|  |
| --- |
|  |
| POWERSHELL:  SCRIPTS |
|  |

IMPLANTACIÓN DE SISTEMAS

JUAN CARLOS NAVIDAD GARCÍA

1. **¿Qué es un script y cuál es su función principal?**

Los scripts son fragmentos de código que tienen como objetivo realizar o añadir funciones a un sistema.

Su función principal es automatizar tareas.

1. **¿Qué significa ISE?**

**ISE 🡪 Integrated Scripting Environment**. Básicamente es el entorno de desarrollo de scripts de Powershell.

1. **Diferencias entre PowerShell y PowerShell ISE**

Powershell a secas es la consola remodelada MS-DOS de Windows, simplemente se encarga de ejecutar funciones o devolver información, PowerShell ISE es el entorno de desarrollo de scripts, este desarrolla los scripts que después pueden ser ejecutados por PowerShell.

1. **¿Cuál puede ser la causa de no poder ejecutar un script en PowerShell?**

Windows por defecto y seguridad inhabilita la ejecución de scripts con un nivel de seguridad restricted.

1. **Comprobar los niveles de seguridad**

Para comprobar el nivel de seguridad de nuestro equipo ejecutaremos el comando **🡪 get-ExecutionPolicy**, este nos puede devolver las siguientes opciones:

1. **Restricted**: no permite ejecutar scripts.
2. **Allsigned**: todos los scripts tienen que estar autenticados.
3. **Unrestricted**: permite ejecutar cualquier script.
4. **RemoteSigned**: permite ejecutar nuestros scripts, pero no los remotos.
5. **¿Cómo se cambian los permisos?**

El nivel de seguridad se cambia con el comando **Set-ExecutionPolicy** <<una de las opciones del ejercicio anterior>>.

1. **¿Cómo creamos los scripts en PowerShell?**

Primero hay que crear un profile o un archivo en formato .ps1, dentro de este archivo, introduciremos el código que nosotros queramos realizar, posteriormente lo abrimos con PowerShell y se ejecutará:

Para crear un profile usaremos el siguiente comando:

New-Item -Path $profile -Type File -Force

Para comprobar que se ha creado:

test-path $profile

Para editarlo tendremos varias opciones, desde PowerShell podemos escribir “ise $profile” y lo podremos editar desde PowerShell ISE, aunque también, este archivo .ps1 de profile se nos crea por defecto en nuestra carpeta documentos y lo podremos editar desde cualquier editor de texto o incluso de programación como VisualStudio Code, Notepad ++, etcc.

1. **Variables:**

a. Definición teórica y sintáctica: las variables son espacios reservados en la memoria que, como su nombre indica, pueden cambiar de contenido a lo largo de la ejecución de un programa. Una variable corresponde a un área reservada en la memoria principal del ordenador.

b. Acceso a la información: para acceder a la información de una variable, basta con lanzar la variable como $variable, nos saldrá por pantalla su información.

c. Asignación de información: la información se le asigna a una variable en el siguiente formato: $variable= x.

d. Declaración con un tipo de dato:

e. ¿Distingue mayúsculas?: PowerShell es keysensitive, quiere decir que no distingue entre mayúsculas ni minúsculas.

f. ¿Qué es el scope de una variable?: El ámbito de una variable (llamado «scope» en inglés) es la zona del programa en la que se define la variable.

g. Borrar el contenido de una variable y borrar variable

Clear-variable -name nombreVariable

Remove-variable -name nombreVariable

h. Tipos de variables con ejemplos

* Variables creadas por el usuario: $variable=2
* Variables predefinidas:
  + Automáticas: $$, $^, $null.
  + Personalización del comportamiento PS: $OutputEncoding.

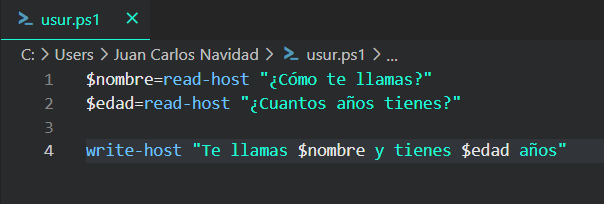
i. ¿En qué se parece y en que se diferencia de las constantes?: Son variables cuyo valor no cambia durante la ejecución de un programa. Únicamente puede ser leído. Las variables como tal, son modificables en comparación de las constantes.

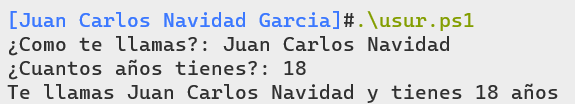
1. **¿Cuál es la función de los comentarios en un script? Que tipos de comentarios existen y con qué símbolo se representan. Pon un ejemplo de cada**

Los comentarios en los scripts de PowerShell son útiles para explicar los detalles de los procedimientos que sigue el script al ejecutarse.

Los comentarios se pueden dividir en comentarios de una línea (#) y varias líneas (<#Texto#>),

1. **Cmdlet para interactuar con el usuario, pedirle informacion y mostrarle informacion.**

****



1. **Realiza un script documentado(comentarios) para Crear dos variables:**

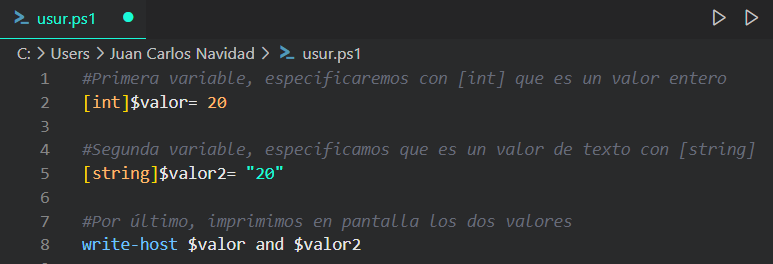
• Asignándole el valor 20, e indicando el tipo entero

• Asignándole el tipo carácter

Muestre el contenido de las variables

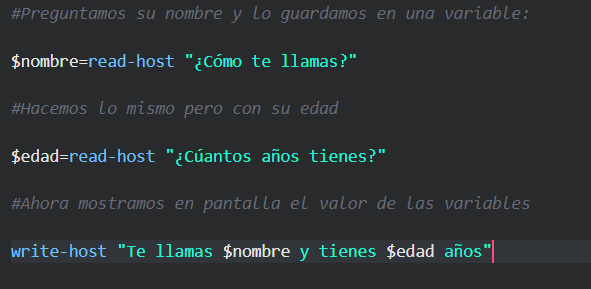
Muestra el tipo de datos de las variables

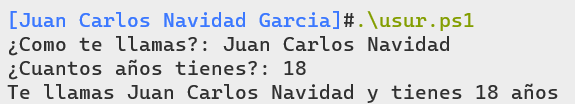
Muestra su valor





1. **Realiza un script documentado(comentarios) para pedirle al usuario su nombre y su edad mostrándole “tu nombre es … y tienes … años. Siendo los … el nombre y la edad del usuario.**

****

****

1. **Crear un script para saludar al usuario. Mostrara**
2. **El Saludo: se proporciona con argumentos $args[0] y puede ser :**

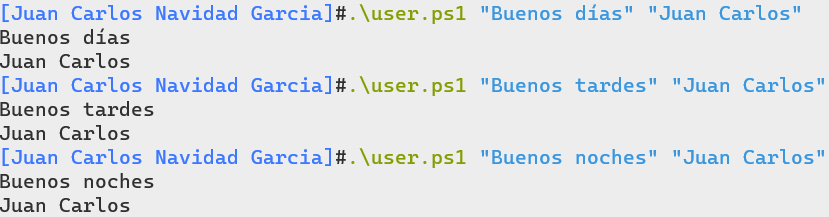
i. Buenos días

ii. Buenas tarde

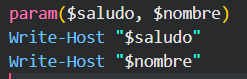
iii. Buenas noches

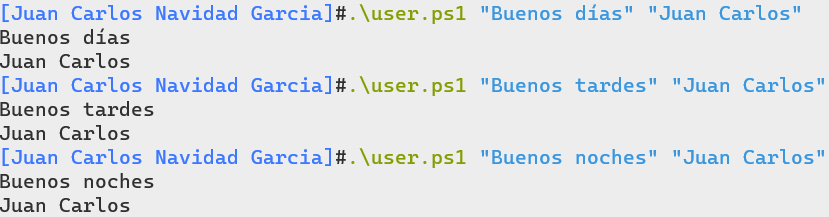
1. **El nombreUsuario: se proporciona con argumentos $args[1]**



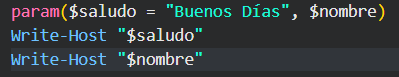


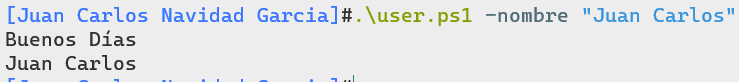
1. **Idem que el anterior pero con parámetros Param()**

****

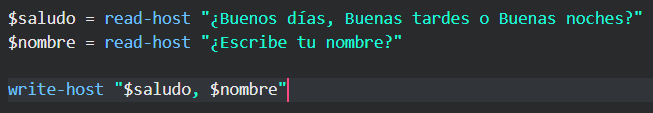


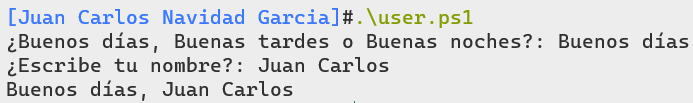
1. **Modifica el anterior para que, si no se introduce saludo, se muestre Buenos Días**

****

****

1. **Idem que el anterior, pero pidiendo los datos al usuario en tiempo de ejecución**

****

****

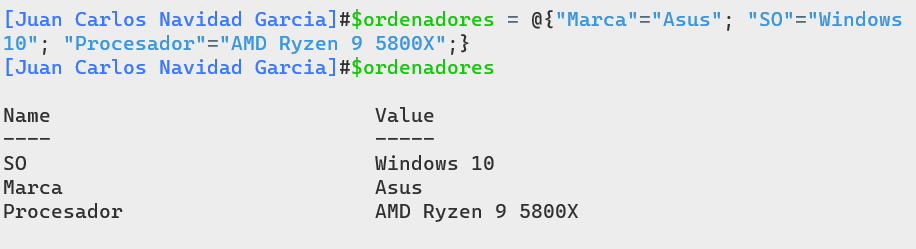
1. **Crea un diccionario llamado ordenador que guarde algunas características de los ordenadores:**

a. Marca

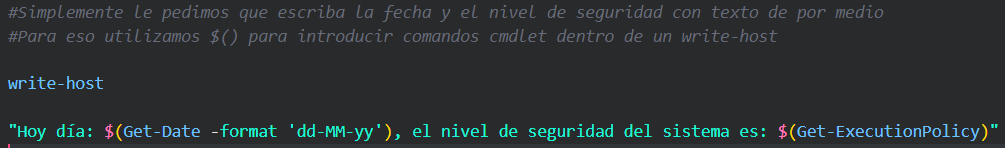
b. So

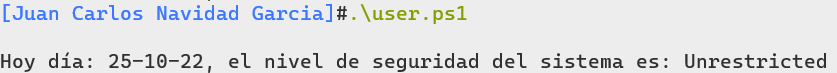
c. Procesador

Después mostrara las marcas, so y el procesador de los ordenadores que tenga. Usa variable.elemento ($ordenador.marca)

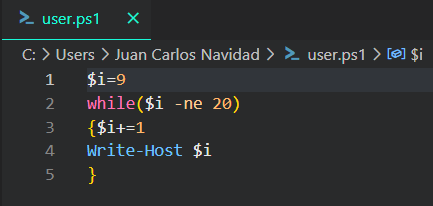


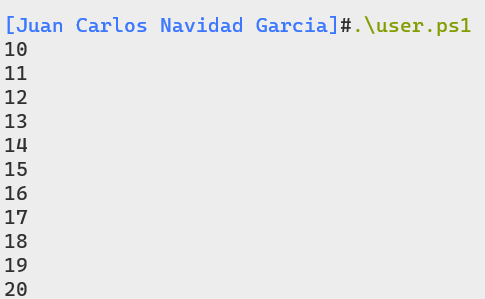
1. **Realiza un script documentado(comentarios) para mostrar el nivel de seguridad que tiene el sistema a fecha de hoy, es decir, tendría que mostrar por pantalla: Hoy día “fecha” el nivel de seguridad del sistema es NOTA: Recordad el uso de: $()**

****

****

1. **Crea un script que muestre los números del 10 al 20**

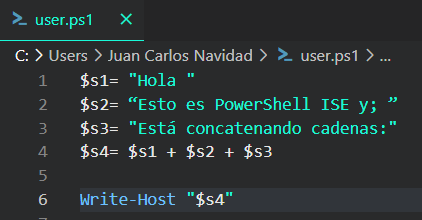
****

****

1. **¿Cuál es el operador de concatenación?**

**+**

1. **En un script crea las siguientes variables y concaténalas en una sola llamada S4.**



$ s1= "Hola"

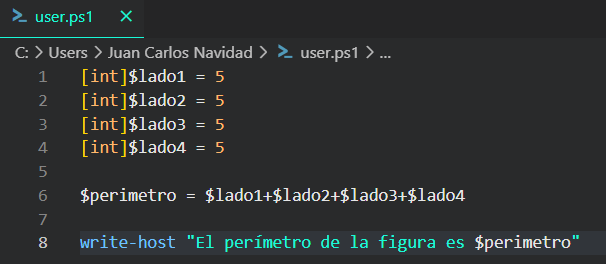
$ s2= “Esto es PowerShell ISE y; ”

$ s3= "Está concatenando cadenas:"

Verifica que se ha realizado bien la concatenación S4



1. **Pedir al usuario el lado de un cuadrado y calcula su perímetro. Almacena los datos en una estructura que te permita guardar la siguiente información:**

Lado1 = 5

Lado2 = 5

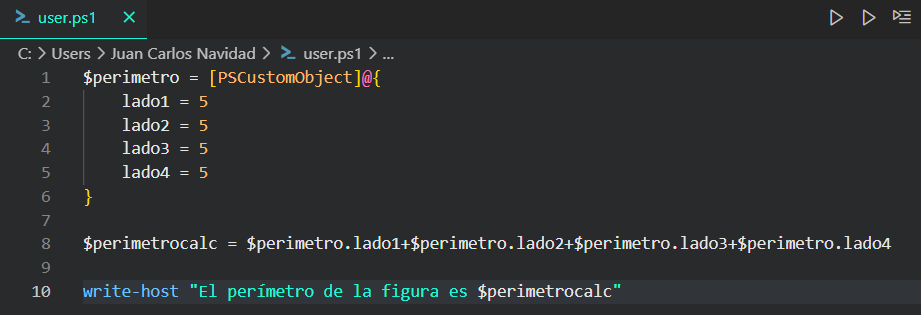
Lado3 = 5

Lado4 = 5

Perímetro = Resultado

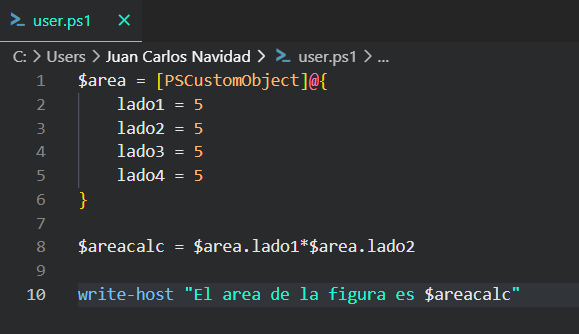


1. **Modifica el script anterior para que muestre los datos de forma ordenada, para ello utiliza [PScustomObjet]**

****

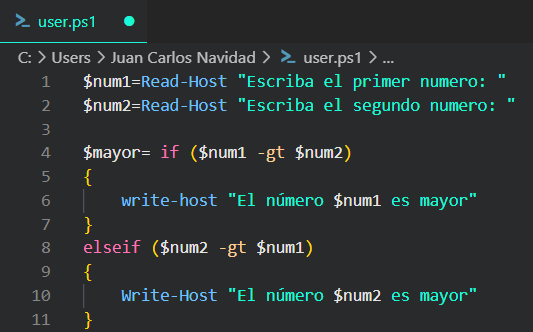


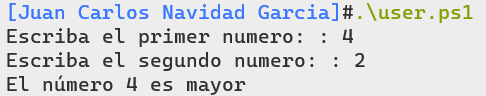
1. **Idem que el anterior, pero calculando el área de un rectángulo**

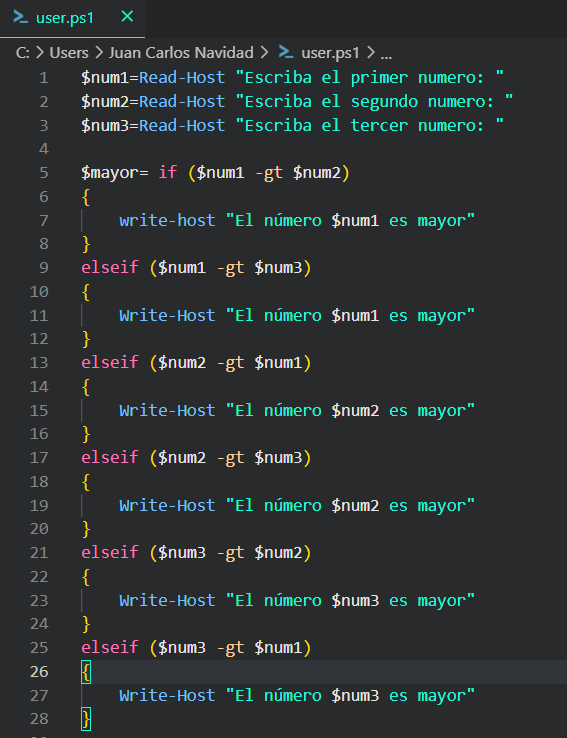
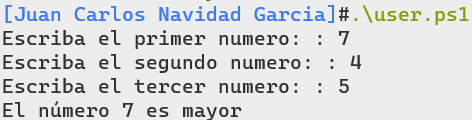
****

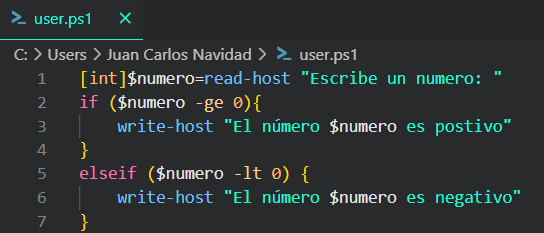
****

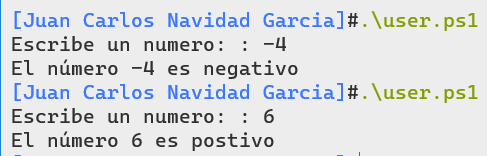
1. **Script que pida dos números y diga cual es el mayor**

****

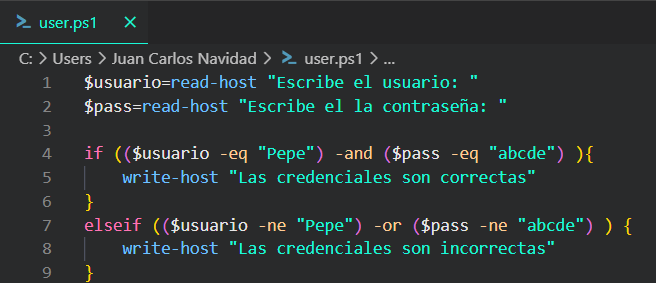
****

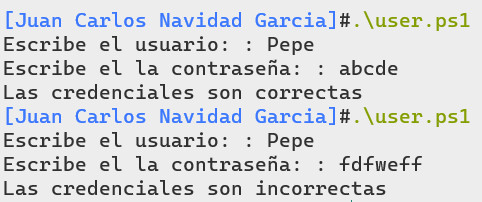
1. **Script que pida 3 números y diga cual es mayor**
2. **Script que pida un numero y diga si es Negativo o positivo.**

****

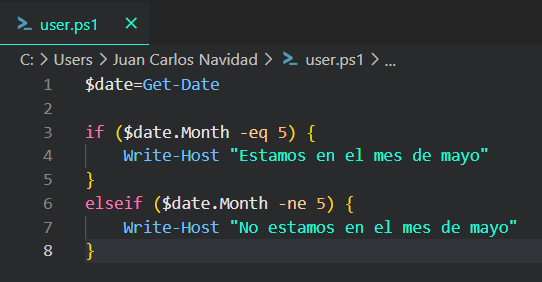
****

1. **Script para pedir usuario y contraseña. Si coincide con “Pepe” y “abcde” indique has entrado en el sistema sino mostrará usuario no reconocido.**

****

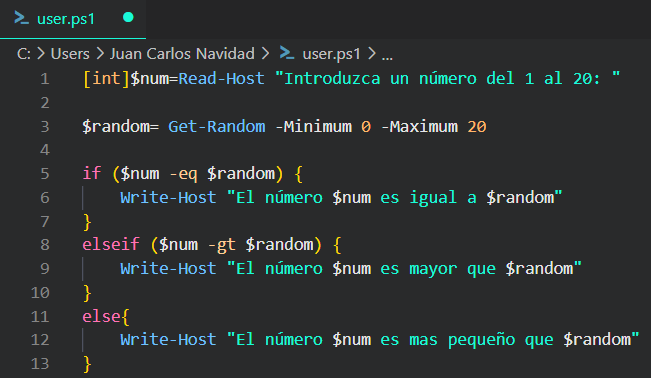
****

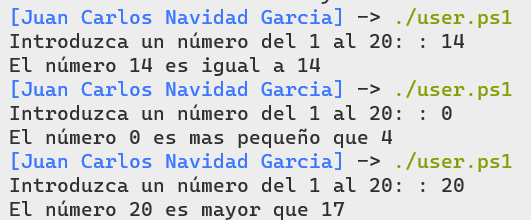
1. **Script que compruebe si estamos en el mes de mayo. Informando Estamos en el mes de mayo o No estamos en el mes de mayo.**

****

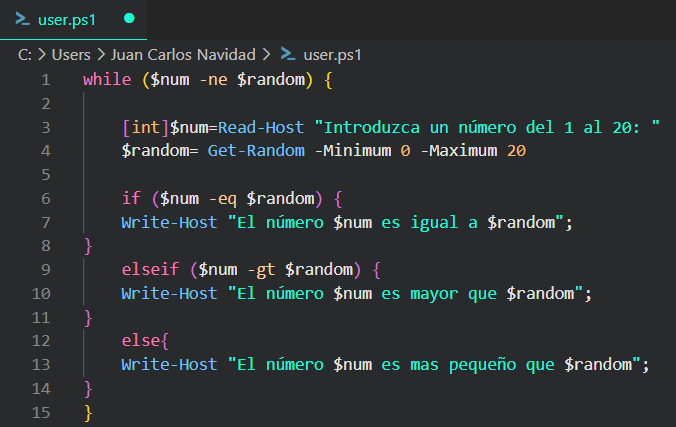
****

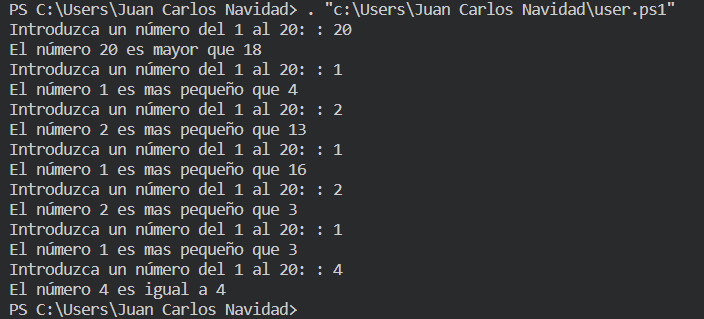
1. **Realizar un script que escoja un número aleatorio entre 1 y 20, pregunte al usuario y le diga si es más pequeño o más grande.**

****

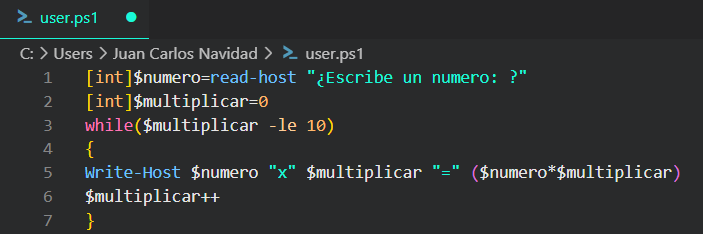
****

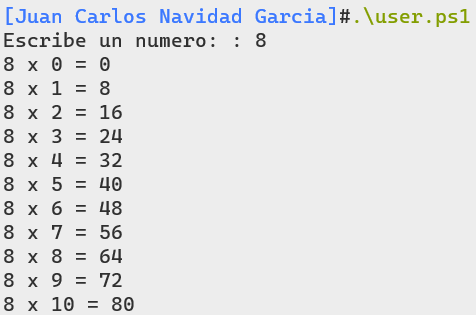
1. **Realizar un script que escoja un número aleatorio entre 1y 20, pregunte al usuario y le diga si es más pequeño o más grande, que continué hasta que acierte. Haz que muestre el número de intentos hasta que lo acierta cuando finalice. Utiliza Get-Randorm(rango).**

****

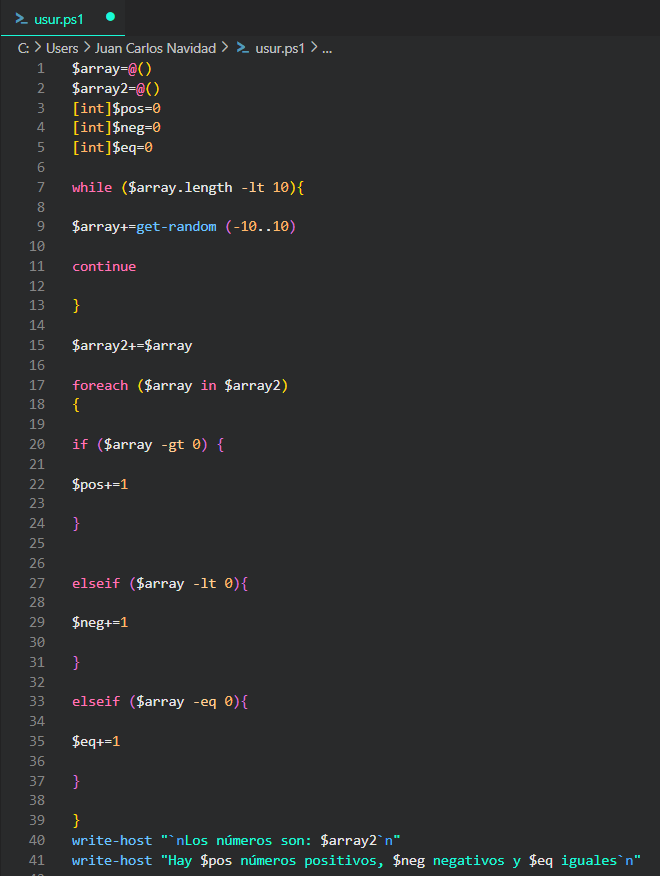
****

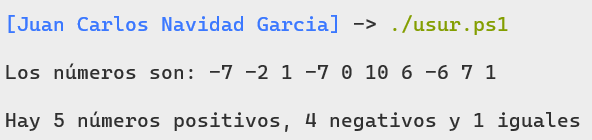
1. **Realizar un script que pide un número al usuario y muestre su tabla de multiplicar tabla\_multiplicar.ps1**

****

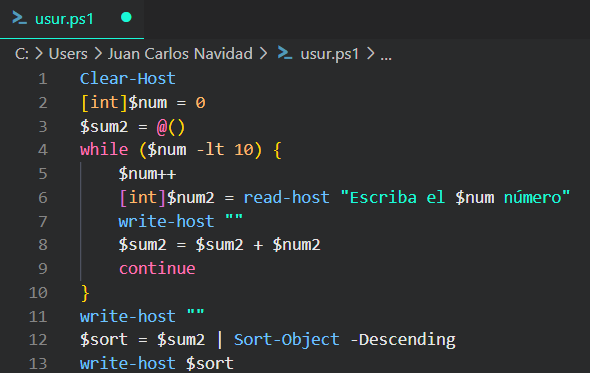
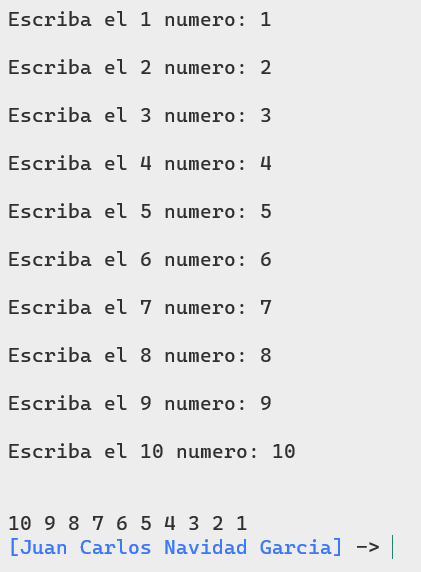
****

1. **Crea un script con:**
   1. un array de 10 números inicializados a 0 (todos los numeros del array son 0.
   2. Inicializalos con numeros aleatorios entre 10 y -10. Utiliza el cmdlet Get-Random para conseguir esos números. Usar lenght y bucle
   3. Contabiliza cuántos números positivos, negativos e iguales a 0 hay en ese array. Usa bucle e If

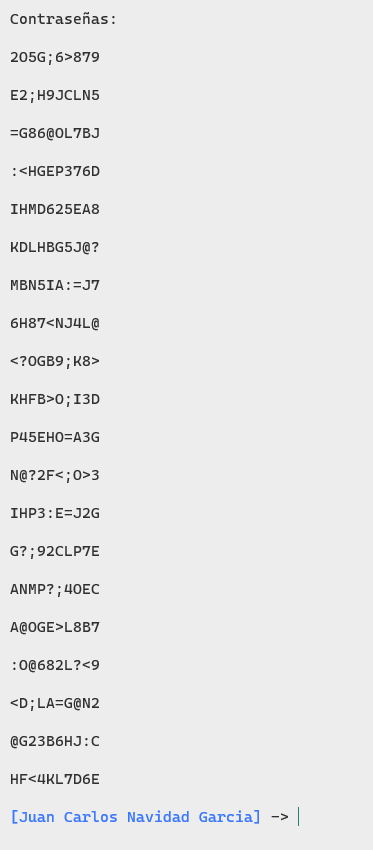
****

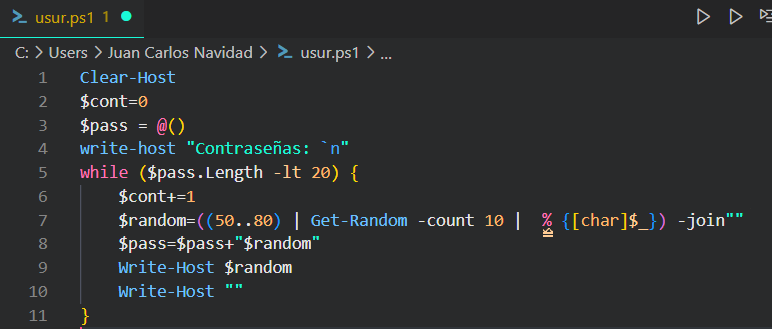
****

1. **Script que pida 10 números y los muestre en orden descendente**

****

1. **Utilizando arrays (para guardar los caracteres que pueden usar la contraseña) y números aleatorios, desarrolla un script de powershell que genere contraseñas. Max longitud 20.**

****

****

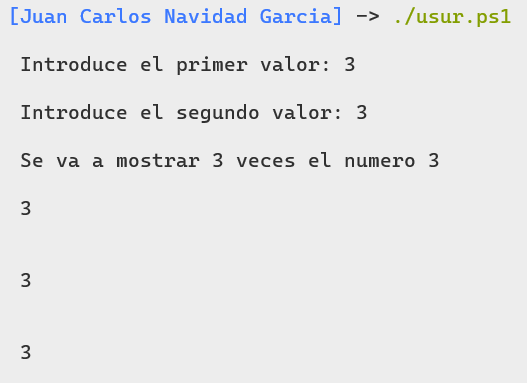
1. **Crea un cmdlet de powershell que espere tres parámetros, un llamado $primero que espere un [int], otro llamado $segundo que espere un String que si no se introduce debe solicitarlo al usuario. Una vez hecho el programa debe hacer un bucle for que muestre el valor de $segundo tantas veces como indique $primero. Recuerda los parámetros se definen**

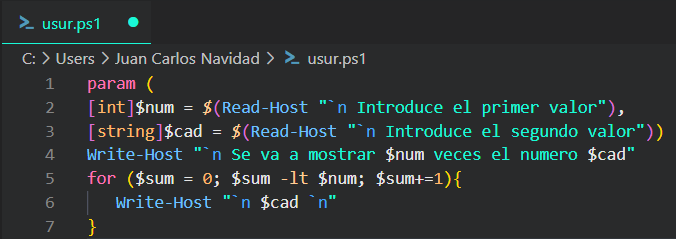
Param(

[tipo]Parametro1,

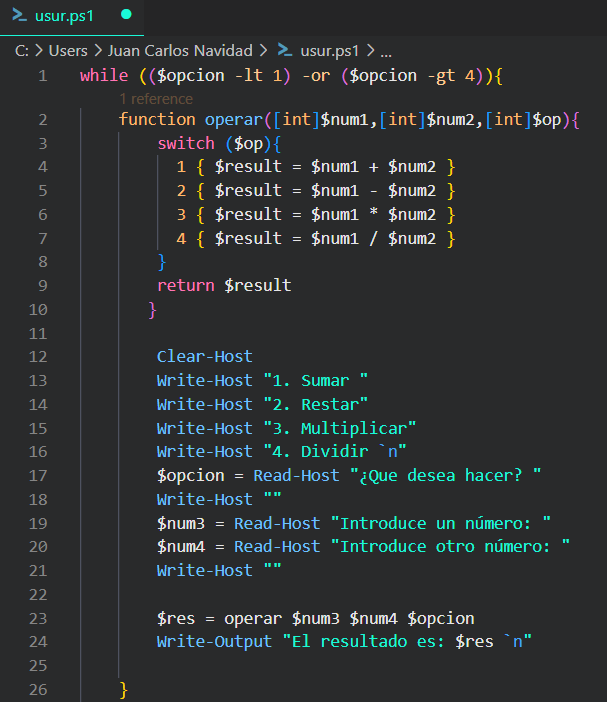
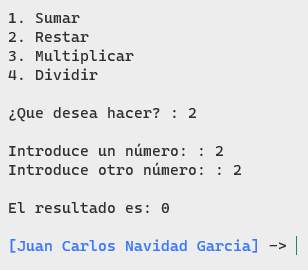
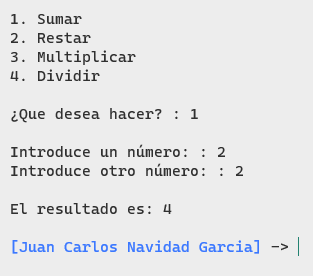
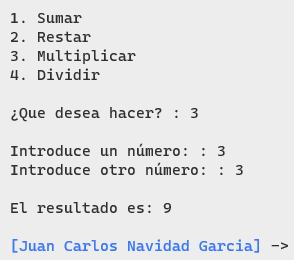
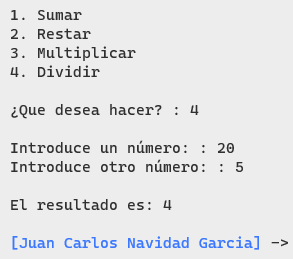
[tipo]Parametro2.

….

)



1. **Crear una calculadora muy básica con PowerShell la cual solo tendrá cuatro posibles operaciones: suma, resta, multiplicación y división. Usa una función llamada "operar" a la que se pasaremos dos números y una operación. Usa switch para realizar operación dependiendo de la operación pasada.**

****

1. **Crea un cmdlet de PowerShell que defina un array relacional con usuarios. Los campos de cada elemento deben ser nombre\_cuenta, nombre y password. El programa debe mantener ese array con un menú en el que se pueda:**

-añadir

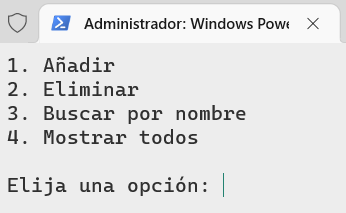
-eliminar

-buscar por nombre

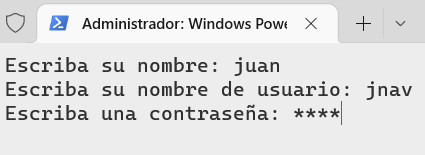
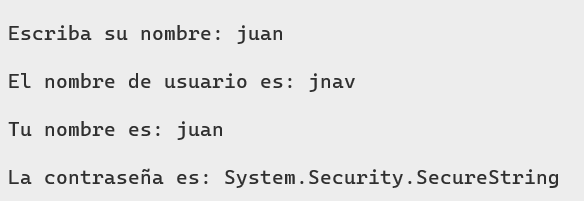
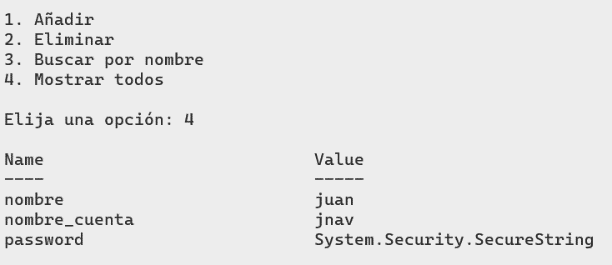
-mostrar todos

Al crear uno nuevo deben solicitarse datos y el campo password debe ocultarse al escribir.





Menú



Opción 4

Opción 3

Opción 1

1. **Realizar un script que muestre 5 números aleatorios del 1 al 10 seguidos de el mismo número de asteriscos.**

