

#1 OPERACIONES BÁSICAS: SUMA, RESTA , PRODUCTO Y DIVISIÓN

```
$a=34
$b=25
$suma= $a+$b
$resta= $a-$b
$producto= $a*$b
$division= $a/$b
write "$a + $b = $suma"
write "$a - $b = $resta"
write "$a * $b = $producto"
write "$a / $b = $division"
```

#2 INTRODUCIR DATOS

```
$nombre = Read-Host "Tu nombre: "
$password = Read-Host -assecurestring "Tu clave: "
#mostrar
write-host "Hola $nombre"
Clear-Host
$YearCalc = Read-Host "?cuando naciste?"
$Now = (Get-Date -uformat "%Y") - $YearCalc
$Maybe = $Now -1
Write-Host "tu tienes $Maybe o $Now años"
```

#3 FUNCIÓN

```
function mensaje{
    "hola desde una función"
}
mensaje

function ver{
    "estas en: "
    gl
}
ver

#función con parámetros
$g=9.81
Function altura{
    param($t)
    if($t -eq 0){
        return 0
    }else{
        return ($g*[System.Math]::Pow($t,2))/2
    }
}
$result=altura(6.309)
write-host "$result"
```

#4 ESTRUCTURAS DE CONTROL

```
Write-Host "uso de do{}while(cond)"
$i = 1
do {
    Write-Host $i
    $i++
}while ($i -le 5)

Write-Host "uso de while(cond){}"
$i = 1
while ($i -le 5) {
    Write-Host $i
    $i++
}

Write-Host "uso de do{}until(cond)"
$i = 1
do {Write-Host $i; $i++}
```

```

until ($i -gt 5)

$strResponse = "Quit"
do {$strResponse = Read-Host "Are you sure you want to quit application? (Y/N)"}
until ($strResponse -eq "Y")

```

```

Write-Host "uso de for"
for ($i=1; $i -le 5; $i++)
{Write-Host $i}

```

```

$ints = @( 1, 2, 3, 4, 5)
for ($i=0; $i -le $ints.Length - 1; $i++)
{Write-Host $ints[$i]}

```

```

Write-Host "uso de foreach"
$ints = @(1, 2, 3, 4, 5)
foreach ($i in $ints)
{Write-Host $i}

```

#5 VECTORES

```

$lista=200,250,300,350,400
write-host "lista [$lista]"
foreach($l in $lista){

    if($l -eq 250){
        "$l, valor encontrado"
    }else{
        $result=$l*2;
        write-host "$l    $l*2: $result"
    }
}

```

#CUENTA VOCALES

```

Clear-Host
$cadena=Read-Host "Introduce cadena"
Write-Host "introduciste " $cadena
$cont=0
$aux=""
for($i=0;$i -le $cadena.Length;$i++){
    $c=$cadena[$i]
    if($c -eq "a" -or $c -eq "e" -or $c -eq "i" -or $c -eq "o" -or $c -eq "u"){
        $cont++
        $aux+=", "+$c
    }
}
Write-Host "no. de vocales: "$cont" ,vocales obtenidas: "$aux

```

#CALCULAR VALOR FUTURO

```

clear-host
$valor=read-host "Valor: "
$tasa=read-host "Tasa:"
$periodo=read-host "Periodo:"
$resultado=0
#mostrar valores
write-host "valor : $valor"
write-host "tasa : $tasa"
write-host "periodo : $periodo"
""
""
"[Resultado]"
foreach ($n in 1..$periodo) {
$resultado=[math]::pow(1+$tasa/100,$n)
$resultado=$resultado*$valor
write-host "$resultado periodo --> $n"
}

```

#6 MENÚ

```
clear-host
write-host "#####"
write-host ""
write-host "          Menu          "
write-host ""
write-host "1. Ver version"
write-host "2. Ver fecha"
write-host "3. Ver ayuda"
write-host "4. Abrir bloc de notas"
write-host "5. Abrir calculadora"
write-host "6. Salir"
write-host "#####"
$opc = Read-Host "Tu opcion: "
write-host ""
write-host "introduciste [$opc]"
#if(opc != 0 || opc >= 6)
if($opc -ne 0 -or $opc -ge 6){
    switch($opc){
        1 {write-host "version" -ForegroundColor Cyan
            get-host
        }
        2 {write-host "fecha" -ForegroundColor Cyan
            get-date
        }
        3 {write-host "ayuda" -ForegroundColor Cyan
            get-help
        }
        4 {write-host "bloc de notas" -ForegroundColor Cyan
            abreBloc
        }
        5 {write-host "calculadora" -ForegroundColor Cyan
            abreCalc
        }
        6 {write-host "fin" -ForegroundColor Red
            exit
        }
    }#fin switch
}
```

#7 DATOS DEL SISTEMA

```
write-host "Datos del sistema:"
New-Object System.io.DriveInfo "C:" | Format-List *
$drive = New-Object System.io.DriveInfo "C:"
$drive.DriveFormat
$drive.VolumeLabel
```

```
#UBICACIÓN
$variable=gl
"estas en: "
$variable
```

```
#FECHA
$fecha=Get-Date
write-host "hoy es "$fecha
```

```
#IMPRESORAS
write-host
write-host "ShowPrnH.ps1, Version 1.01"
write-host "Show available printers in HTML"
write-host "Written by Rob van der Woude"
write-host "http://www.robvanderwoude.com"
write-host
```

```
get-wmiobject -class Win32_Printer | convertto-html
Name,Default,Network,PortName,DriverName,ServerName,ShareName -head "<title>All printers
available on $env:computername</title>`n<style type=`"text/css`">`nbody { padding: 8px;
line-height: 1.33 }`ntable { border-style: ridge }`ntd, th { padding: 10px; border-style:
```

```

dotted; border-width: 1px }`nth { font-weight: bolder; text-align: center }`n</style>" |
out-file -FilePath "showprnh.html" -Encoding "ASCII"
    invoke-item "showprnh.html"

#ARCHIVOS y guardarla en un archivo de texto *.txt

gci > listaArchivos.txt

#mostrar el nombre de los ARCHIVOS y el tamaño ordenados por tamaño
gci |select name,length |sort length -desc

#Cómo mostrar el nombre de los ARCHIVOS y el tamaño cuyo tamaño sea 79 bytes en
powershell ?

gci |select name,length | where {$_.length -eq 76}

#8 ARCHIVOS txt
Clear-Host
$Path = "C:\Program Files\"
Get-Childitem $Path -recurse -force | Foreach {
    If ($_.extension -eq ".txt") {
        Write-Host $_.fullname
    }
}

$strResponse = "salir"
do {$strResponse = Read-Host "Quiere salir de la aplicación? (Y/N)"}
until ($strResponse -eq "Y")

New-Item -Type f freespace.txt

$date = ( get-
date ).ToString('yyyyMMdd')
$file = New-Item -type file "$date-
freespace.txt"
$date = ( get-date ).ToString('yyyyMMdd')

Foreach ($system in Get-Content "servicio.txt")
{Write-Host
$system}

#ATRIBUTOS DE ARCHIVOS
$Path = "C:\Program Files\"
"{0,10} {1,-24} {2,-2}" -f `
" Size", "Last Accessed", "File Name "
Foreach ($file in Get-Childitem $Path -recurse -force)
{If ($file.extension -eq ".txt")
{
    "{0,10} {1,-24} {2,-2}" -f `
    $file.length, $file.LastAccessTime, $file.fullname
}
}

# CONTENIDO DE ARCHIVOS
clear
Get-Content C:\servicio.txt

$a = Get-Content C:\Users\fernando\Documents\Ejemplos\servicio.txt
write-host "$a"

(Get-Content C:\Users\fernando\Documents\Ejemplos\servicio.txt)[0 .. 2]

$sarch=get-content C:\Users\fernando\Documents\Ejemplos\servicio.txt
Foreach-Object {Write-Host $sarch -foregroundcolor red}

foreach ($number in 1..10 ) { $number * 4}

```

#9 ARCHIVOS Y MENÚ Y FUNCIONES

```
clear
get-host
$cadena="pelo"
$aarchivo="C:\Users\fernando\Documents\Ejemplos\servicio.txt"
$fecha=(get-date)
$fecha
$nombre = Read-Host "Tu nombre es: "
$password = Read-Host -assecurestring "Tu password: "
write-host "Bienvenido $nombre"
"Te encuentras en:"
gl
"usuario:"
whoami
$cad=$cadena-replace("l","rr")
$cad
"contenido del archivo:"
get-content $aarchivo
$compara=whoami
if($nombre -eq $compara){"iguales"}else{"no iguales"}
function abre{
    param($nombre)
    if($nombre -eq "fernando"){start notepad}else{"start calc, nada igual"}
}

#invocar a la función
abre($nombre)

$numero= read-host "número: "
switch($numero){
    1 {" abre"}
    2 {" cierra"}
    3 {" apaga"}
    default {"inactivo"}
}

try{
    "abriendo archivo, se ejecuto esto"
    abre($nombre)
}catch{
    "no se encuentra el archivo"
}
```

#10 PROCESOS

```
$ubicacion=gl
Write-Host "te encuentras en:" $ubicacion
Get-Help -Name Get-Process

#PROCESOS Y SERVICIOS
#####3
#Get-Process | ForEach-Object {Write-Host $_.name -foregroundcolor cyan}
#write-host "Algo"
#####
#$a = (get-date).day
#$a = (get-date).dayofweek
#$a = (get-date).dayofyear
#$a = (get-date).hour
#$a = (get-date).millisecond
#$a = (get-date).minute
#$a = (get-date).month
#$a = (get-date).second
#$a = (get-date).timeofday
#$a = (get-date).year
#get-date -DisplayHint date
#$now=Get-Date -format "dd-MMM-yyyy HH:mm"
#get-date -format g
#(get-date).dayofyear
#$a = get-wmiobject win32_bios -computer SERVER64
```

```

#$a | format-list -property Name, @{Label="BIOS Date "; `
#Expression={$_.ConvertToDateTime($_.ReleaseDate)}}
#####
#dir <enter>
#ls <enter>
#gci <enter>
Get-ChildItem <enter>
#asignar un alias
Set-Alias gs Get-Service <enter>
#exportar contenido a un txt
Export-Alias -Path Aliases.txt <enter>

#INICIAR PROCESO CON FUNCIONES
Function abreBloc{
    start notepad
}

Function abreCalc{
    start calc
}

#EMULAR EL COMANDO TOP DE LINUX
while (1) { ps | sort -desc cpu | select -first 30; sleep -seconds 2; cls }

#ENVIAR CORREO
$filename = "c:\scripts_scott\test9999.xls"
$smtpserver = "smtp.gmail.com"
$msg = new-object Net.Mail.MailMessage
$att = new-object Net.Mail.Attachment($filename)
$smtp = new-object Net.Mail.SmtpClient($smtpServer )
$smtp.EnableSsl = $True
$smtp.Credentials = New-Object System.Net.NetworkCredential("username",
"password here"); # Put username without the @GMAIL.com or - @gmail.com
$msg.From = "username@gmail.com"
$msg.To.Add("boss@job.com")
$msg.Subject = "Monthly Report"
$msg.Body = "Good Morning, Last month's LOGINS & GROUPCALLS FOR ALL GIDS IN SYSTEM IS
ATTACHED"
$msg.Attachments.Add($att)
$smtp.Send($msg)

```