```
#Remove-ExchangeForwardingRuleRandom.ps1
#Written by: Jeff Brusoe
#Last Updated: October 7, 2020
#
#This file randomly pulls mailboxes to search them for forwarding rules.

[CmdletBinding()]
param (
  [ValidateNotNullOrEmpty()]
  [int]$MaximumSearches  = 2000
)

$Error.Clear()

try {
  Set-HSCEnvironment -ErrorAction Stop
  Connect-HSCExchangeOnline -ErrorAction Stop
}
catch {
  Write-Warning "Unable to configure HSC environment. Program is exiting."
  Invoke-HSCExitCommand
}

$MailboxCount = 0
Write-Output "Reading exclusion lists"

try {
  #Exclusion lists should be in same directory as ps1 file
  $ExclusionList1 = Import-Csv ExclusionList1.csv
  $ExclusionList2 = Import-Csv ExclusionList2.csv
}
catch {
  Write-Warning "Unable to read HSC exclusion lists. Program is exiting"
  Invoke-HSCExitCommand
}

#Forwarding rule log files
try {
  $ForwardingRuleLogFile = (Get-Date -format yyyy-MM-dd-HH-mm) + "-ExchangeForwardingFules.txt"
  New-Item -Name $ForwardingRuleLogFile -Path "$PSScriptRoot\Logs\" -Force -type file -ErrorAction Stop
  Add-Content -Path "$PSScriptRoot\Logs\$ForwardingRuleLogFile" -Value "PrimarySMTPAddress,RuleName,RuleForwardTo" -ErrorAction Stop
}
catch {
  Write-Warning "Unable to generate forwarding rule log"
}

try {
  Write-Output "Getting mailbox list"
  $Mailboxes = Get-Mailbox -ResultSize Unlimited -ErrorAction Stop |
    Where-Object {$_.PrimarySMTPAddress -notlike "*rni.*" -AND $_.PrimarySMTPAddress -notlike "*wvurni*"}
}
catch {
  Write-Warning "Unable to generate list of HSC mailboxes"
  Invoke-HSCExitCommand
}

```

```powershell
Write-Output "Processing Mailboxes"
$RuleDeletes = 0
$SearchCount = 0
$MailboxTotal = $Mailboxes.Count
Write-Output "Mailbox Total: $MailboxTotal"

while ($SearchCount -lt $MaximumSearches)
{

  $MailboxCount++
  Write-Output "Mailbox Count: $MailboxCount"

  Write-Output "Rule Deletes: $RuleDeletes"

  $MailboxSelected = Get-Random -Minimum 0 -Maximum $MailboxTotal
  Write-Output "Mailbox Selected: $MailboxSelected"

  try {
    $Mailbox = $Mailboxes[$MailboxSelected]

    Write-Output $("Primary SMTP Address: " + $Mailbox.PrimarySMTPAddress)
  }
  catch {
    Write-Warning "Unable to properly select mailbox"
    $SearchCount++

    Continue
  }

  try {
    $Rules = Get-InboxRule -Mailbox $Mailbox.PrimarySMTPAddress -ErrorAction Stop
  }
  catch {
    Write-Warning "Unable to get list of rules for this mailbox"
  }

  Write-Output $("Rule Count: " + $Rules.Count)

  foreach ($Rule in $Rules)
  {
    Write-Output $("Rule Name: " + $Rule.Name)
    Write-Output $("Rule Description: " + $Rule.Description)
    Write-Output "Rule Deletes: $RuleDeletes"

    if (($ExclusionList1.RuleName -contains $Rule.Name) -AND ($ExclusionList1.Mailbox -contains $Mailbox.PrimarySMTPAddress))
    {
      Write-Output "Rule is in ExclusionList1 and is being skipped"
    }
    elseif ($ExclusionList2.PrimarySMTPAddress -contains ($Mailbox.PrimarySMTPAddress) -AND (($ExclusionList2.ForwardingAddress -contains $rule.ForwardTo) -OR ($ExclusionList2.ForwardingSmtpoAddress -contains $rule.ForwardTo)))
    {
      Write-Output "Rule is excluded based on exclusion list 2"
    }
    elseif (![string]::IsNullOrEmpty($rule.ForwardTo))
    {
      Write-Output $("Forward To: " + $rule.ForwardTo)
      Add-Content -Path "$PSScriptRoot\Logs\$ForwardingRuleLogFile" -Value $($mailbox.PrimarySMTPAddress + "," + $rule.Name + "," + $rule.ForwardTo)

      [string]$ForwardTo = ($rule.ForwardTo -split " ") -like "*smtp*"
      Write-Output $("ForwardTo[1]: " + $ForwardTo)

      if (($ForwardTo.indexOf("@") -gt 0) -AND ($ForwardTo.indexOf("hsc.wvu.edu") -lt 0))
      {
        Write-Output "Rule will be deleted"

        try {
          Remove-InboxRule -Identity $Rule.Identity -Confirm:$false -ErrorAction Stop
          Write-Output "Successfully deleted rule."

          $RuleDeletes++
        }
        catch {
          Write-Output "Error removing forwarding rule"
          #Invoke-HSCExitCommand
        }
      }
      else
      {
        Write-Output "Rule will not be deleted"
      }
    }
    else {
      Write-Output "Rule is not a forwarding rule"
    }

      Write-Output "------------------------------------"
  }

  $SearchCount++
  Write-Output "****************************"
}

Invoke-HSCExitCommand -ErrorCount $Error.Count
```