CIS161 Assignment 1

HW 1.0

Objective:

1. Learn some basic cmdlets.

Upload your scripts to Blackboard by clicking the corresponding link.

Script 1.1:

The legacy command nslookup returns information about a host. Given an IP address nslookup returns the registered name provide the name is registered with DNS. We may execute nslookup from the command line as below:

```
PS D:\>
PS D:\>  nslookup 198.111.176.6
Server:  google-public-dns-a.google.com
Address:  8.8.8.8

Name:    www.wccnet.org
Address:  198.111.176.6
```

Use notepad or the ISE to create the script lookup.ps1 that returns the name associated with the IP address 198.111.176.6. The script does the following:

1. Pipe the output of **nslookup 198.111.176.6** into **select-string**.
2. The select-string cmdlet selects the line that contains the string "Name".

Execute the script from the console as below.

```
PS>
PS>  .\1.1-lookup.ps1

Name:    www.wccnet.org
```

This script uses the legacy command **nslookup** and the **select-string** cmdlet.

Script 1.2:

Alter script 1.1 so that an IP address is passed as a parameter to the script. The following are steps required to develop the script:

1. Add the syntax below as the first line in the script.

```
param ( $ipaddress )
```

2. Change the line containing the nslookup command accordingly.

Test the script with multiple address as shown below.

```
PS>
PS>  .\1.2-lookup.ps1 198.111.176.6

Name:    www.wccnet.org

PS>  .\1.2-lookup.ps1 8.8.8.8

Name:    google-public-dns-a.google.com
```

This script uses the legacy command **nslookup** and the **select-string** cmdlet.

Script 1.3:

Based upon what you learned in the last script, create a script called dirlist.ps1 that is passed a path as a parameter. The script uses get-childitem to list all objects in the given path. Do not show contents of any subdirectories, i.e., do not use -recurse.

Approach:

1. From the console execute get-childitem specifying the path c:\windows. Do not recurse subdirectories. Make sure this step works before proceeding.
2. Use Notepad to create script file with the command developed above. Make sure you enter the cmdlet syntax exactly as in step 1.
3. Test the script. Make sure it works with the hard-code path before proceeding. You should see no errors in the console.
4. Edit the script by replacing c:\windows with another directory path.
5. Test the script again. Make sure it works before proceeding. You should see no errors in the console.
6. Edit the script, adding the command below as the first line of the script:

```
param ( $path )
```

7. Make other changes as necessary.
8. Test the script by passing the path c:\windows. Test again with another path.

The script should runs as shown below.

```
PS>
PS>  .\1.3-dirlist.ps1 c:\windows

    Directory: C:\windows

Mode                LastWriteTime         Length Name
----                -------------         ------ ----
d-----        7/10/2015   7:04 AM                addins
d-----        8/14/2015  10:31 PM                ADFS
d-----        8/15/2015   3:34 AM                appcompat
```

This script uses the **get-childitem** cmdlet.

Script 1.4:

Write a script called **showEP.ps1** that shows either the TCP or UDP connections and/or endpoints of the host. First, write the script so that it shows the information for TCP. The script does the following:

1. Execute the legacy command as **netstat -na** piping the output into **select string**.
2. The select string cmdlet selects the lines that contain the string **TCP** and pipes the output into **Out-GridView**.

Running the script as this point as shown below:

```
PS>
PS>   .\1.4-showEP.ps1
PS>
```

Results in a window that looks like the image below.

netstat -na | select-string 'TCP' | out-gridview

Filter

Add criteria

IgnoreCase	LineNumber	Line					Filename	Path	Pattern	Context	Matches
True	5	TCP	0.0.0.0:135	0.0.0.0:0	LISTENING		InputStream	InputStream	TCP		{TCP}
True	6	TCP	0.0.0.0:445	0.0.0.0:0	LISTENING		InputStream	InputStream	TCP		{TCP}
True	7	TCP	0.0.0.0:2179	0.0.0.0:0	LISTENING		InputStream	InputStream	TCP		{TCP}

Once this script executes correctly. Alter the script so the protocol is **passed a parameter called $proto.** Use what you learned in the previous scripts to determine the necessary modifications.

A properly working script may be executed as shown below and produces the corresponding output in the Grid View window.

```
PS>
PS>
PS>   .\1.4-showEP.ps1 TCP
PS>
PS>   .\1.4-showEP.ps1 UDP
PS>
```

Script 1.5:

Write a script called **profile.ps1** that executes upon PowerShell startup that customizes the UI.as follows:

1. Changes the background to gray.
2. Changes the foreground color to blue.
3. Changes the PS window title to "<your name> PowerShell". Note, <your name> is your first and last name
4. Changes the cursor size to 50.
5. Executes the clear-host cmdlet

Approach: you will need to set $host.ui.rawui properties. Use get-member to see the properties. Create a script and test it. Once satisfied with the script, do the following:

1. In the c:\users\<profile>\documents directory, create directory called WindowsPowerShell.
2. Copy your script into this directory and name it profile.ps1.
3. Exit PowerShell then open a PowerShell console to see the effect.

Hint, wildcards may be used with help.