```powershell
#Enable-AccountAfterPasswordReset.ps1
#Written by: Jeff Brusoe
#Last Updated: January 4, 2021
#
#This file enables an AD account after a password reset.

[CmdletBinding()]
param (
  [ValidateNotNullOrEmpty()]
  [int]$Minutes = 60,

  [ValidateNotNullOrEmpty()]
  [string]$LogsToSaveDirectory = "$PSScriptRoot\LogsToSave\",

  [switch]$Testing

)

try {
  $SessionTranscriptFile = Set-HSCEnvironment -ErrorAction Stop
}
catch {
  Write-Warning "Unable to configure environment. Pogram is exiting"
  Invoke-HSCExitCommand -ErrorCount $Error.Count
}

try
{
  $ADUsers = Get-ADUser -Filter * -Properties PasswordLastSet -ErrorAction Stop |
    Where-Object {$_.PasswordLastSet -gt (Get-Date).AddMinutes(-1*$Minutes)}
}
catch
{
  Write-Warning "There was an error generating the AD user list. Program is
exiting."
  Invoke-HSCExitCommand -ErrorCount $Error.Count
}

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

  $NewEnable = $false

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

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

      Write-Output "User enabled."
      $NewEnable = $true
    }
    else {
```

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

    if ($Testing) {
      Write-Warning "Program is exiting due to testing parameter"
      Invoke-HSCExitCommand -ErrorCount $Error.Count
    }
  }

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

if ($NewEnable) {
  #An account has been enabled. Copy session transcript to lots to save directory.
  try {
    Copy-Item -Path $SessionTranscriptFile -Destination $LogsToSaveDirectory
  }
  catch {
    Write-Warning "Error moving session transcript to logs to save directory"
  }

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