```powershell
<#
  .SYNOPSIS
    This file looks at all accounts that have had their passwords changed in the last 7 days. For these accounts,
    it sets the BlockCredential attribute of the MSOL User object to false.

  .DESCRIPTION
    Requires
    1. Connection to the HSC tenant (Get-MsolUser etc)
    2. Connection to Exchange online and PowerShell cmdlets

  .PARAMETER ChangeDays
    This parameter specifies how many days back to look for password changes.

  .NOTES
    Author: Jeff Brusoe
    Last Update: September 5, 2020
#>

[CmdletBinding()]
[Diagnostics.CodeAnalysis.SuppressMessageAttribute("PSAvoidTrailingWhiteSpace","",Justification = "Not relevant")]
[Diagnostics.CodeAnalysis.SuppressMessageAttribute("PSAvoidUsingCmdletAliases","",Justification = "Only MS Provided Aliases are Used")]
param (
  [int]$ChangeDays = 2 #How far back to look for any password changes
)

#Configure environment
try {
  Set-HSCEnvironment -ErrorAction Stop
  Connect-HSCOffice365MSOL -ErrorAction Stop
}
catch {
  Write-Warning "Error configuring environment. Program is exiting."
  Invoke-HSCExitCommand
}

if ($ChangeDays -gt 0)
{
  $ChangeDays = $ChangeDays * (-1)
}

#The logs2 directory is one where logs are written every 2 hours.
Remove-HSCOldLogFile -Path "$PSScriptRoot\Logs2\" -TXT -Delete

$BlockCredentialSet = "$PSScriptRoot\Logs\" + (Get-Date -Format yyyy-MM-dd-HH-mm) + "-BlockCredentialSet.txt"
New-Item -type file -path $BlockCredentialSet

#Generating user list where users are
#1. not disabled
#2. password last set within past $ChangeDays
try
{
  Write-Output "Generating list of AD users"
  $users = Get-ADuser -Filter * -Properties PasswordLastSet | where {($_.Enabled) -AND ($_.PasswordLastSet -gt (Get-Date).AddDays($ChangeDays))}
}
catch
```

```
{
  Write-Warning "Unable to query AD users. Program is exiting."
  Invoke-HSCExitCommand
}

$count = 0

foreach ($user in $users)
{
  $count++

  Write-Output $("Current User: " + $user.UserPrincipalName)
  Write-Output $("Password Last Set Date: " + $user.PasswordLastSet.toString())

  Add-Content -path $BlockCredentialSet -value $("Current User: " +
$user.userprincipalname)
  Add-Content -path $BlockCredentialSet -value $("Password Last Set Date: " +
$user.PasswordLastSet.toString())
  Add-Content -path $BlockCredentialSet -value
$("************************************")

  #find user in cloud
  Write-Verbose "Searching for user in the cloud"

  try {
    $MSOLUser = Get-MsolUser -SearchString $user.userprincipalname -ErrorAction
Stop
  }
  catch {
    Write-Warning "Unable to find MSOLUser object"
  }

  if ($null -ne $MSOLUser)
  {
    Write-Output $("Block Credential: " + $MSOLUser.BlockCredential)
    Write-Output "User was found in the cloud."

    try
    {
      $MSOLUser | Set-MsolUser -BlockCredential $false -ErrorAction Stop
      Write-Output "Successfully set block credential"
    }
    catch
    {
      Write-Warning "Error setting user's block credential"
    }
  }
  else
  {
    Write-Output "User not found in the cloud."
  }

  Write-Output "Count: $count"
  Write-Output "**************************"
}

#Copy log files every two hours to Logs2 directory.
$CurrentDate = Get-Date
if (($CurrentDate.Hour % 2 -eq 0) -AND (($CurrentDate.Minute -gt 56) -OR
($CurrentDate.Minute -lt 4)))
```

```powershell
{
  $FilesToCopy = Get-HSCLastFile -DirectoryPath "$PSScriptRoot\Logs\" -
NumberOfFiles 2
  Write-Output "Copying files:"
  Write-Output $FilesToCopy

  $FilesToCopy | Copy-Item -Destination "$PSScriptRoot\Logs2"
}

Invoke-HSCExitCommand
```