# Respuestas al Ejercicio 1 del TP 1

1. **En base al siguiente script de Powershell se pide que lo ejecute y analice y en base a los resultados responda las preguntas que figuran más abajo.**

Param (

[Parameter(Position = 1, Mandatory = $false)][String] $pathsalida = ".",

[int] $cantidad = 3

)

$existe = Test-Path $pathsalida

if ($existe -eq $true)  
 {

$listaproceso = Get-Process

foreach ($proceso in $listaproceso)

{

$proceso | Format-List -Property Id,Name >> procesos.txt

}

for ($i = 0; $i -lt $cantidad ; $i++)

{

Write-Host $listaproceso[$i].Name - $listaproceso[$i].Id }

}

else

{

Write-Host "El path no existe"

}

1. **¿Cuál es el objetivo del script?**

El Objetivo del script es generar un listado de los procesos en ejecución.

1. **¿Agregaría alguna otra validación a los parámetros?¿Cuál/es?**

Agregaríamos la validación del Length de parámetro $pathsalida para que sea consistente con el largo de nombre de archivo especificado por la plataforma .Net. Adicionalmente, agregaríamos el valor por defecto ".\" a $pathsalida en lugar de "." para poder utilizarlo cómodamente dentro del script y usaríamos $pathsalida como directorio seleccionado para alojar el archivo procesos.txt, ya que no se esta utilizando este parámetro mas que para validar la existencia del path. Así podríamos seleccionar el directorio donde serán depositados los resultados indicándolo en la línea de comandos.

Agregaríamos a la validación de $pathsalida la posibilidad de detectar si estamos hablando de una ruta valida hacia un directorio, ya que de la forma actual, si ingresamos una ruta a un archivo existente, cualquiera sea este, la validación de la existencia del path también será verdadera y la ejecución del script se llevara a cabo de todas formas sobre el directorio de actual. Sin embargo, la validación de un path cualquiera y el path de salida real no son consistentes. Por esto, hemos vinculado el parámetro $pathsalida al directorio de salida de los resultados, y así, cobran sentido tanto la validación de la existencia del path recibido como la existencia del propio parámetro.

Fijaríamos el parámetro $cantidad a la posición 2 para que sea posible ejecutar el script omitiendo los nombres de ambos parámetros, proporcionando flexibilidad al uso del mismo.

una vez obtenida la lista de objetos proceso, verificaríamos si la cantidad ingresada en $cantidad supera a la cantidad de procesos obtenidos en $listaprocesos, de ser así, tomaríamos los recaudos necesarios para no intentar mostrar por pantalla mas procesos de los que realmente hay.

A continuación se detalla el código con las modificaciones.

Param

(

# Validamos el length de $pathsalida para que no supere el largo permitido por windows.

[ValidateLength(1,260)][Parameter(Position = 1, Mandatory = $false)][String] $pathsalida = ".\",

# Fijamos $cantidad en la posicion 2 para poder ejecutar el script sin el nombre del parametro.

[Parameter(Position = 2, Mandatory = $false)][int] $cantidad = 3

)

# Evaluamos si el path proporcionado es un directorio.

$esdirectorio = Test-Path $pathsalida -IsValid -PathType Container

if ($esdirectorio -eq $true) {

# Ahora evaluamos si existe.

$existe = Test-Path $pathsalida

if ($existe -eq $true) {

# Usoamos el path seleccionado, haciendo que la validacion del path tenga sendtido.

$archivo = $pathsalida + "procesos.txt"

$listaproceso = Get-Process

foreach ($proceso in $listaproceso) {

$proceso | Format-List -Property Id,Name >> $archivo

}

# Reasignamos la cantidad solicitada, para no mostrar mas Procesos de los que hay.

if ($cantidad -gt $listaproceso.Count) {

$cantidad = $listaproceso.count

}

for ($i = 0; $i -lt $cantidad; $i++) {

Write-Host $listaproceso[$i].Name - $listaproceso[$i].Id

}

} else {

Write-Host "El directorio no existe."

}

} else {

Write-Host "El path no es un directorio."

}