```powershell
#--------------------------------------------------------------------------------------------------
#1Verify-BlockCredential.ps1
#
#Written by: Jeff Brusoe
#
#Last Modified: May 20, 2020
#
#Version: 1.2
#
#Purpose: 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.
#
#This file assumes a connection to the HSC Office 365 tenant has been established. If it isn't, then it will
#look for the Connect-ToOffice365-MS1.ps1 file to attempt a connection.
#--------------------------------------------------------------------------------------------------

<#
.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
    Paramter information.

.NOTES
    Author: Jeff Brusoe
    Last Update: May 20, 2020
#>

[CmdletBinding()]
param (
    #Common HSC PowerShell Parameters
    [switch]$NoSessionTranscript,
    [string]$LogFilePath = "$PSScriptRoot\Logs\",
    [switch]$StopOnError, #$true is used for testing purposes
    [int]$DaysToKeepLogFiles = 5, #this value used to clean old log files

    #File specific param
    [int]$ChangeDays = 7 #How far back to look for any password changes
)

##############################
#Import HSC PowerShell Modules#
##############################

#Build path to HSC PowerShell Modules
$PathToHSCPowerShellModules = $PSScriptRoot
$PathToHSCPowerShellModules = $PathToHSCPowerShellModules.substring(0,$PathToHSCPowerShellModules.lastIndexOf("\")+1)
$PathToHSCPowerShellModules += "1HSC-PowerShell-Modules"
Write-Output $PathToHSCPowerShellModules

#Attempt to load common code module
$CommonCodeModule = $PathToHSCPowerShellModules + "\HSC-CommonCodeModule.psm1"
Write-Output "Path to common code module: $CommonCodeModule"
Import-Module $CommonCodeModule -Force -ArgumentList $NoSessionTranscript,$LogFilePath,$true,$DaysToKeepLogFiles
```

```

#Attempt to load HSC Office 365 Module
$Office365Module = $PathToHSCPowerShellModules + "\HSC-Office365Module.psm1"
Write-Output "Path to HSC Office 365 module: $Office365Module"
Import-Module $Office365Module -Force

if ($Error.Count -gt 0)
{
    #Any errors at this point are from loading modules. Program must stop.
    Write-Warning "There was an error configuring the environment. Program is exiting."
    Exit-Commands
}

######################################
#End of Import HSC PowerShell Modules#
######################################

#############################
#Configure environment block#
#############################
Write-Output "Getting Parameter Information"
Get-Parameter -ParameterList $PSBoundParameters

Set-Environment
Set-WindowTitle

#See this page to understand what is going on here.
#https://www.thecloudjournal.net/2016/07/create-your-own-powershell-module-for-exchange-online/
##ConnectTo-Office365 #from Office 365 module
##Import-Module ExchangeOnline -Force #comes from HSC-OFfice365Module.psm1
& "C:\Users\microsoft\Documents\GitHub\HSC-PowerShell-Repository\6OldConnectionFiles\Connect-ToOffice365-MS4.ps1"

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

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

########################################
#End of environment configuration block#
########################################

#Generating user list where users are
#1. not disabled
#3. password last set $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."
    Exit-Command
}

$count = 0

foreach ($user in $users)
{
```

```powershell
    $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"
    $MSOLUser = Get-MsolUser -SearchString $user.userprincipalname

    if ($MSOLUser -ne $null)
    {
        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 "***************************"
}

if (!$NoSessionTranscript)
{
    Stop-Transcript
}
```