop-process

get-process -name notepad | foreach-object {$\_.kill()}

Now you’re ready to watch the lab review Nugget!