10961B PS Command File

Copy and paste code from this ISO into PowerShell with Ctrl+C and Ctrl+V.

Module 1 Labs A - B

Lab A Exercise 1

Start-Transcript C:\DayOne.txt

-----------------------------------------------------

Lab A Exercise 2

ise

-----------------------------------------------------

Lab B Exercise 1

help \*resolve\*

or:

Get-Command \*resolve\*

-----------------------------------------------------

help \*adapter\*

or:

Get-Command \*adapter\*

-----------------------------------------------------

help set-netadapter

-----------------------------------------------------

help \*sched\*

or:

Get-Command \*sched\*

-----------------------------------------------------

Get-Command –Verb Block

Or:

help \*block\*

-----------------------------------------------------

help \*branch\*

or:

help \*cache\*

or:

Get-Command \*cache\*

-----------------------------------------------------

help \*firewall\*

or:

Get-Command \*firewall\*

or:

help \*rule\*

or:

Get-Command \*rule\*

-----------------------------------------------------

help get-netfirewallrule –full

-----------------------------------------------------

help \*address\*

-----------------------------------------------------

help \*suspend\*

or:

Get-Command –verb suspend

-----------------------------------------------------

help \*format\*

or:

Get-Command –Verb format

-----------------------------------------------------

Get-NetFirewallRule -Enabled True

-----------------------------------------------------

Lab B Exercise 2

Get-NetFirewallRule -Enabled True

-----------------------------------------------------

Get-NetIPAddress –AddressFamily IPv4

-----------------------------------------------------

Set-Service –Name BITS –StartupType Automatic

-----------------------------------------------------

Test-Connection -ComputerName LON-DC1 -Quiet

-----------------------------------------------------

Get-EventLog –LogName Security –Newest 10

-----------------------------------------------------

Lab B Exercise 3

help \*comparison\*

then run:

help about\_comparison\_operators -ShowWindow

-----------------------------------------------------

$env:computername

-----------------------------------------------------

help \*signing\*

then run:

help about\_signing

-----------------------------------------------------

Module 2 Labs A - D

Lab A Exercise 1

help \*date\*

-----------------------------------------------------

Get-Date | Get-Member

-----------------------------------------------------

Get-Date | Select-Object -Property DayOfYear

-----------------------------------------------------

help \*hotfix\*

-----------------------------------------------------

Get-Hotfix | Get-Member

-----------------------------------------------------

Get-Hotfix | Select-Object -Property HotFixID,InstalledOn,InstalledBy

-----------------------------------------------------

help \*scope\*

-----------------------------------------------------

Help Get-DHCPServerv4Scope –ShowWindow

-----------------------------------------------------

Get-DHCPServerv4Scope -ComputerName LON-DC1

-----------------------------------------------------

Get-DHCPServerv4Scope -ComputerName LON-DC1 | Select-Object -Property ScopeId,SubnetMask,Name

-----------------------------------------------------

help \*rule\*

-----------------------------------------------------

Get-NetFirewallRule

-----------------------------------------------------

Help Get-NetFirewallRule -ShowWindow

-----------------------------------------------------

Get-NetFirewallRule -Enabled True

-----------------------------------------------------

Get-NetFirewallRule -Enabled True | Select-Object -Property DisplayName,Profile,Direction,Action | Sort-Object -Property DisplayName

-----------------------------------------------------

help \*neighbor\*

-----------------------------------------------------

help Get-NetNeighbor -ShowWindow

-----------------------------------------------------

Get-NetNeighbor

-----------------------------------------------------

Get-NetNeighbor | Sort-Object -Property State

-----------------------------------------------------

Get-NetNeighbor | Sort-Object -Property State | Select-Object -Property IPAddress,State

-----------------------------------------------------

ping LON-DC1

-----------------------------------------------------

ping LON-CL1

-----------------------------------------------------

help \*cache\*

-----------------------------------------------------

Get-DnsClientCache

-----------------------------------------------------

Get-DnsClientCache | Sort Name | Select Name,Type,TimeToLive

-----------------------------------------------------

Lab B Exercise 1

Get-Process

-----------------------------------------------------

Get-Process | Sort Name -Descending | Select Name,ID,VM,PM

-----------------------------------------------------

help ConvertTo-HTML –ShowWindow

-----------------------------------------------------

Get-Process | Sort Name -Descending | ConvertTo-HTML -Property Name,ID,VM,PM

-----------------------------------------------------

Get-Process | Sort Name -Descending | ConvertTo-HTML -Property Name,ID,VM,PM | Out-File ProcReport.html

-----------------------------------------------------

Invoke-Expression .\ProcReport.html

-----------------------------------------------------

Get-Process | Sort Name -Descending | ConvertTo-HTML -Property Name,ID,VM,PM -PreContent "Processes" -PostContent (Get-Date) | Out-File ProcReport.html

-----------------------------------------------------

Invoke-Expression .\ProcReport.html

-----------------------------------------------------

Lab B Exercise 2

Get-EventLog -Newest 10 -LogName System

-----------------------------------------------------

Get-EventLog -Newest 10 -LogName System | ConvertTo-CSV

-----------------------------------------------------

Get-EventLog -Newest 10 -LogName System | Export-Csv SysEvents.csv

-----------------------------------------------------

Notepad SysEvents.csv

-----------------------------------------------------

Get-EventLog -Newest 10 -LogName System | Export-Csv SysEvents.csv -NoTypeInformation

-----------------------------------------------------

Notepad SysEvents.csv

-----------------------------------------------------

Get-Service | Sort Status -Descending

-----------------------------------------------------

Get-Service | Sort Status -Descending | Export-CliXML Services.xml

-----------------------------------------------------

Notepad Services.xml

-----------------------------------------------------

Get-Service | Sort Status -Descending | Select Name,DisplayName,Status | Export-CliXML Services.xml

-----------------------------------------------------

Notepad Services.xml

-----------------------------------------------------

Help ConvertTo-CSV -ShowWindow

-----------------------------------------------------

Get-EventLog -newest 20 -LogName Security

-----------------------------------------------------

Get-EventLog -newest 20 -LogName Security | Select EventID,TimeWritten,Message

-----------------------------------------------------

Get-EventLog -newest 20 -LogName Security | Select EventID,TimeWritten,Message | Export-CSV Security.pdd -Delimiter '|'

-----------------------------------------------------

Notepad Security.pdd

-----------------------------------------------------

Import-Csv Security.pdd -Delimiter '|' | Select -First 10

-----------------------------------------------------

Lab C Exercise 1

help \*user\*

-----------------------------------------------------

help Get-ADUser -ShowWindow

-----------------------------------------------------

Get-ADUser -Filter \*

-----------------------------------------------------

Get-ADUser -Filter \* -SearchBase "cn=Users,dc=Adatum,dc=com"

-----------------------------------------------------

Get-EventLog -LogName Security | Where EventID -eq 4624

-----------------------------------------------------

Get-EventLog -LogName Security | Where EventID -eq 4624 | Select TimeWritten,EventID,Message

-----------------------------------------------------

Get-EventLog -LogName Security | Where EventID -eq 4624 | Select TimeWritten,EventID,Message | ConvertTo-HTML | Out-File EventReport.html

-----------------------------------------------------

Invoke-Expression .\EventReport.html

-----------------------------------------------------

Get-ChildItem -Path CERT: -Recurse

-----------------------------------------------------

Get-ChildItem -Path CERT: -Recurse | Get-Member

-----------------------------------------------------

Get-ChildItem -Path CERT: -Recurse | Where HasPrivateKey -eq $False

-----------------------------------------------------

Get-ChildItem -Path CERT: -Recurse | Where { $PSItem.HasPrivateKey -eq $False -and $PSItem.NotAfter -gt (Get-Date) -and $PSItem.NotBefore -lt (Get-Date) }

-----------------------------------------------------

Get-ChildItem -Path CERT: -Recurse | Where { $PSItem.HasPrivateKey -eq $False -and $PSItem.NotAfter -gt (Get-Date) -and $PSItem.NotBefore -lt (Get-Date) } | Select Issuer,NotBefore,NotAfter

-----------------------------------------------------

Get-Volume

-----------------------------------------------------

Get-Volume | Get-Member

-----------------------------------------------------

Get-Volume | Where-Object { $PSItem.SizeRemaining -gt 0 }

-----------------------------------------------------

Get-Volume | Where-Object { $PSItem.SizeRemaining -gt 0 -and $PSItem.SizeRemaining / $PSItem.Size -lt .99 }

-----------------------------------------------------

Get-Volume | Where-Object { $PSItem.SizeRemaining -gt 0 -and $PSItem.SizeRemaining / $PSItem.Size -lt .1 }

-----------------------------------------------------

help \*control\*.

-----------------------------------------------------

Get-ControlPanelItem

-----------------------------------------------------

Get-ControlPanelItem -Category 'System and Security'

-----------------------------------------------------

Lab D Exercise 1

Get-ChildItem -Path CERT: -Recurse

-----------------------------------------------------

Get-ChildItem -Path CERT: -Recurse | Get-Member

-----------------------------------------------------

Get-ChildItem -Path CERT: -Recurse | ForEach GetKeyAlgorithm

-----------------------------------------------------

Get-ChildItem -Path CERT: -Recurse | Select Issuer,@{n='KeyAlgorithm';e={$PSItem.GetKeyAlgorithm()}}

-----------------------------------------------------

help \*random\*

-----------------------------------------------------

help Get-Random -ShowWindow

-----------------------------------------------------

1..100

-----------------------------------------------------

1..100 | ForEach { Get-Random -SetSeed $PSItem }

-----------------------------------------------------

Get-WmiObject -Class Win32\_OperatingSystem -EnableAllPrivileges

-----------------------------------------------------

Get-WmiObject -Class Win32\_OperatingSystem -EnableAllPrivileges | Get-Member

-----------------------------------------------------

Get-WmiObject -Class Win32\_OperatingSystem -EnableAllPrivileges | ForEach Reboot

-----------------------------------------------------

Module 3 Lab 1

Exercise 1

Get-ADComputer -Filter \* | Get-Service –Name

-----------------------------------------------------

Get-ADComputer -Filter \* | Select @{n='ComputerName';e={$PSItem.Name}} | Get-Service -Name \*

-----------------------------------------------------

Get-ADComputer -Filter \* | Select @{n='ComputerName';e={$PSItem.Name}} | Get-WmiObject -Class Win32\_BIOS

-----------------------------------------------------

Get-Content Names.txt | Get-Service

-----------------------------------------------------

Get-Service -ComputerName (Get-Content Names.txt)

-----------------------------------------------------

Get-Service -ComputerName (Get-ADComputer -Filter \*)

-----------------------------------------------------

Get-EventLog -LogName Security -ComputerName (Get-ADComputer -Filter \* | Select -Expand Name)

-----------------------------------------------------

Get-EventLog -LogName System -Newest 50 -ComputerName (Get-ADComputer -filter \* | Select-Object -ExpandProperty Name)

-----------------------------------------------------

Get-Content Names.txt | Restart-Computer

-----------------------------------------------------

Test-Connection -ComputerName (Get-Content Names.txt)

-----------------------------------------------------

Get-ADComputer -filter \* | Select-Object @{n='ComputerName';e={$PSItem.Name}} | Set-Service -Name WinRM -StartupType Auto

Module 4 Lab 1

Exercise 1

Help New-Item -ShowWindow

-----------------------------------------------------

New-Item -Path c:\ -Name ScriptOutput -ItemType Directory

-----------------------------------------------------

Exercise 2

Help New-PSDrive -ShowWindow

-----------------------------------------------------

New-PSDrive -Name Output -Root C:\ScriptOutput -PSProvider FileSystem

-----------------------------------------------------

Exercise 3

Help New-Item -ShowWindow

-----------------------------------------------------

New-Item -Path HKCU:\Software -Name Scripts

-----------------------------------------------------

Exercise 4

Help New-ItemProperty -ShowWindow

-----------------------------------------------------

New-ItemProperty -Path HKLM:\Software\Microsoft\Windows\CurrentVersion\Run -Name "Windows PowerShell" -Value "C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe"

-----------------------------------------------------

Exercise 5

Help Set-Item -ShowWindow

-----------------------------------------------------

set-item -Path WSMan:\localhost\Service\MaxConnections -Value 250

-----------------------------------------------------

Module 5 Lab 1

Exercise 1

Get-CimInstance -Class Win32\_ComputerSystem

-----------------------------------------------------

Get-CimInstance -Class Win32\_ComputerSystem | Format-List -Property PSComputerName,Description,Domain,Manufacturer,Model,NumberOfProcessors,@{n='TotalPhysicalMemory';e={$PSItem.TotalPhysicalMemory / 1GB}}

-----------------------------------------------------

Get-Process | Format-Table -Property Name,ID,@{n='VM(MB)';e={$PSItem.VM / 1MB};formatString='N2'},@{n='PM(MB)';e={$PSItem.PM / 1MB};formatString='N2'} -AutoSize | Out-File Procs.txt

-----------------------------------------------------

Get-Process | Sort BasePriority | Format-Table -GroupBy BasePriority

-----------------------------------------------------

Get-NetRoute | Format-Table -Property AddressFamily,RouteMetric,TypeOfRoute, @{n='DestinationPrefix';e={$PSItem.DestinationPrefix};align='right'} -AutoSize

-----------------------------------------------------

Module 6 Lab 1

Exercise 1

Get-WmiObject -namespace root\cimv2 -list | Where Name -like '\*configuration\*' | Sort Name

-----------------------------------------------------

Get-WmiObject -Class Win32\_NetworkAdapterConfiguration | Where DHCPEnabled -eq $True | Select IPAddress

-----------------------------------------------------

Get-WmiObject -namespace root\cimv2 -list | Where Name -like '\*operating\*' | Sort Name

-----------------------------------------------------

Get-WmiObject -Class Win32\_OperatingSystem | Get-Member

-----------------------------------------------------

Get-WmiObject -Class Win32\_OperatingSystem | Select Version,ServicePackMajorVersion,BuildNumber

-----------------------------------------------------

Get-WmiObject -namespace root\cimv2 -list | Where Name -like '\*system\*' | Sort Name

-----------------------------------------------------

Get-WmiObject -class Win32\_ComputerSystem | Format-List -Property \*

-----------------------------------------------------

Get-WmiObject -class Win32\_ComputerSystem | Select Manufacturer,Model,@{n='RAM';e={$PSItem.TotalPhysicalMemory}}

-----------------------------------------------------

Get-WmiObject -namespace root\cimv2 -list | Where Name -like '\*service\*' | Sort Name

-----------------------------------------------------

Get-WmiObject -Class Win32\_Service | FL \*

-----------------------------------------------------

Get-WmiObject -Class Win32\_Service -Filter "Name LIKE 'S%'" | Select Name,State,StartName

-----------------------------------------------------

Exercise 2

Get-WmiObject -namespace root\cimv2 -list | Where Name -like '\*user\*' | Sort Name

-----------------------------------------------------

Get-CimInstance -Class Win32\_UserAccount | Get-Member

-----------------------------------------------------

Get-CimInstance -Class Win32\_UserAccount | Format-Table -Property Caption,Domain,SID,FullName,Name

-----------------------------------------------------

Get-WmiObject -namespace root\cimv2 -list | Where Name -like '\*bios\*' | Sort Name

-----------------------------------------------------

Get-CimInstance -Class Win32\_BIOS

-----------------------------------------------------

Get-CimInstance -Classname Win32\_NetworkAdapterConfiguration

-----------------------------------------------------

Get-CimInstance -Classname Win32\_NetworkAdapterConfiguration -ComputerName LON-DC1

-----------------------------------------------------

Get-WmiObject -namespace root\cimv2 -list | Where Name -like '\*group\*' | Sort Name

-----------------------------------------------------

Get-CimInstance -ClassName Win32\_Group -ComputerName LON-DC1

-----------------------------------------------------

Exercise 3

Invoke-CimMethod -ClassName Win32\_OperatingSystem -ComputerName LON-DC1 -MethodName Reboot

-----------------------------------------------------

Get-WmiObject -Class Win32\_Service -Filter "Name='WinRM'" | Invoke-WmiMethod -Name ChangeStartMode -Argument 'Automatic'

-----------------------------------------------------

Module 7 Lab 1

Exercise 1

Get-ExecutionPolicy

Set-ExecutionPolicy Restricted

-----------------------------------------------------

"Get-Service" | Out-File C:\Test.ps1

-----------------------------------------------------

C:\Test.ps1

-----------------------------------------------------

Set-ExecutionPolicy RemoteSigned

-----------------------------------------------------

E:\Mod07\demoCode\Test-CodesignedFile.ps1

-----------------------------------------------------

Module 8 Labs A - D

Lab A Exercise 1

ise

-----------------------------------------------------

Set-ExecutionPolicy RemoteSigned

-----------------------------------------------------

Lab A Exercise 2

[CmdletBinding()]

Param(

[Parameter(Mandatory=$True)]

[string]$ComputerName,

[int]$DriveType = 3

)

-----------------------------------------------------

$DriveType

-----------------------------------------------------

$ComputerName

-----------------------------------------------------

LON-CL1

-----------------------------------------------------

Lab A Exercise 3

Write-Verbose "Getting drive types of $DriveType from $ComputerName"

-----------------------------------------------------

Cd \

-----------------------------------------------------

.\Get-DiskInfo –Comp localhost

-----------------------------------------------------

.\Get-DiskInfo -Comp localhost -Verbose

-----------------------------------------------------

Lab A Exercise 4

<#

.SYNOPSIS

Retrieves disk space information.

.DESCRIPTION

Retrieves disk information from a single computer.

.PARAMETER ComputerName

The name of the computer to query.

.PARAMETER DriveType

The type of drive to query. Defaults to 3, representing local fixed disks.

.EXAMPLE

.\Get-DiskInfo -ComputerName localhost -Verbose

#>

-----------------------------------------------------

Cd \

-----------------------------------------------------

Help .\Get-DiskInfo -ShowWindow

-----------------------------------------------------

Lab B Exercise 1

Function Get-DiskInfo {

-----------------------------------------------------

}

-----------------------------------------------------

Get-DiskInfo -Comp localhost

-----------------------------------------------------

Lab B Exercise 2

Get-DiskInfo -comp localhost

-----------------------------------------------------

Remove-Module MyTools

-----------------------------------------------------

Lab B Exercise 3

Get-DiskInfo -comp localhost

-----------------------------------------------------

Get-DiskInfo -Comp localhost -Debug

-----------------------------------------------------

Import-Module MyTools

-----------------------------------------------------

$ComputerName

-----------------------------------------------------

Exit

-----------------------------------------------------

Remove-Module MyTools

-----------------------------------------------------

Lab C Exercise 1

Try {

-----------------------------------------------------

} Catch {

Write-Verbose "Error querying $ComputerName"

}

-----------------------------------------------------

Get-WMIObject -Class Win32\_LogicalDisk -Filter "DriveType=$DriveType" -ComputerName $ComputerName -ErrorAction Stop |

-----------------------------------------------------

Remove-Module MyTools

-----------------------------------------------------

Get-DiskInfo -Comp BAD -Verbose

-----------------------------------------------------

Remove-Module MyTools

-----------------------------------------------------

Lab C Exercise 2

Try {

-----------------------------------------------------

} Catch {

Write-Verbose "Failed to connect to $ComputerName"

}

-----------------------------------------------------

Get-EventLog -LogName Security -Newest $MaxEntries -ErrorAction Stop -ComputerName $ComputerName |

-----------------------------------------------------

Lab D Exercise 2

function Get-OSInfo {

}

-----------------------------------------------------

[CmdletBinding()]

param(

[Parameter(Mandatory=$True)]

[string]$ComputerName

)

-----------------------------------------------------

Get-CimInstance -ClassName Win32\_OperatingSystem -ComputerName $ComputerName |

-----------------------------------------------------

Remove-Module MyTools

-----------------------------------------------------

Get-OSInfo -comp localhost

-----------------------------------------------------

WinRM qc

-----------------------------------------------------

WinRM qc

-----------------------------------------------------

Remove-Module MyTools

-----------------------------------------------------

Lab D Exercise 3

[string[]]$ComputerName

-----------------------------------------------------

ForEach ($name in $ComputerName) {

-----------------------------------------------------

}

-----------------------------------------------------

Get-CimInstance -ClassName Win32\_OperatingSystem -ComputerName $name |

-----------------------------------------------------

Get-OSInfo -comp localhost,LON-DC1

-----------------------------------------------------

Remove-Module MyTools

-----------------------------------------------------

Lab D Exercise 4

Try {

-----------------------------------------------------

} Catch {

Write "Error connecting to $name"

}

-----------------------------------------------------

Get-CimInstance -ClassName Win32\_OperatingSystem -ErrorAction Stop -ComputerName $name |

-----------------------------------------------------

Get-OSInfo –comp localhost,BAD,LON-CL1

-----------------------------------------------------

Remove-Module MyTools

-----------------------------------------------------

Module 9 Labs A - B

Lab A Exercise 1

set-executionPolicy RemoteSigned

-----------------------------------------------------

Enable-PSremoting

-----------------------------------------------------

help \*sessionconfiguration\*

-----------------------------------------------------

Get-PSSessionConfiguration

-----------------------------------------------------

Lab A Exercise 2

Enter-PSSession -ComputerName LON-DC1

-----------------------------------------------------

Install-WindowsFeature NLB

-----------------------------------------------------

Exit-PSSession

-----------------------------------------------------

Enter-PSSession -ComputerName LON-DC1

-----------------------------------------------------

Enter-PSSession -ComputerName LON-CL1

-----------------------------------------------------

Exit-PSSession

-----------------------------------------------------

Enter-PSSession -ComputerName localhost

-----------------------------------------------------

Notepad

-----------------------------------------------------

Exit-PSSession

-----------------------------------------------------

Lab A Exercise 3

help \*adapter\*

-----------------------------------------------------

help Get-NetAdapter

-----------------------------------------------------

Invoke-Command -ComputerName LON-CL1,LON-DC1 -ScriptBlock { Get-NetAdapter -Physical }

-----------------------------------------------------

Get-Process | Get-Member

-----------------------------------------------------

Invoke-Command -ComputerName LON-DC1 -ScriptBlock { Get-Process } | Get-Member

-----------------------------------------------------

Lab B Exercise 1

$dc = New-PSSession -ComputerName LON-DC1

-----------------------------------------------------

$dc

-----------------------------------------------------

Get-Module -ListAvailable -PSSession $dc

-----------------------------------------------------

Get-Module -ListAvailable -PSSession $dc | Where { $\_.Name -Like '\*share\*' }

-----------------------------------------------------

Import-Module -PSSession $dc -Name SMBShare -Prefix DC

-----------------------------------------------------

Get-DCSMBShare

-----------------------------------------------------

Get-SMBShare

-----------------------------------------------------

Get-PSSession | Remove-PSSession

-----------------------------------------------------

Get-PSSession

-----------------------------------------------------

Lab B Exercise 2

$computers = New-PSSession -ComputerName LON-CL1,LON-DC1

-----------------------------------------------------

$computers

-----------------------------------------------------

Get-Module -ListAvailable | Where { $\_.Name -like ‘\*security\*’ }

-----------------------------------------------------

Invoke-Command -Session $computers -ScriptBlock { Import-Module NetSecurity }

-----------------------------------------------------

Get-Command -Module NetSecurity

-----------------------------------------------------

Help Get-NetFirewallRule -ShowWindow

-----------------------------------------------------

Invoke-Command -Session $computers -ScriptBlock { Get-NetFirewallRule -Enabled True } | Select Name,PSComputerName

-----------------------------------------------------

Invoke-Command -Session $computers -ScriptBlock { Remove-Module NetSecurity }

-----------------------------------------------------

Get-WmiObject -Class Win32\_LogicalDisk -Filter "DriveType=3"

-----------------------------------------------------

Invoke-Command -Session $computers -ScriptBlock { Get-WmiObject -Class Win32\_LogicalDisk -Filter "DriveType=3" }

-----------------------------------------------------

Invoke-Command -Session $computers -ScriptBlock { Get-WmiObject -Class Win32\_LogicalDisk -Filter "DriveType=3" } | ConvertTo-HTML -Property PSComputerName,DeviceID,FreeSpace,Size

-----------------------------------------------------

Get-PSSession | Remove-PSSession

-----------------------------------------------------

Module 10 Lab 1

Exercise 1

Set-ExecutionPolicy RemoteSigned

-----------------------------------------------------

[CmdletBinding()]

Param(

)

-----------------------------------------------------

[Parameter(Mandatory=$True)]

[string]$MACAddress,

-----------------------------------------------------

$LocalCredential = (Get-Credential -Message "Provide credential for target machine"),

-----------------------------------------------------

$DomainCredential = (Get-Credential -Message "Provide domain credential to add machine to domain"),

-----------------------------------------------------

[Parameter(Mandatory=$True)]

[string]$NewComputerName,

-----------------------------------------------------

[Parameter(Mandatory=$True)]

[string]$NewIPAddress,

-----------------------------------------------------

[Parameter(Mandatory=$True)]

[string]$Role,

-----------------------------------------------------

[string]$Domain = "ADATUM",

-----------------------------------------------------

[Parameter(Mandatory=$True)]

[string]$ScopeID,

-----------------------------------------------------

[Parameter(Mandatory=$True)]

[string]$DHCPServerName

-----------------------------------------------------

Exercise 2

ipconfig /all

-----------------------------------------------------

Logoff

-----------------------------------------------------

$MACAddress = "01-23-45-01-23-45". Replace 01-23-45-01-23-45

-----------------------------------------------------

Help \*lease\*

-----------------------------------------------------

help \*scope\*

-----------------------------------------------------

-----------------------------------------------------

$DHCPServerName = "LON-DC1"

-----------------------------------------------------

Get-DhcpServerv4Scope -ComputerName $DHCPServerName

-----------------------------------------------------

$ScopeID = "10.0.0.0"

-----------------------------------------------------

Get-DHCPServerv4Lease -ScopeId $ScopeID -ComputerName $DHCPServerName

-----------------------------------------------------

$OldIPAddress = Get-DHCPServerv4Lease -ScopeId $ScopeID -ComputerName $DHCPServerName | Where-Object { $PSItem.ClientID -eq $MACAddress } | Select-Object -ExpandProperty IPAddress | Select-Object -ExpandProperty IPAddressToString

-----------------------------------------------------

$OldIPAddress = "$OldIPAddress"

-----------------------------------------------------

$NewComputerName = "LON-SVR2"

-----------------------------------------------------

[CmdletBinding()]

Param(

[Parameter(Mandatory=$True)]

[string]$MACAddress,

$LocalCredential = (Get-Credential -Message "Provide credential for target machine"),

$DomainCredential = (Get-Credential -Message "Provide domain credential to add machine to domain"),

[Parameter(Mandatory=$True)]

[string]$NewComputerName,

[Parameter(Mandatory=$True)]

[string]$NewIPAddress,

[Parameter(Mandatory=$True)]

[string]$Role,

[string]$Domain = "ADATUM",

[Parameter(Mandatory=$True)]

[string]$ScopeID,

[Parameter(Mandatory=$True)]

[string]$DHCPServerName

)

**$OldIPAddress = Get-DhcpServerv4Lease -ScopeId $ScopeID -ComputerName $DHCPServerName |**

**Where-Object { $PSItem.ClientId -eq $MACAddress } |**

**Select-Object -ExpandProperty IPAddress |**

**Select-Object -ExpandProperty IPAddressToString**

**$OldIPAddress = "$OldIPAddress"**

-----------------------------------------------------

Exercise 3

help \*reservation\*

-----------------------------------------------------

$NewIPAddress = "10.0.0.10"

-----------------------------------------------------

Add-DhcpServerv4Reservation -ClientId $MACAddress -IPAddress $NewIPAddress -ScopeId $ScopeID -ComputerName $DHCPServerName

-----------------------------------------------------

[CmdletBinding()]

Param(

[Parameter(Mandatory=$True)]

[string]$MACAddress,

$LocalCredential = (Get-Credential -Message "Provide credential for target machine"),

$DomainCredential = (Get-Credential -Message "Provide domain credential to add machine to domain"),

[Parameter(Mandatory=$True)]

[string]$NewComputerName,

[Parameter(Mandatory=$True)]

[string]$NewIPAddress,

[Parameter(Mandatory=$True)]

[string]$Role,

[string]$Domain = "ADATUM",

[Parameter(Mandatory=$True)]

[string]$ScopeID,

[Parameter(Mandatory=$True)]

[string]$DHCPServerName

)

$OldIPAddress = Get-DhcpServerv4Lease -ScopeId $ScopeID -ComputerName $DHCPServerName |

Where-Object { $PSItem.ClientId -eq $MACAddress } |

Select-Object -ExpandProperty IPAddress |

Select-Object -ExpandProperty IPAddressToString

$OldIPAddress = "$OldIPAddress"

**# Add a reservation**

**Add-DhcpServerv4Reservation -ClientId $MACAddress `**

**-IPAddress $NewIPAddress -ScopeId $ScopeID `**

**-ComputerName $DHCPServerName**

-----------------------------------------------------

Exercise 4

$OriginalTrustedHosts = Get-Item WSMan:\localhost\Client\TrustedHosts | Select-Object -ExpandProperty Value

-----------------------------------------------------

$OriginalTrustedHosts

-----------------------------------------------------

Dir WsMan:\localhost\Client

-----------------------------------------------------

Set-Item WSMan:\localhost\Client\TrustedHosts -Value "$OldIPAddress"

-----------------------------------------------------

Dir WSMan:\localhost\Client

-----------------------------------------------------

Set-Item WSMan:\localhost\Client\TrustedHosts -Value "$OriginalTrustedHosts"

-----------------------------------------------------

Dir WSMan:\localhost\Client

-----------------------------------------------------

Set-Item WSMan:\localhost\Client\TrustedHosts -Value $OldIPAddress

-----------------------------------------------------

Dir WSMan:\localhost\Client

-----------------------------------------------------

[CmdletBinding()]

Param(

...

$OldIPAddress = Get-DhcpServerv4Lease -ScopeId $ScopeID –ComputerName $DHCPServerName |

Where-Object { $PSItem.ClientId -eq $MACAddress } |

Select-Object -ExpandProperty IPAddress |

Select-Object -ExpandProperty IPAddressToString

$OldIPAddress = "$OldIPAddress"

# Add a reservation

Add-DhcpServerv4Reservation -ClientId $MACAddress `

-IPAddress $NewIPAddress -ScopeId $ScopeID `

-ComputerName $DHCPServerName

**# Save TrustedHosts**

**$OriginalTrustedHosts = Get-Item WSMan:\localhost\Client\TrustedHosts | select -ExpandProperty value**

**# Set TrustedHosts**

**Set-Item WSMan:\localhost\Client\TrustedHosts -Value $OldIPAddress**

**# Restore TrustedHosts**

**Set-Item WSMan:\localhost\Client\TrustedHosts -Value "$OriginalTrustedHosts"**

-----------------------------------------------------

Exercise 5

help \*feature\*

-----------------------------------------------------

$LocalCredential = Get-Credential

-----------------------------------------------------

Invoke-Command -ComputerName $OldIPAddress -Credential $LocalCredential -ScriptBlock { Install-WindowsFeature Telnet-Client }

-----------------------------------------------------

Get-WindowsFeature

-----------------------------------------------------

[CmdletBinding()]

Param(

...

# Add a reservation

Add-DhcpServerv4Reservation -ClientId $MACAddress `

-IPAddress $NewIPAddress -ScopeId $ScopeID `

-ComputerName $DHCPServerName

# Save TrustedHosts

$OriginalTrustedHosts = Get-Item WSMan:\localhost\Client\TrustedHosts | select -ExpandProperty value

# Set TrustedHosts

Set-Item WSMan:\localhost\Client\TrustedHosts -Value $OldIPAddress

**# Install role**

**Invoke-Command -ComputerName $OldIPAddress `**

**-Credential $LocalCredential `**

**-ScriptBlock { Install-WindowsFeature Telnet-Client }**

# Restore TrustedHosts

Set-Item WSMan:\localhost\Client\TrustedHosts -Value "$OriginalTrustedHosts"

-----------------------------------------------------

Exercise 6

help \*computer\*

-----------------------------------------------------

$DomainCredential = Get-Credential

-----------------------------------------------------

Invoke-Command -ComputerName $OldIPAddress -Credential $LocalCredential -ScriptBlock { param($x,$y) Add-Computer -DomainName ADATUM -NewName $x -Credential $y -Restart } -ArgumentList $NewComputerName,$DomainCredential

-----------------------------------------------------

CmdletBinding()]

...

# Install role

Invoke-Command -ComputerName $OldIPAddress `

-Credential $LocalCredential `

-ScriptBlock { Install-WindowsFeature Telnet-Client }

**# Add to domain and rename**

**Invoke-Command -ComputerName $OldIPAddress `**

**-Credential $LocalCredential `**

**-ScriptBlock { param($x,$y) Add-Computer -DomainName ADATUM `**

**-NewName $x `**

**-Credential $y `**

**-Restart } `**

**-ArgumentList $NewComputerName,$DomainCredential**

# Restore TrustedHosts

Set-Item WSMan:\localhost\Client\TrustedHosts -Value "$OriginalTrustedHosts"

-----------------------------------------------------

Set-Item WSMan:\localhost\Client\TrustedHosts -Value "OriginalTrustedHosts"

-----------------------------------------------------

Exercise 7

ipconfig /all

-----------------------------------------------------

Logoff

-----------------------------------------------------

Set-ExecutionPolicy RemoteSigned

-----------------------------------------------------

C:\Scripts\Set-ServerCoreInstance -NewComputerName LON-SVR2 -NewIPAddress 10.0.0.10 -Role Telnet-Client -MACAddress *00-15-5D-24-3D-14* -ScopeID *10.0.0.0* -DHCPServerName LON-DC1

-----------------------------------------------------

Get-ADComputer –filter \*

-----------------------------------------------------

Get-WindowsFeature -ComputerName LON-SVR2

-----------------------------------------------------

Dir WSMan:\localhost\Client

-----------------------------------------------------

Module 11 Labs A - B

Lab A Exercise 1

Enable-PSRemoting

-----------------------------------------------------

Invoke-Command -ScriptBlock { Get-NetAdapter -Physical } -ComputerName LON-DC1,LON-CL1 -AsJob -JobName RemoteNetAdapt

-----------------------------------------------------

Invoke-Command -ScriptBlock { Get-SMBShare } -ComputerName LON-DC1,LON-CL1 -AsJob -JobName RemoteShares

-----------------------------------------------------

Invoke-Command -ScriptBlock { Get-CimInstance -ClassName Win32\_Volume } -ComputerName (Get-ADComputer -Filter \* | Select -Expand Name) -AsJob -JobName RemoteDisks

-----------------------------------------------------

Start-Job -ScriptBlock { Get-EventLog -LogName Security } -Name LocalSecurity

-----------------------------------------------------

Start-Job -ScriptBlock { 1..100 | ForEach-Object { Dir C:\ -Recurse } } -Name LocalDir

-----------------------------------------------------

Lab A Exercise 2

Get-Job

-----------------------------------------------------

Get-Job -Name Remote\*

-----------------------------------------------------

Stop-Job -Name LocalDir

-----------------------------------------------------

Get-Job

-----------------------------------------------------

Receive-Job -Name RemoteNetAdapt

-----------------------------------------------------

Get-Job -Name RemoteDisks | Select -Expand ChildJobs | Where Location -eq 'LON-DC1' | Receive-Job

-----------------------------------------------------

Lab B Exercise 1

$option = New-ScheduledJobOption -WakeToRun -RunElevated

-----------------------------------------------------

$trigger1 = New-JobTrigger -Once -At (Get-Date).AddMinutes(10)

-----------------------------------------------------

$trigger2 = New-JobTrigger -AtLogOn

-----------------------------------------------------

Register-ScheduledJob -ScheduledJobOption $option -Trigger $trigger1 -ScriptBlock { Get-EventLog -LogName Security } -MaxResultCount 5 -Name LocalSecurityLog

-----------------------------------------------------

Register-ScheduledJob -ScheduledJobOption $option -Trigger $trigger2 -ScriptBlock { Get-Process } -Name ProcList

-----------------------------------------------------

Get-ScheduledJob -Name LocalSecurityLog | Select -Expand JobTriggers

-----------------------------------------------------

Logoff

-----------------------------------------------------

Import-Module PSScheduledJob

Get-Job

-----------------------------------------------------

Receive-Job -Name LocalSecurityLog

-----------------------------------------------------

Receive-Job -Name ProcList

-----------------------------------------------------

Module 12 Lab 1

Exercise 1

Set-ExecutionPolicy RemoteSigned

-----------------------------------------------------

. E:\Mod12\Labfiles\Lab12.ps1

-----------------------------------------------------

$ServiceNames -contains "WinRM"

-----------------------------------------------------

"Spooler" -in $ServiceNames

-----------------------------------------------------

$ComputerNames[1]

-----------------------------------------------------

$ComputerNames -Join ","

-----------------------------------------------------

$ComputerNames -Join "`t"

-----------------------------------------------------

$phrase -replace 'dog','gelding'

-----------------------------------------------------

($List -Split ',')[1]

-----------------------------------------------------

$phrase.ToUpper()

-----------------------------------------------------

$phrase.ToLower()

-----------------------------------------------------

$phrase.Replace('over','around').ToUpper()

-----------------------------------------------------

$List.Substring(5,3)

-----------------------------------------------------

$padded.Trim()

-----------------------------------------------------

$phrase.Length

-----------------------------------------------------

$phrase.StartsWith('The')

-----------------------------------------------------

$unpadded.PadLeft(10)

-----------------------------------------------------

$today = Get-Date

-----------------------------------------------------

$today.AddDays(-10)

-----------------------------------------------------

$today.AddDays(30)

-----------------------------------------------------

$today.Hour

-----------------------------------------------------

$today.Month

-----------------------------------------------------

$today.ToShortDateString()

-----------------------------------------------------

$today.ToUniversalTime()

-----------------------------------------------------

$today.ToLongTimeString()

-----------------------------------------------------

Exercise 2

$admin = Get-Credential -Credential ADATUM\Administrator

-----------------------------------------------------

Invoke-Command -ComputerName LON-DC1 -Credential $admin -ScriptBlock { Get-ADUser -Filter \* | Export-CSV C:\Users.csv }

-----------------------------------------------------

cd\

-----------------------------------------------------

dir

-----------------------------------------------------

Notepad

-----------------------------------------------------

Exercise 3

ISE

-----------------------------------------------------

Cd c:\

$cred = Get-Credential -Credential ADATUM\Administrator

-----------------------------------------------------

$PSDefaultParameterValues=@{"Get-EventLog:LogName"="Security";"Get-EventLog:Newest"=10}

-----------------------------------------------------

$cred

-----------------------------------------------------

Get-EventLog