



IMPLANTACIÓN SISTEMAS OPERATIVOS

PRÁCTICA 4 PowerShell.

REALIZADO POR : JESÚS PADILLA CRESPO

Ejercicio 1

Si sabemos que el cmdlet "**Test-Path** *nombre_de_fichero*" nos devuelve True si existe un fichero o carpeta.

realizar un script de PowerShell que nos pida un nombre de un fichero para crearlo con la fecha del día como contenido. Debe comprobar si existe, en cuyo caso nos debe devolver un mensaje por pantalla indicando que no lo puede crear y lo vuelva a pedir de nuevo y en caso negativo crearlo.

```
do
{
    $fichero = Read-Host "Como quieres llamar al fichero"
    $a = Test-Path $fichero
    if ($a -eq "True")
    {
        Write-Host "El nombre seleccionado ya existe, elija
otro"
    }
    else
    {
        Get-Date > $fichero
        Write-Host "Se ha generado el archivo correctamente"
    }
}while ($a -eq "True")
```

```
Como quieres llamar al fichero: LBP
Se ha generado el archivo correctamente

PS C:\Users\lasir-27> do
{
    $fichero = Read-Host "Como quieres llamar al fichero"
    $a = Test-Path $fichero
    if ($a -eq "True")
    {
        Write-Host "El nombre seleccionado ya existe, elija otro"
    }
    else
    {
        Get-Date > $fichero
        Write-Host "Se ha generado el archivo correctamente"
    }
}while ($a -eq "True")

Como quieres llamar al fichero: LBP
El nombre seleccionado ya existe, elija otro
Como quieres llamar al fichero:
```

Ejercicio 2

Buscar el comando para crear usuarios en el sistema y crear un script que permita crear 5 usuarios:

```
$Prefijo = Read-Host "¿Qué prefijo quieres poner a los usuarios?"
if ((Get-LocalUser).name -eq $Prefijo)
{
    Write-Host "No se puede crearlos usuarios, ya existe uno con ese nombre"
}
else
{
    $contraseña = Read-Host "Indica la contraseña que quieras establecer"
    while (($contraseña).Length -lt 6)
    {
        Write-Host "Error, debe tener más de 6 caracteres"
        $contraseña = Read-Host "Indica la contraseña que quieras establecer"
    }
    for($i=1 ; $i -ne 6 ; $i++)
    {
        $contraseña = ConvertTo-SecureString $contraseña -AsPlainText -Force
        $Usuario = $Prefijo+'-'+$i
        New-LocalUser $Usuario -Password $contraseña
    }
}
```

```
PS C:\Windows\system32> $Prefijo = Read-Host "¿Qué prefijo quieres poner a los usuarios?"
if ((Get-LocalUser).name -eq $Prefijo)
{
    Write-Host "No se puede crearlos usuarios, ya existe uno con ese nombre"
}
else
{
    $contraseña = Read-Host "Indica la contraseña que quieras establecer"
    while (($contraseña).Length -lt 6)
    {
        Write-Host "Error, debe tener más de 6 caracteres"
        $contraseña = Read-Host "Indica la contraseña que quieras establecer"
    }
    for($i=1 ; $i -ne 6 ; $i++)
    {
        $contraseña = ConvertTo-SecureString $contraseña -AsPlainText -Force
        $Usuario = $Prefijo+'-'+$i
        New-LocalUser $Usuario -Password $contraseña
    }
}
¿Qué prefijo quieres poner a los usuarios?: ASIR21
Indica la contraseña que quieras establecer: 123456789

Name      Enabled Description
----      -
ASIR21-1  True
ASIR21-2  True
ASIR21-3  True
ASIR21-4  True
ASIR21-5  True
```

```
¿Qué prefijo quieres poner a los usuarios?: prefijo2
Indica la contraseña que quieras establecer: 12345
Error, debe tener más de 6 caracteres
Indica la contraseña que quieras establecer: |
```

Ejercicio 3

Realizar un script que analice si un ordenador (cuyo nombre nos indique el usuario) está conectado. Si está conectado debe mostrarnos un menú donde nos permita elegir entre saber cuál es su IP y saber cuáles son sus recursos compartidos (buscar en Internet el comando para hacer esto último).

```
$ordenador = Read-Host "Introduzca nombre o dirección del ordenador"
$comprobacion = Test-Connection $ordenador
if (($comprobacion).count -gt 0)
{
    Write-Host "El ordenador indicado está conectado"
    Write-Host -----
    Write-Host "Elija una opción"
    Write-Host -----
    Write-Host "1. Ver dirección IP del ordenador indicado"
    Write-Host "2. Listado de los recursos compartidos por el ordenador indicado"
    $op = Read-Host "Indique la opción"
    switch ($op)
    {
        "1" {Test-Connection $ordenador -Count 1 | select-object IPV4Address}
        "2" {Get-Wmiobject -classname win32_share -computername $ordenador}
    }
}
```

*He realizado pruebas con localhost y www.google.es para comprobar el funcionamiento.

```
Introduzca nombre o dirección del ordenador: www.google.es
El ordenador indicado está conectado
-----
Elija una opción
-----
1. Ver dirección IP del ordenador indicado
2. Listado de los recursos compartidos por el ordenador indicado
Indique la opción: 1

IPV4Address
-----
142.250.200.131
```

```
Indique la opción: 2

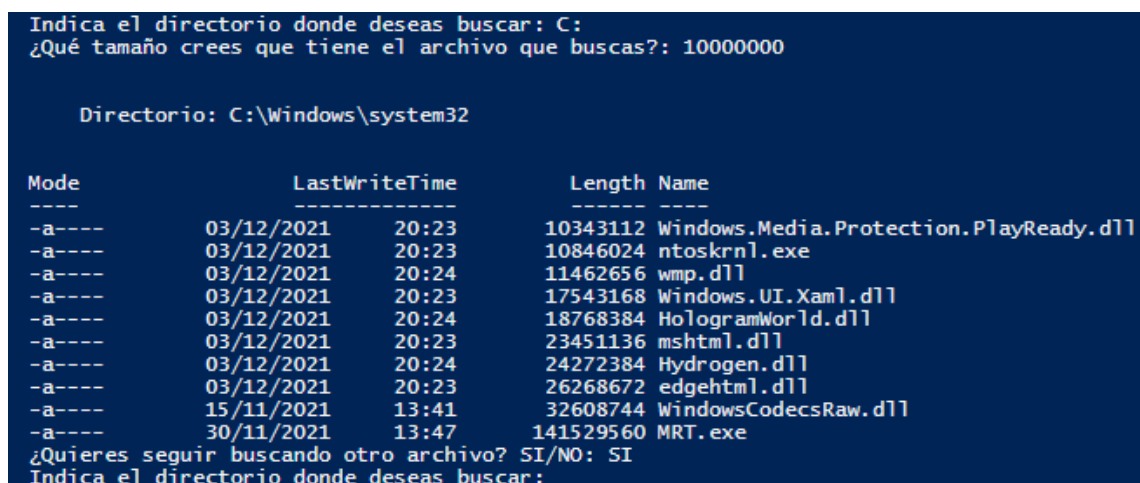
Name      Path      Description
----      -
ADMIN$    C:\Windows Admin remota
C$        C:\       Recurso predeterminado
IPC$      IPC$      IPC remota
```

Ejercicio 4

Realiza un script que nos pida el nombre de un directorio y un número correspondiente al tamaño de los ficheros que queremos buscar en él.

- Si el directorio no existe el programa debe sacar un mensaje de error.
- Si el directorio existe buscará dentro de él todos los ficheros cuyo tamaño sea mayor o igual que el indicado ordenados por tamaño de archivo.
- En cualquiera de los dos casos, después debe preguntarnos si deseamos volver a intentarlo y en caso afirmativo repetir la ejecución del script desde el principio.

```
do {
    $directorio = Read-Host "Indica el directorio donde deseas buscar"
    $comprobacion = Test-Path $directorio
    $tamaño = Read-Host "¿Qué tamaño crees que tiene el archivo que buscas?"
    if ($comprobacion -eq "True")
    {
        Get-ChildItem $directorio | Where-Object {$_.Length -ge $tamaño} | Sort-Object {$_.Length}
    }
    else
    {
        Write-Host "Error, lo sentimos, pero el directorio indicado no existe"
    }
    $repetir = Read-Host "¿Quieres seguir buscando otro archivo? SI/NO"
}
while ($repetir -eq "SI")
exit ($repetir -eq "NO")
```



```
Indica el directorio donde deseas buscar: C:
¿Qué tamaño crees que tiene el archivo que buscas?: 10000000

Directorio: C:\Windows\system32

Mode                LastWriteTime         Length Name
----                -
-a----            03/12/2021    20:23      10343112 Windows.Media.Protection.PlayReady.dll
-a----            03/12/2021    20:23      10846024 ntoskrnl.exe
-a----            03/12/2021    20:24      11462656 wmp.dll
-a----            03/12/2021    20:23      17543168 Windows.UI.Xaml.dll
-a----            03/12/2021    20:24      18768384 HologramWorld.dll
-a----            03/12/2021    20:23      23451136 mshtml.dll
-a----            03/12/2021    20:24      24272384 Hydrogen.dll
-a----            03/12/2021    20:23      26268672 edgehtml.dll
-a----           15/11/2021    13:41      32608744 WindowsCodecsRaw.dll
-a----           30/11/2021    13:47      141529560 MRT.exe
¿Quieres seguir buscando otro archivo? SI/NO: SI
Indica el directorio donde deseas buscar:
```

Ejercicio 5

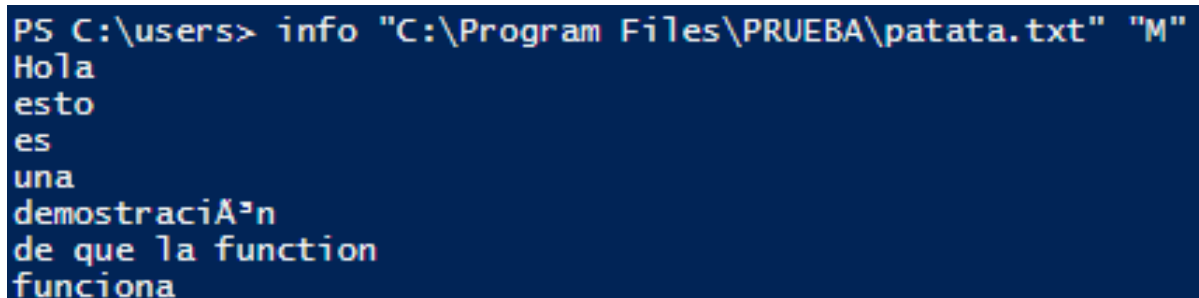
Realizar un script que utilice una función llamada "info" a la que le podamos pasar el nombre de un fichero y una operación. La función debe comprobar que exista el fichero y en caso afirmativo realizar la operación correspondiente: (B:eliminarlo, M:mostrar la información). En cualquier otro caso mostrar un mensaje por pantalla con el error correspondiente.

El código del script se puede probar con las siguientes llamadas:

info "pruebas.txt" "B" debería borrarlo si existe

info "pruebas.txt" "M" debería mostrar sus datos si existe

```
Function info ($ruta, $opcion)
{
    $comprobacion = (Test-Path $ruta)
    if ($comprobacion -eq "True")
    {
        switch ($opcion)
        {
            "B" {Remove-Item $ruta}
            "M" {Get-Content $ruta}
        }
    }
    else {Write-Host "ERROR"}
```



```
PS C:\users> info "C:\Program Files\PRUEBA\patata.txt" "M"
Hola
esto
es
una
demostraci3n
de que la function
funciona
```

Ejercicio 6

Realizar un menú de opciones que nos permita hacer lo siguiente en cada caso:

1. Buscar ficheros más grandes de un tamaño y ruta indicado por el usuario
2. Averiguar si un usuario indicado por teclado existe.
3. Mostrar la Memoria del sistema

```
do{
    Write-Host "Seleccione una de las siguientes opciones:"
    Write-Host "1. Buscar ficheros más grandes de un tamaño y ruta indicado por el usuario"
    Write-Host "2. Averiguar si un usuario indicado por teclado existe."
    Write-Host "3. Mostrar la Memoria del sistema"
    Write-Host "4. Salir"
    $menu = Read-Host "¿Que acción quieres realizar?"

switch($menu)
{
    "1" {
        Clear-Host
        Write-Host
        "*****"
        Write-Host "Buscar ficheros más grandes de un tamaño y ruta indicado por el usuario"
        Write-Host
        "*****"
        do {
            $directorio = Read-Host "Indica el directorio donde deseas buscar"
            $comprobacion = Test-Path $directorio
            $tamaño = Read-Host "¿Qué tamaño crees que tiene el archivo que buscas?"
            if ($comprobacion -eq "True")
            {
                Get-ChildItem $directorio | Where-Object {$_.Length -ge $tamaño} | Sort-Object
                {$_.Length}
            }
            else
            {
                Write-Host "Error, lo sentimos, pero el directorio indicado no existe"
            }
            $repetir = Read-Host "¿Quieres seguir buscando otro archivo? SI/NO"
        }
        while ($repetir -eq "SI")
    }
}
```

```

"2" {
    Clear-Host
    Write-Host
    "*****"

    Write-Host "2. Averiguar si un usuario indicado por teclado existe."
    Write-Host
    "*****"

    $comprobacion = Read-Host "Indique el nombre de usuario"
    $comprobacion = (Get-LocalUser -name $comprobacion -ErrorAction
SilentlyContinue).count
    if ($comprobacion -eq 0)
    {
        Write-Host
        "*****"

        Write-Host "El usuario indicado NO existe."
        Write-Host
        "*****"

        }
        else
        {
            Write-Host
            "*****"

            Write-Host "El usuario indicado SI existe."
            Write-Host
            "*****"

        }
    }
    "3" {
        Clear-Host
        Write-Host
        "*****"

        Write-Host "Mostrando memoria del sistema."
        Write-Host
        "*****"

        systeminfo |Select-String "Memoria"
    }
    "4" {
        Write-Host
        "*****"

        Write-Host "Cerrando aplicación...."
        Write-Host
        "*****"

        Exit
    }
    default {Write-Host "Error, debe seleccionar una opción"}
}
while ( $menu -ne 5)

```



```
PS C:\Windows\system32> C:\Users\asir27\Desktop\script5.ps1
Seleccione una de las siguientes opciones:
1. Buscar ficheros más grandes de un tamaño y ruta indicado por el usuario
2. Averiguar si un usuario indicado por teclado existe.
3. Mostrar la Memoria del sistema
4. Salir
¿Que acción quieres realizar?:
```

```
*****
Buscar ficheros más grandes de un tamaño y ruta indicado por el usuario
*****
Indica el directorio donde deseas buscar: C:
¿Qué tamaño crees que tiene el archivo que buscas?: 9999999
```

Directorio: C:\Windows\system32

Mode	LastWriteTime	Length	Name
-a----	03/12/2021 20:23	10343112	Windows.Media.Protection.PlayReady.dll
-a----	03/12/2021 20:23	10846024	ntoskrnl.exe
-a----	03/12/2021 20:24	11462656	wmp.dll
-a----	03/12/2021 20:23	17543168	Windows.UI.Xaml.dll
-a----	03/12/2021 20:24	18768384	HologramWorld.dll
-a----	03/12/2021 20:23	23451136	mshtml.dll
-a----	03/12/2021 20:24	24272384	Hydrogen.dll
-a----	03/12/2021 20:23	26268672	edgehtml.dll
-a----	15/11/2021 13:41	32608744	WindowsCodecsRaw.dll
-a----	30/11/2021 13:47	141529560	MRT.exe

¿Quieres seguir buscando otro archivo? SI/NO: |

```
*****
2. Averiguar si un usuario indicado por teclado existe.
*****
```

Indique el nombre de usuario: Administrador

El usuario indicado SI existe.

```
*****
Mostrando memoria del sistema.
*****
```

Cantidad total de memoria física:	7.168 MB
Memoria física disponible:	4.545 MB
Memoria virtual: tamaño máximo:	8.320 MB
Memoria virtual: disponible:	5.252 MB
Memoria virtual: en uso:	3.068 MB

¿Que acción quieres realizar?: 4

Cerrando aplicación....

PS C:\Windows\system32>