

NAME : B S SUDHEENDRA

EMAIL ID : bssudheendra2006@gmail.com

TOPIC - 1 : Powershell cmdlets commands

COMMANDS

Get-CimClass *network* | Sort CimClassName

Get-WmiObject -Class Win32_Keyboard

Get-WmiObject

Get-WmiObject -Namespace root -List | Out-File D:\Demo\Namespace.txt -Append

Get-WmiObject -Namespace root\cimv2 -List | Sort Name

Get-WmiObject -Namespace root\CIMv2 -List | Out-File D:\Demo\cimv2.txt -Append

Get-WmiObject -Namespace root\cimv2 -List | Sort Name | Out-File D:\Demo\sortcimv2.txt -Append

Get-CimClass -Namespace root\CIMv2 | Sort CimClassName

Get-WmiObject -Class CIM_Chip

Get-WmiObject -ClassName CIM_InstalledOS

Get-WmiObject -ClassName CIM_InstalledOS

Get-CimInstance -ClassName Win32_LogicalDisk

Get-CimInstance -Class Win32_LogicalDisk

Get-WmiObject -Class Win32_LogicalDisk

Get-WmiObject Win32_LogicalDisk

Get-WmiObject -Class Win32_LogicalDisk -Filter "DriveType=3"

Get-CimInstance -ClassName Win32_LogicalDisk -Filter "DriveType=3"

Get-WmiObject -Query "SELECT * FROM Win32_LogicalDisk WHERE DriveType = 3"

Get-CimInstance -Query "SELECT * FROM Win32_LogicalDisk WHERE DriveType = 3"

Get-NetIPConfiguration | Out-File D:\wipropowershell\datafiles.txt -Append

Get-NetIPConfiguration | Out-File -FilePath D:\aa\wiprofiledwc\netfileipconfigs

Get-Service | Out-File D:\aa\wiprofiledwc\processfile -Append

Get-PSDrive -PSProvider FileSystem | Out-File D:\wipropowershell\datefilea.txt -Append

Get-Command | Out-File D:\Demo\datefile.txt -Append

Get-PSDrive -PSProvider FileSystem | Out-File D:\Demo\datefile.txt -Append

Get-NetIPConfiguration | Out-File D:\Demo\datefile.txt -Append

Get-PSDrive -PSProvider FileSystem | Out-File D:\Demo\datefile.txt -Append

Get-ChildItem | Out-File D:\Demo\datefile.txt -Append

Get-PSDrive -PSProvider FileSystem | Out-File D:\wipropowershell\datafiles.txt -Append

Get-Service | Out-File D:\Demo\opfilecommands.txt -Append

Get-Process

Get-Process | Sort-Object CPU -Descending | Select-Object -First 5 | Select-Object Name, CPU

#THIS COMMAND USED FOR

\$name = Read-Host "please enter name"

Write-Output "Helo ,lets welcome \$name"

The screenshot shows the Windows PowerShell ISE interface. The main editor window has two tabs: 'Untitled1.ps1*' and 'Untitled2.ps1*'. The 'Untitled2.ps1' tab is active and contains the following script:

```
1 $name = Read-Host "please enter name"
2
3 Write-Output "Hello, lets welcome $name"
```

The console window below the editor shows the execution of this script. It prompts for a name and then outputs a message:

```
PS C:\Users\SUDHA>
$name = "John"
$age = 30
"My name is {0} and I am {1} years old" -f $name, $age

My name is John and I am 30 years old

PS C:\Users\SUDHA>
$name = Read-Host "please enter name"
Write-Output "Helo ,lets welcome $name"
please enter name: jhon
Helo ,lets welcome jhon

PS C:\Users\SUDHA>
```

The right-hand pane is titled 'Commands' and shows a list of PowerShell cmdlets under the 'A' category, including Add-AppxPackage, Add-AppxProvisionedPackage, Add-AppxVolume, Add-BitLockerKeyProtector, Add-BitsFile, Add-CertificateEnrollmentPolicyServer, Add-Computer, Add-Content, Add-DnsClientNrptRule, Add-DtcClusterTMMMapping, Add-EtwTraceProvider, Add-History, Add-InitiatorIdToMaskingSet, Add-JobTrigger, Add-KdsRootKey, and Add-LocalGroupMember. At the bottom of this pane are 'Run', 'Insert', and 'Copy' buttons.

The status bar at the bottom indicates 'Completed' on the left and 'Ln 3 Col 20' with a zoom level of '100%' on the right.

`$name = "John"`

`$age = 30`

`"My name is {0} and I am {1} years old" -f $name, $age`

The screenshot shows the Windows PowerShell ISE interface. The main editor window has one tab: 'Untitled1.ps1*'. The script in the editor is:

```
1
2 $name = "John"
3 $age = 30
4 "My name is {0} and I am {1} years old" -f $name, $age
5
```

The console window below the editor shows the execution of this script, which assigns values to variables and then outputs a formatted string:

```
PS C:\Users\SUDHA>
$name = "John"
$age = 30
"My name is {0} and I am {1} years old" -f $name, $age

My name is John and I am 30 years old

PS C:\Users\SUDHA> |
```

The right-hand pane is titled 'Commands' and shows the same list of PowerShell cmdlets as the first screenshot. At the bottom of this pane are 'Run', 'Insert', and 'Copy' buttons.

The status bar at the bottom indicates 'Completed' on the left and 'Ln 8 Col 20' with a zoom level of '100%' on the right.

```
$fruits = "banana","apple"
```

```
$fruits[0]
```

```
$fruits+="kiwi"
```

```
$fruits[2]
```

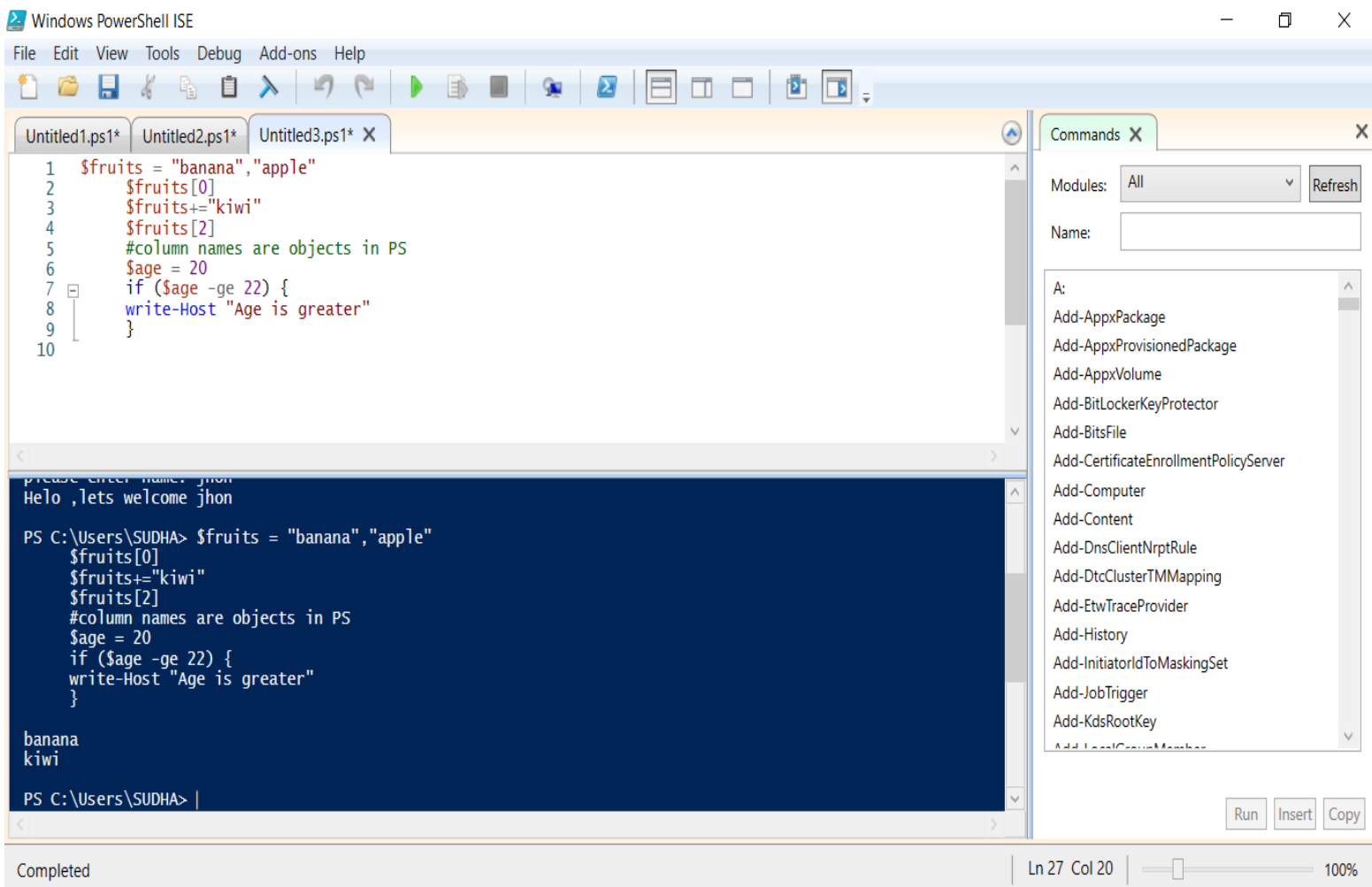
```
#column names are objects in PS
```

```
$age = 20
```

```
if ($age -ge 22) {
```

```
write-Host "Age is greater"
```

```
}
```



The screenshot displays the Windows PowerShell ISE interface. The main editor window shows a script with the following content:

```
1 $fruits = "banana","apple"
2 $fruits[0]
3 $fruits+="kiwi"
4 $fruits[2]
5 #column names are objects in PS
6 $age = 20
7 if ($age -ge 22) {
8     write-Host "Age is greater"
9 }
10
```

The output pane at the bottom shows the execution results:

```
PS C:\Users\SUDHA> $fruits = "banana","apple"
$fruits[0]
$fruits+="kiwi"
$fruits[2]
#column names are objects in PS
$age = 20
if ($age -ge 22) {
write-Host "Age is greater"
}

banana
kiwi
PS C:\Users\SUDHA>
```

The right-hand pane shows the 'Commands' window with a list of modules and a search bar. The status bar at the bottom indicates 'Completed' and 'Ln 27 Col 20'.

```
$age = 25
```

```
if ($age -ge 18) {
```

```
    Write-Host "Adult"
```

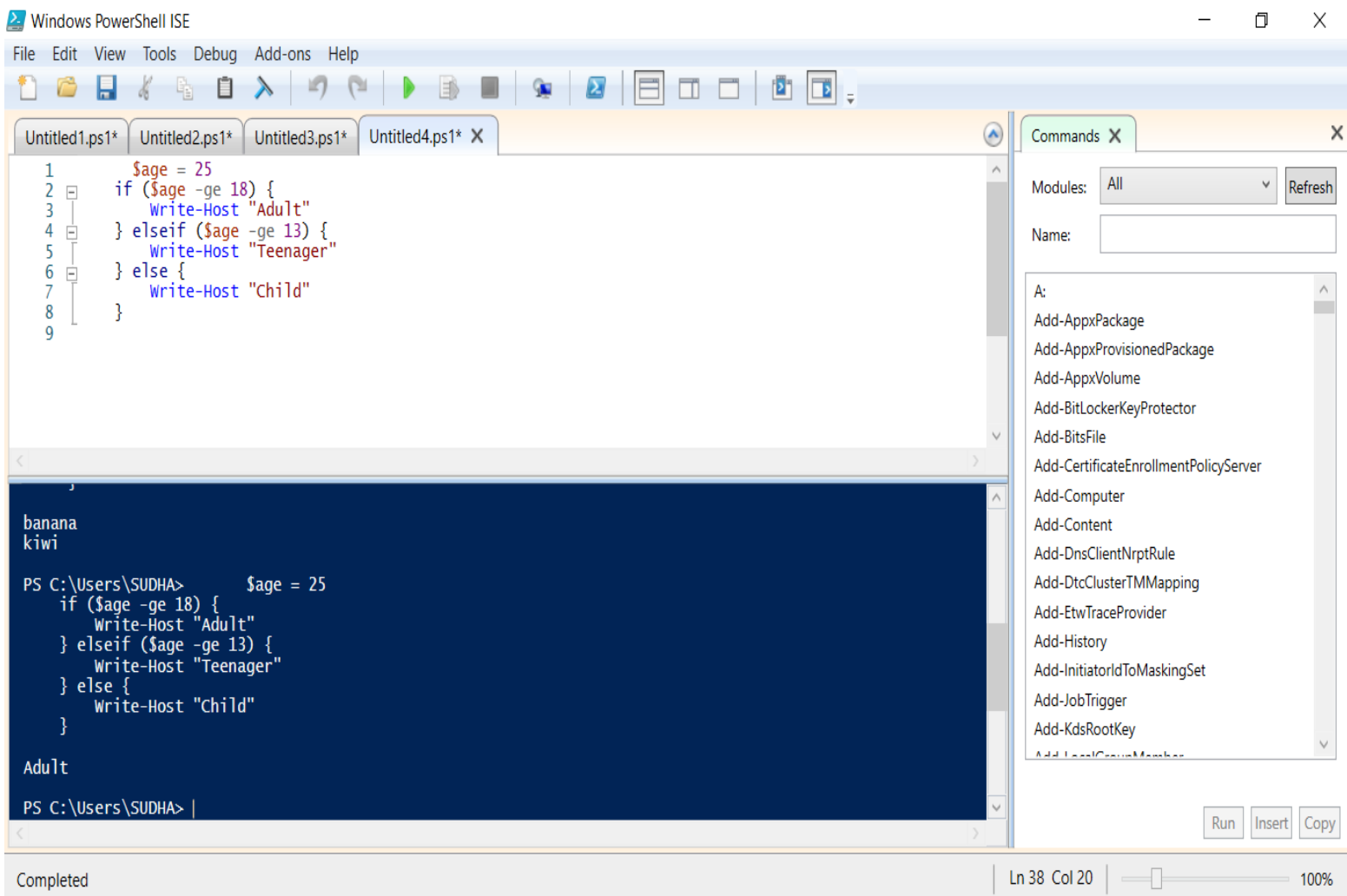
```
} elseif ($age -ge 13) {
```

```
    Write-Host "Teenager"
```

```
} else {
```

```
    Write-Host "Child"
```

```
}
```



The screenshot shows the Windows PowerShell ISE interface. The main editor window displays a PowerShell script with line numbers 1 through 9. The script sets a variable `$age` to 25 and uses an `if-elseif-else` block to write the age category to the host. The script is saved in a file named `Untitled4.ps1`. The output console at the bottom shows the execution of the script, displaying the output `Adult`. The right-hand pane shows the `Commands` window with a list of available commands, including `Add-AppxPackage`, `Add-AppxProvisionedPackage`, `Add-AppxVolume`, `Add-BitLockerKeyProtector`, `Add-BitsFile`, `Add-CertificateEnrollmentPolicyServer`, `Add-Computer`, `Add-Content`, `Add-DnsClientNrptRule`, `Add-DtcClusterTMMMapping`, `Add-EtwTraceProvider`, `Add-History`, `Add-InitiatorIdToMaskingSet`, `Add-JobTrigger`, `Add-KdsRootKey`, and `Add-LocalGroupMember`. The status bar at the bottom indicates that the script has been completed.

```
1 $age = 25
2 if ($age -ge 18) {
3     Write-Host "Adult"
4 } elseif ($age -ge 13) {
5     Write-Host "Teenager"
6 } else {
7     Write-Host "Child"
8 }
9
```

```
banana
kiwi

PS C:\Users\SUDHA> $age = 25
if ($age -ge 18) {
    Write-Host "Adult"
} elseif ($age -ge 13) {
    Write-Host "Teenager"
} else {
    Write-Host "Child"
}

Adult

PS C:\Users\SUDHA>
```

Get-Service | Get-Member | Out-File D:\Demo\opfilecommands.txt -Append

Get-Date | Get-Member | Out-File D:\Demo\opfilecommands.txt -Append

Get-Date | Select-Object -Property Second

Get-Date | Select-Object -Property TimeOfDay

Get-Command *hotfix* Out-File D:\Demo\opfilecommands.txt -Append

Get-Hotfix | Get-Member

00 -gt 10

500 -le 100

'hello' -eq 'HELLO'

'hello' -ceq 'HELLO'

Get-Service | Where status -ne Running | Out-File D:\Demo\a.txt -Append

Get-Service | Where status -ne Running

Get-process | Where CPU -gt 100

Get-process -Name EXCEL | Stop-Process

Stop-process -Name

Get-Service | ForEach StatusService | ForEach Displayname

Get-EventLog -List

Get-EventLog -List | Where Log -eq 'System'

Get-EventLog -List | Where Log -eq 'System'

Get-Process | ConvertTo-HTML

Get-Process | ConvertTo-HTML | Out-File D:\Demo\htmldata.txt

