	Ciclo: ASIR	Departamento de Informática <small>www.informaticarodrigo.com.es</small> Informática Rodrigo Caro
	Módulo: Implantación de Sistemas Operativos	

Bloque:II	Administración de W10
Unidad:	
Práctica/Tarea:1	Práctica: PowerShell Scripts básicos
Nombre alumno:	

1.- Realiza un script: ArchivosGrandes1.ps1

Descripción: El script buscará aquellos archivos cuyo tamaño sea superior a uno dado.

Entrada: El script preguntará:

Ruta para buscar archivos.

Tamaño a partir del cual buscaremos: (Ejemplo: 100MB,2GB, etc)

Salida: Muestra aquellos archivos que sean mayores que el indicado.

Se exportará los resultados a un archivo html: ArchivosGrandes.html


Observaciones: Dado que el usuario nos puede dar una ruta en la que no tenga permisos, vamos a decirle que no nos muestre esos warnings, con la opción:

-ErrorAction SilentlyContinue

Detrás del comando.

Test de pruebas: Para probar si funciona, puedes poner como ruta c:\

Pista: Pista: Where-object y ConvertTo-HTML

	Ciclo: ASIR	Departamento de Informática <small>www.informaticarodrigo.com.es</small> Informática Rodrigo Caro
	Módulo: Implantación de Sistemas Operativos	

2.- Realiza un script: ArchivosGrandes2.ps1

Descripción: El script buscará aquellos archivos **en el directorio de trabajo del usuario cuyo tamaño sea superior a uno pasado por parámetro** en la ejecución del script.

Hay que diseñarlo, de forma que el script sea válido, independiente del usuario que lo ejecute.

Entrada: El script se ejecutará:

Ejemplo1: C:/material/ArchivosGrandes2.ps1 100MB


Ejemplo2: C:/material/ArchivosGrandes2.ps1 1GB

Está esperando un parámetro. Ese parámetro es obligatorio.

Salida: Muestra aquellos archivos que sean mayores que el indicado.

Se exportará los resultados a un archivo html: ArchivosGrandes2.html

Pista: Pista: Where-object, \$HOME, ConvertTo-HTML,
 [Parameter(Mandatory=\$true)]\$variable

	Ciclo: ASIR	Departamento de Informática <small>www.informaticarodrigo.com.es</small> Informática Rodrigo Caro
	Módulo: Implantación de Sistemas Operativos	

3.- Cambiar las extensiones a archivos: CambiarExtensiones.ps1

Descripción: El script cambiará las extensiones de los archivos contenidos en una carpeta.

Entrada: El script preguntará:

Ruta que contiene los archivos.

Extensión de los archivos que quieres cambiar.

Extensión final.

Salida: Muestra el nombre de los archivos antes de ser cambiados y una vez cambiados.

Observaciones:


Suponemos que la ruta proporcionada es accesible por el usuario.

Pista:

1ª forma: Utilizando la opción -replace de Rename-Item, que nos permite reemplazar una cadena por otra.

2ª forma: la clase [IO.Path] de .NET, tiene un procedimiento llamado: ChangeExtension, que nos permite cambiar la extensión de un archivo.

Sintaxis: [io.path]::ChangeExtension(Nombre_archivo,"Extensión_nueva")

	Ciclo: ASIR	Departamento de Informática <small>www.informaticarodrigo.com.es</small> Informática Rodrigo Caro
	Módulo: Implantación de Sistemas Operativos	

4.- Cambiar las extensiones a archivos: CambiarExtensiones2.ps1

Descripción: El script cambiará las extensiones de los archivos contenidos en una carpeta

Entrada: El script preguntará:

Ruta que contiene los archivos.

Extensión final. Si no introducimos ningún valor, se les quitará la extensión a los archivos.

Salida: Muestra el nombre de los archivos antes de ser cambiados y una vez cambiados.


Observaciones:

Suponemos que la ruta proporcionada es accesible por el usuario.

Pista:

La clase [IO.Path] de .NET, tiene un procedimiento llamado: ChangeExtension, que nos permite cambiar la extensión de un archivo.

Sintaxis: [io.path]::ChangeExtension(Nombre_archivo,"Extensión_nueva")

	Ciclo: ASIR	Departamento de Informática <small>www.informaticarodrigo.com.es</small> Informática Rodrigo Caro
	Módulo: Implantación de Sistemas Operativos	

5.- Cambiar las extensiones a archivos: CambiarExtensiones3.ps1

Descripción: El script cambiará las extensiones de los archivos que no tienen extensión contenidos en una carpeta

Entrada: El script preguntará:

Ruta que contiene los archivos.

Extensión final. Si no introducimos ningún valor, se les quitará la extensión a los archivos.

Salida: Muestra el nombre de los archivos sin extensión antes de ser cambiados y una vez cambiados.

Observaciones:

Suponemos que la ruta proporcionada es accesible por el usuario.

Pista:

La clase [IO.Path] de .NET, tiene un procedimiento llamado: ChangeExtension, que nos permite cambiar la extensión de un archivo.

Sintaxis: [io.path]::ChangeExtension(Nombre_archivo,"Extensión_nueva")

Para mostrar los archivos sin extensión: ls -r -file \$ruta -exclude "*.*)"

6.- Realiza un script: ConectividadServidores.ps1

Descripción: El script nos mostrará si hay conectividad con los servidores.

Los datos de los servidores se encuentran almacenados en archivos **csv**.

planetas.csv	satelites.csv
<pre> nombre,ip jupiter,192.168.0.1 saturno,192.168.1.1 tierra,192.168.2.1 </pre>	<pre> nombre,ip sol,192.168.4.1 sirio,192.168.5.1 </pre>

Para realizar conectividad utilizaremos, Test-Connection:

Test-Connection IP -count 1 -quiet

-count 1 → Manda un ping

-quiet → Nos devuelve true o false

Ejemplo: Test-Connection 192.168.0.1 -count 1 -quiet

Nos devuelve true o false

Entrada:

El script recibirá como parámetro el nombre del archivo que contiene la información de los servidores.

Procedimiento:

Declara **una función llamada conectividad**, a la que se llamará para determinar si hay conectividad.


Salida:

Nos muestra en pantalla:

```

Resultados de conexión
192.168.0.1 -Conexión establecida
192.168.1.1 -ERROR de conexión
192.168.2.1 -ERROR de conexión

```

	Ciclo: ASIR	Departamento de Informática <small>www.informaticarodrigo.com.es</small> Informática Rodrigo Caro
	Módulo: Implantación de Sistemas Operativos	

7.- Realiza un script: ConectividadServidores2.ps1

Descripción:

Modifica el ejercicio anterior, para que el resultado no aparezca en pantalla, sino que nos cree un archivo de nombre: Resultados-FechaActual.txt.

Pistas:

Sustituir Write-Host por Write-Output

Formato fecha: Get-Date -format -d (Formato de fecha corta /dd/MM/AAA)

Concatenar: ="Resultados-" + \$(Get-Date -UFormat %d-%m-%y) + ".txt"
 -UFormat → Formato Unix

8.- Realiza un script: MenuConectividad.ps1

Descripción: El script nos mostrará el siguiente menú:

Conectividad:

- 1.- Comprobar conectividad con un equipo
- 2.- Ver equipos que tienen conectividad.
- 3.- Salir

Procedimiento:

Se debe diseñar las siguientes funciones:

menu → Para mostrarnos el menú.

comprobarConectividad → Realiza la opción 1 del menú.

equiposAccesibles → Realiza la opción 2 del menú.

ComprobarConectividad

Pregunta la IP que quieres comprobar y nos devuelve un mensaje: Sí hay conexión o No hay conexión.

Suponemos que la IP introducida es correcta.

EquiposAccesibles

Preguntará: La IP de la red (Vamos a suponer que es de clase C)

Rango inicial y rango final.

Nos mostrará en pantalla solo los equipos accesibles.

Ejemplo:

IP de la red: 192.168.2

Rango inicial: 10

Rango final:20

Irá comprobando las IP: 192.168.2.10, 192.168.2.11,,192.168.2.20