POWERSHELL

- ➔ PowerShell is basically a scripting language and command line shell (means we write commands to perform the operations either on the OS or on the technology we are interacting with like azure).
- ➔ It is used to deal with object rather than just text. Meaning we can generate, manipulate, invoke methods, or change properties and extract information from these objects.
- ➔ **Example:**
  System administrator wants to create 'n' number of AD users, we can do so using PowerShell cmdlets.

PowerShell Cmdlets:

- ➔ PowerShell commands are known as cmdlets (Command-Lets).
- ➔ Syntax of cmdlets: Verb (Do something to)-Noun (this thing).
- ➔ PowerShell cmdlets are **case-Insensitive**.
- ➔ The output of a PowerShell cmdlet is always a .NET object.
- ➔ **Example:**
  Get-ChildItem -> It will display properties of files and folders that are there in the present directory.
  Get-ExecutionPolicy
  Get-Service
  Set-ExecutionPolicy Unrestricted
  Show-Command

PowerShell Scripting:

- ➔ Scripts in PowerShell are basically just text files with the special filename extension of .ps1
- ➔ To execute the script, either enter full path or write .\filename.ps1

Variables:

- ➔ Always have a dollar sign ($) before them.
- ➔ **Example:**
  $numberofcmdlets=(Get-Command).count [[Gives the Count of Commands]]
  $numberofcmdlets
- ➔ To get the type of the variable:
  $variablename.GetType()

Write to the Terminal:

- ➔ **Write-Host** cmdlet: Simply writes text to the screen of the machine hosting the PowerShell session.
- ➔ '' and "" different.
- ➔ "" consider $variablename a variable.
- ➔ To insert new line:
  Write-Host "`n"

Array:

- ➔ Can store multiple values of same or different types.
- ➔ Array size is fixed.
- ➔ **Example:**
  $variablename = @("item1", item2, "item3")
- ➔ Display value: $variablename[indexvalue] -> indexvalue starts from 0.
- ➔ Create array using:
  [system.collections.arraylist]$variablename=@(“item”, “item”, “item”)
- ➔ Add element to array:
  $variablename += @(“newitem”, “anotheritem”)
- ➔ Remove element from array:
  $variablename.RemoveAt(indexvalue)
  $variablename.Remove(“item”)
- ➔ Replace values in array:
  $variablename[indexvalue] = “updatedvalue”

Hash Table:

- ➔ We can store data in key-value pair using hash tables and it will give unordered table.
- ➔ KEY IS UNIQUE, VALUE CAN BE PRESENT MANY TIMES.

  $variablename = @{"key1" = "value"; "key2" = "value"}
  $variablename.keys --> Display keys
  $variablename.values --> Display values
  $variablename.count --> Display number of key-value pairs.

- ➔ Add key-value pair:
  $variablename.key = “value”
- ➔ Remove key-value pair:
  $variablename.Remove(“key”)

- ➔ **Example:**
  $StateCapitals = @{"North Carolina" = "Raleigh"; "California" = "Sacramento"; "New York" = "Albany"; "Florida" = "Tallahassee"; "Texas" = "Austin"}
  $StateCapitals