$vms = Get-AzVm -ResourceGroupName AZUREB8

foreach ($vm in $vms) {

$size = $vm.HardwareProfile.VmSize

if ($size -eq 'Standard\_B1ls') {

Write-Output "$size ITS A VALID SIZE"

}

else {

Write-Output "$size ITS A INVALID SIZE"

}

}

##############################################################

$vms = Get-AzVm -ResourceGroupName AZUREB8

$vmnames = $vms.Name

foreach ($vm in $vmnames) {

if ($vm -like "STD\*") {

Write-Output "$vm is PROD VM"

}

else {

Write-Output "$vm is a DEV VM & Iam Reboot $vm"

}

}

###############################################################

$vms = Get-AzVm -ResourceGroupName AZUREB8

$vmnames = $vms.Name

for ($i = 0; $i -lt $vmnames.Length; $i++) {

if ($vmnames[$i] -like "STD\*") {

Write-Output $i

Write-Output $vmnames[$i]

Write-Output "$vmnames[$i] is PROD VM"

}

else {

Write-Output $i

Write-Output "$vmnames[$i] is a DEV VM & Iam Reboot $vm"

}

}

###############################################################

$vms = Get-AzVm -ResourceGroupName AZUREB8

$vmnames = $vms.Name

$i = 0

while ($i -lt $vmnames.Length) {

if ($vmnames[$i] -like "STD\*") {

$vm = $vmnames[$i]

Write-Output "$vm is PROD VM"

}

else {

$vm = $vmnames[$i]

Write-Output "$vm is DEV VM"

}

$i = ++$i

}

#####################################################################

#Woring With Arrays

$MyArray = @('Anand', 'Bala', 'Chitra', 'David')

$i = 0

foreach ($person in $MyArray) {

Write-Output "The Name of the person in Position $i is: $person"

$i = $i + 1

}

####################################################################

$myhash = @{Number = 1; Shape = "Square"; Color = "Blue" }

Write-Output $myhash.Keys

Write-Output $myhash.Values

Write-Output $myhash.count

#3

Write-Output $myhash['Color']

#Blue

Write-Output $myhash['Number']

#1

Write-Output $myhash['Shape']

#Square

###################################################################

echo "Running the script for Stanard Load Balancer now..!!"

echo "Creating Azure Resource Group"

az group create --location eastus -n AZUREB8

echo "Creating Azure Virtual Network"

az network vnet create -g AZUREB8 -n AZUREB8-vNET1 --address-prefix 10.1.0.0/16 \

--subnet-name AZUREB8-Subnet-1 --subnet-prefix 10.1.1.0/24 -l eastus

echo "Creating Azure Subnets"

az network vnet subnet create -g AZUREB8 --vnet-name AZUREB8-vNET1 -n AZUREB8-Subnet-2 \

--address-prefixes 10.1.2.0/24

az network vnet subnet create -g AZUREB8 --vnet-name AZUREB8-vNET1 -n AZUREB8-Subnet-3 \

--address-prefixes 10.1.3.0/24

echo "Creating Azure NSG & Rules"

az network nsg create -g AZUREB8 -n AZUREB8\_NSG1

az network nsg rule create -g AZUREB8 --nsg-name AZUREB8\_NSG1 -n AZUREB8\_NSG1\_RULE1 --priority 100 \

--source-address-prefixes '\*' --source-port-ranges '\*' --destination-address-prefixes '\*' \

--destination-port-ranges '\*' --access Allow --protocol Tcp --description "Allowing All Traffic For Now"

az vm create --resource-group AZUREB8 --name STDLBVM1 --image UbuntuLTS --vnet-name AZUREB8-vNET1 \

--subnet AZUREB8-Subnet-1 --admin-username testuser --admin-password "India@123456" --size Standard\_B1ls \

--nsg AZUREB8\_NSG1 --zone 1 --custom-data cloud-init.txt

az vm create --resource-group AZUREB8 --name STDLBVM2 --image UbuntuLTS --vnet-name AZUREB8-vNET1 \

--subnet AZUREB8-Subnet-1 --admin-username testuser --admin-password "India@123456" --size Standard\_B1ls \

--nsg AZUREB8\_NSG1 --zone 2 --custom-data cloud-init.txt

az vm create --resource-group AZUREB8 --name STDLBVM3 --image UbuntuLTS --vnet-name AZUREB8-vNET1 \

--subnet AZUREB8-Subnet-1 --admin-username testuser --admin-password "India@123456" --size Standard\_B1ls \

--nsg AZUREB8\_NSG1 --zone 3 --custom-data cloud-init.txt

az vm create --resource-group AZUREB8 --name B1SVM4 --image UbuntuLTS --vnet-name AZUREB8-vNET1 \

--subnet AZUREB8-Subnet-1 --admin-username testuser --admin-password "India@123456" --size Standard\_B1s \

--nsg AZUREB8\_NSG1 --zone 3 --custom-data cloud-init.txt

ls

==============TRY-CATCH-FINALLY===============

Try { Connect-AzureAD -ErrorAction Stop}

Catch [Microsoft.Open.Azure.AD.CommonLibrary.AadAuthenticationFailedException] { "Authentication failed." }

Catch [Microsoft.IdentityModel.Clients.ActiveDirectory.AdalServiceException] { "It looks like someone canceled the login process." }

Catch [System.AggregateException] { "Anything else"}

Finally { "Thanks for playing!" }

===============FUNCTION==========================

function azure {

Try { Connect-AzureAD -ErrorAction Stop}

Catch [Microsoft.Open.Azure.AD.CommonLibrary.AadAuthenticationFailedException] { "Authentication failed." }

Catch [Microsoft.IdentityModel.Clients.ActiveDirectory.AdalServiceException] { "It looks like someone canceled the login process." }

Catch [System.AggregateException] { "Anything else"}

Finally { "Thanks for playing!" }

}

azure

=====================BREAK-CONTINUE---------------------

function hello {

foreach ($i in 1..10) {

if ($i % 2 -eq 0) {

Write-Output "$i is EVEN"

#break

continue

}

else {

Write-Output "$I is ODD"

}

}

}

hello

======================FUNCTION-RETURN========================

function hello {

[int]$a = Read-Host "Enter The Number"

return $a

}

[int]$x = hello

[int]$y = $x+90

Write-Output $y