```powershell
#Enable-AccountAfterPasswordReset.ps1
#Written by: Jeff Brusoe
#Last Updated: May 22, 2020
#
#Purpose: This file enables AD accounts after the password has been updated.

[CmdletBinding()]
param (
    #Common HSC PowerShell Parameters
    [switch]$NoSessionTranscript,
    [string]$LogFilePath = "$PSScriptRoot\Logs\",
    [switch]$StopOnError, #$true is used for testing purposes
    [int]$DaysToKeepLogFiles = 7, #this value used to clean old log files

    #File specific parameters
    [int]$Minutes = 60,
    [switch]$Testing
)

$Error.Clear()

################################
#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
$ADModule = $PathToHSCPowerShellModules + "\HSC-ActiveDirectoryModule.psm1"
Write-Output "Path to HSC Active Directory Module: $ADModule"
Import-Module $ADModule -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-Command
}

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

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

Set-Environment
Set-WindowTitle

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

```

```
try
{
    $users = Get-ADUser -Filter * -Properties PasswordLastSet -ErrorAction Stop | where
    {$_.PasswordLastSet -gt (Get-Date).AddMinutes(-1*$Minutes)}
}
catch
{
    Write-Warning "There was an error generating the AD user list. Program is exiting."
    Exit-Command
}

foreach ($user in $users)
{
    Write-Output $("Current user: " + $user.SamAccountName)
    Write-Output $("PasswordLastSet: " + $user.PasswordLastSet)
    Write-Output $("Enabled: " + $user.Enabled)

    try
    {
        if (!$user.Enabled)
        {
            Write-Output "Attempting to enable user"

            if (!$Testing)
            {
                $user | Enable-ADAccount -ErrorAction Stop
            }

            Write-Output "User enabled."
        }
        else
        {
            Write-Output "User is already enabled."
        }
    }
    catch
    {
        Write-Warning "Unable to enable account"

        if ($Testing)
        {
            Stop-Transcript
            return
        }
    }

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

Exit-Command
```