Powershell Lab 1.0-1.4

Total Time Taken – 2.5 Hours (Typing in the Commands and Verifying the Results also figuring out what the command does took time)

Issues Faced – Got Some Errors in Syntax which are fixed by carefully reviewing the Script Code

Name – Armaandeep Singh Brar

Student ID - 991708718

I Analysed all the Lab Steps after all this I found that Visual Studio Code is already Installed in my Computer.

As a Proof, I attached all the Screenshots below to show by every possible method that it is installed

.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Confirmed Installation of VS Code on my PC.

**Let’s Move on to Installing latest PowerShell 7.4 in my PC: -**

That is also already installed on my PC, but I remember that how I installed that. Now I am going to Explain that How I installed that?

Step 1 – I opened Command Prompt as an Administrator in my PC and enter the Command **winget install --id Microsoft.PowerShell** hit enter and it will display the packages from which we can install

Step 2 - When we again hit enter it will initiate the Windows Package Manager for Installation of Latest version of Powershell in PC.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Confirmed Installation of Latest Version of PowerShell which is 7.4.5**

**Now it’s Confirmed that Visual Studio Code and PowerShell’s Latest Version is Installed**

Let’s Begin Learning How to Install PowerShell Extension in Visual Studio Code and Use it for running Script Commands.

Step 1 – Installing Powershell Extension in VS code  
  
A screenshot of a computer

Description automatically generated

It’s Already Installed on my PowerShell as I did while in-Class Instructions from Professor.

**After Successful Installation of PowerShell, I have Created file named DemoExercise.ps1 in which I will be practicing Auto-Completion of Commands**

A screenshot of a computer

Description automatically generated

Let’s Just Type Get in the Command-line Interface and It will Display All the Commands starting from Get

A screenshot of a computer

Description automatically generated

**Now Let’s Run any Command for example I ran Get-Alias to check if it works**

A screenshot of a computer screen

Description automatically generated

**It is Successfully Working and Showing the Results in Terminal**

**Visual Studio Code Also helps in shortcut of Commands as I tested with GPS, After Clicking on Bulb Icon It Shows Possible options to replace the Commands**

**A screenshot of a computer

Description automatically generated**

**So, I ran Replaced Commands Get-Process and got the Result in Terminal**

**A screenshot of a computer

Description automatically generated**

**A screen shot of a computer

Description automatically generated**

**Let’s Explore the VS Code and it’s PowerShell Commands**

**Extension of PowerShell file should be always .ps1**

**F8 is used to run Selected Commands**

**For Running All Commands, we Must press F5**

**A screenshot of a computer

Description automatically generated**

**I copied the File Script and ran it in Terminal, It displays me the File Contents**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**I typed the Command and got the Result. This Command Reveals that What kind of Execution Policies are applied at User Level of Machine. As We can check process can be bypassed, local machine is Remote signed while others are undefined.**

**Get- Date**

**A screenshot of a computer

Description automatically generated**



**Get- Date -format hh:mm:ss**

**A screenshot of a computer

Description automatically generated**



**Difference Between Both Commands**

**After Running Both the Commands, I Observed that Get-Date Depicts the Full Format of Date and Time while other one depicts specified Format which we want to be displayed in result**

**A screenshot of a computer

Description automatically generated**

**It Depicts the Time only in Hours and Minutes Format**

**Command – Get-Module -ListAvailable**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer screen

Description automatically generated**

**Description – Depicts all the Modules installed on PC**

**Command - Get-Command -Name \*IP\***

**A screenshot of a computer screen

Description automatically generated**

**Command - Get-Command -Module NetTCPIP -Name \*IP\***

**A screenshot of a computer screen

Description automatically generated**

Difference Between Both Commands - Get-Command -Name \*IP\* filters commands with "IP" in their name.

Get-Command -Module NetTCPIP -Name \*IP\* restricts the command search to the NetTCPIP module.

A screenshot of a computer

Description automatically generated

After Running the Command - Start-Process notepad.exe $NotepadProc = Get-Process -Name notepad $NotepadProc.WaitForExit() Start-Process calc.exe . I run it and it Opened the Notepad and waited for it to exit and then opened the Calculator

**A screenshot of a computer screen

Description automatically generated**

This command displayed the properties and methods of the Process object.