**Exercise 1: Filtering objects**  
**Scenario**In this exercise, you will use filtering to produce lists of management information that include only  
specified data and elements for the reports you must produce.  
  
The main tasks for this exercise are as follows:  
1. Display a list of all the users in the Users container of Active Directory.  
2. Create a report showing the Security event log entries that have the event ID 4624.  
3. Display a list of the encryption certificates installed on the computer.  
4. Create a report that shows the disk volumes that are running low on space.  
5. Create a report that displays specified Control Panel items

**Task 1: Display a list of all the users in the Users container of Active Directory**  
  
1. On **LON-CL1**, open **Windows PowerShell** as an administrator.  
2. Using a keyword such as **user,** find a command that can list Active Directory users.  
3. View the help for the command, and identify any mandatory parameters.  
4. Display a list of all the users in Active Directory in a format that lets you easily compare properties.  
5. Display the same list of all the users in the same format. This time, however, display only those users  
in the Users container of Active Directory. Use a search base of **"cn=Users,dc=adatum,dc=com"** for  
this task.

**Answers:**

help \*user\*

Get-Help Get-ADUser –ShowWindow

Get-ADUser –Filter \* | ft

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

**Task 2: Create a report showing the Security event log entries that have the event  
ID 4624**  
  
1. Display only the total number of **Security** event log entries that have the event ID 4624.  
2. Display the full list of the **Security** event log entries that have the event ID 4624, and show only the  
time written, event ID, and message.  
3. Display only the 10 oldest entries in a format that lets you view the message details.

**Answers:**

Get-EventLog -LogName Security | Where EventID -eq 4624 | Measure-Object | fw

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

Get-EventLog -LogName Security | Where EventID -eq 4624 | Select TimeWritten,EventID,Message -Last 10 | fl

**Task 3: Display a list of the encryption certificates installed on the computer**1. Display a directory listing of all the items on the **CERT** drive. Include subfolders in the list.  
2. Display the list, again and show the name and issuer for only the certificates that do not have a  
private key. Show the results in one column.  
3. Display the list, again and show only the current certificates. Those certificates have a **NotBefore** date  
that is before today and a **NotAfter** date that is after today. Include the **NotBefore** and **NotAfter**properties in the results, and display the results in a format that allows you to easily compare dates.  
Also, make sure that no data is truncated.  
  
**Answers:**

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

Get-ChildItem -Path CERT: -Recurse | Where HasPrivateKey -eq $False | Select-Object -Property FriendlyName,Issuer | fl

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

**Task 4: Create a report that shows the disk volumes that are running low on space**1. Display a list of the disk volumes.  
2. Display a list in one column of the volumes that have more than zero bytes of free space.  
3. Display a list of the volumes that have less than 99 percent free space and more than zero bytes of  
free space. Show only the drive letter and disk size, in MB.  
4. Display a list of the volumes that have less than 10 percent free space and more than zero bytes of  
free space. This command might produce no results if no volumes on your computer meet the  
criteria.

**Answers:**

Get-Volume

Get-Volume | Get-Member

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

Get-Volume | Where-Object { $PSItem.SizeRemaining -gt 0 -and $PSItem.SizeRemaining /$PSItem.Size -lt .99 }| Select-Object DriveLetter, @{n='Size';e={'{0:N2}' -f ($PSItem.Size/1MB)}}

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

**Task 5: Create a report that displays specified Control Panel items**  
1. Display a list of all the Control Panel items.  
2. Display the names and descriptions, sorted by name, of the Control Panel items in the **System and  
Security** category.  
3. Display the same list, excluding any Control Panel items that exist in more than one category. Make  
sure the command performance is optimized.

**Answers:**

help \*control\*

Get-ControlPanelItem

Get-ControlPanelItem –Category 'System and Security' | Sort Name

Get-ControlPanelItem -Category 'System and Security' | Where-Object -FilterScript {-not ($PSItem.Category -notlike '\*System and security\*')} | Sort Name

**Exercise 2: Enumerating objects**

The main tasks for this exercise are as follows:  
1. Display a list of files on the E: drive of your computer.  
2. Use enumeration to produce 100 random numbers.  
3. Run a method of a Windows Management Instrumentation (WMI) object.

**Task 1: Display a list of files on the E: drive of your computer**  
  
1. On **LON-CL1**, open **Windows PowerShell** as an administrator.  
2. Display a directory listing of all the items on the **E** drive. Include subfolders in the list.  
3. Display a list of all the files on the E: drive, without displaying directory names.

**Answers:**  
Get-ChildItem -Path E: -Recurse

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

Get-ChildItem -Path E: -Recurse | ForEach GetFiles

**Task 2: Use enumeration to produce 100 random numbers**1. Using a keyword such as **random**, find a command that produces random numbers.  
2. View the help for the command.  
3. Run **1..100** to put 100 numeric objects into the pipeline.  
4. Run the command again. For each numeric object, produce a random number that uses the numeric  
object as the seed.

**Answers:**  
help \*random\*

help Get-Random –ShowWindow

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

**Task 3: Run a method of a Windows Management Instrumentation (WMI) object**1. Close all applications other than the **Windows PowerShell** console.  
2. Run the command **Get-WmiObject -Class Win32\_OperatingSystem -EnableAllPrivileges**.  
3. Display the members of the object produced by the previous command.  
4. In the member list, find a method that restarts the computer.  
5. Run the command again, and use enumeration to run the method that restarts the computer.

**Answers:**

Get-WmiObject –Class Win32\_OperatingSystem -EnableAllPrivileges

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

Get-WmiObject –Class Win32\_OperatingSystem -EnableAllPrivileges | ForEach Reboot (reboot the machine you run it on)