```
<#
  .SYNOPSIS
    This file disables accounts in the New Users OU and moves them out of there if
    they have been there for more than 60 days.

  .PARAMETER MoveDays
    A flag used to indicate that a user should be moved from the NewUsers OU.

  .PARAMETER MoveTargetName
    The OU to move a user to (the FromNewUsers OU)

  .PARAMETER OUsToDisable
    This is an array which contains the OUs that are to be disabled

  .NOTES
    Author: Jeff Brusoe
    Last Updated by: Jeff Brusoe
    Last Udated: September 1, 2021
#>

[CmdletBinding()]
param (
    [ValidateRange(-75,75)]
  [int]$MoveDays = 60,

  [ValidateNotNullOrEmpty()]
  [string]$MoveTargetName = "FromNewUsers",

  [ValidateNotNullOrEmpty()]
  $OUsToDisable = @(
        "NewUsers",
        "FromNewUsers",
        "DisabledDueToInactivity2"
      )
)

try {
  Write-Verbose "Configuring Environment"

  Import-Module ActiveDirectory -ErrorAction Stop
  Set-HSCEnvironment -ErrorAction Stop

  $NewDisabledAccounts = 0
  $NewAccountMoves = 0

  if ($MoveDays -gt 0) {
    $MoveDays = -1*$MoveDays
  }
  elseif ($MoveDays -eq 0) {
    Write-Warning "Invalid move days entry"
    Invoke-HSCExitCommand -ErrorCount $Error.Count
  }

  Write-Output "Move Target Name: $MoveTargetName"
  $MoveTargetDN = (Get-ADOrganizationalUnit -Filter * -ErrorAction Stop |
            Where-Object {$_.Name -eq $MoveTargetName}).DistinguishedName
}
catch {
  Write-Warning "Unable to configure environment"
  Invoke-HSCExitCommand -ErrorCount $Error.Count
```

```
}

foreach ($OUToDisable in $OUsToDisable)
{
    Write-Output "OU to Disable: $OUToDisable"

    try {
    $CurrentOU = Get-ADOrganizationalUnit -Filter * -ErrorAction Stop |
      Where-Object {$_.Name -eq $OUToDisable}
    }
    catch {
        Write-Warning "Unable to find OU: $OUToDisabled"
        break
    }

  $SearchBase = $CurrentOU.DistinguishedName
    Write-Output "OU Distinguished Name:"
    Write-Output $SearchBase

    #Get list of AD Users
    try {
        Write-Output "Getting AD user list"
    $Properties = @(
            "extensionAttribute1",
            "extensionAttribute7",
            "whenCreated",
            "whenChanged"
          )

    $GetADUserParams = @{
      Filter = "*"
      SearchBase = $SearchBase
      ErrorAction = "Stop"
      Properties = $Properties
    }

        $ADUsers = Get-ADUser @GetADUserParams
    }
    catch {
        Write-Warning "Unable to generate AD user list"
        Invoke-HSCExitCommand -ErrorCount $Error.Count
    }

    foreach ($ADUser in $ADUsers)
    {
        Write-Output $("SamAccountName: " + $ADUser.SamAccountName)
      Write-Output $("extensionAttribute1: " + $ADUser.extensionAttribute1)
      Write-Output $("extensionAttribute7: " + $ADUser.extensionAttribute7)
    Write-Output $("Account Creation Date: " + $ADUser.whenCreated)
    Write-Output $("Account Last Changed: " + $ADUser.whenChanged)
    Write-Output $("Enabled: " + $ADUser.Enabled)
    Write-Output "Distinguished Name:"
    Write-Output $ADUser.DistinguishedName

    if ($ADUser.Enabled)
    {
      try {
        Write-Output "Disabling Account"
        $ADUser | Disable-ADAccount -ErrorAction Stop

        $NewDisabledAccounts++
        Write-Output "`nTotal New Disabled Accounts: $NewDisabledAccounts"
      }
      catch {
        Write-Warning "Unable to disable account"
      }
    }

    [DateTime]$AccountCreated = Get-Date $ADuser.whenCreated
    [DateTime]$AccountChanged = Get-Date $ADUser.whenChanged

    if (($AccountCreated -lt (Get-Date).AddDays($MoveDays)) -AND
      ($OUToDisable -eq "NewUsers"))
    {
      try {
        if ($AccountChanged -lt (Get-Date).AddDays(-1)) {
          Write-Output "Account is being moved."
          $ADUser | Move-ADObject -TargetPath $MoveTargetDN -ErrorAction Stop

          $NewAccountMoves++
          Write-Output "Total New Account Moves: $NewAccountMoves"
        }
        else {
          Write-Output "Account is not being moved due to change date"
        }
      }
      catch {
        Write-Warning "Unable to move user"
      }
    }

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

Write-Output "Total New Disabled Accounts: $NewDisabledAccounts"
Write-Output "Total New Account Moves: $NewAccountMoves"

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