

```
1 #!/bin/bash
2 cd /home/spect
3 mv temp/* archive/
4 echo "Temporary files stored in the archive" >> log.txt
5 name="spect"
6 age=21
7 echo "Hello my name is $name and I am $age years old"
8 user=$(whoami)
9 echo "The script is run by the user : $user"
10 FirstNumber=1
11 SecondNumber=2
12 echo "${FirstNumber}$SecondNumber";
13 FirstNumber=1
14 SecondNumber=2
15 echo $((FirstNumber+SecondNumber))
16 Number=1
17 ((Number += 5)) # 1 + 5 = 6
18 ((Number /= 2)) # 6 / 2 = 3
19 SecondNumber=2
20 Result=$((Number+SecondNumber))
21 echo $Result
22 age= 38;
23 threshold= 18;
24 if [ $age -lt $threshold ]; then
25 echo "You are still a minor."
26 else
27 echo "You are of legal age."
28 fi
29 for i in {1..10..2}
30 do
```

Bash Scripting

Básico- Avanzado



¿Que es Bash Scripting?

Bash Scripting es una forma de automatizar tareas en sistemas operativos basados en Unix, como Linux y macOS. Es como crear un “guión” para que tu computadora realice una serie de acciones de forma automática.

- Como Funciona:

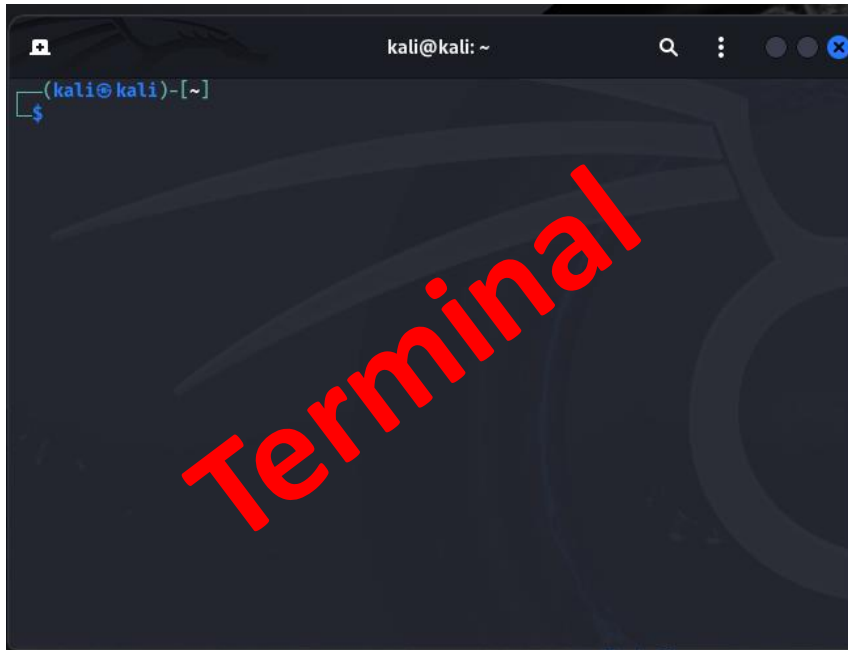
- **Bash:** Es un intérprete de comandos, como un traductor entre tú y tu computadora.
- **Script:** Es un archivo de texto que contiene una secuencia de comandos Bash, es decir; las instrucciones que quieres que tu computadora ejecute.

- Para que Sirve aprender Bash: Es fundamental si trabajas con entornos Linux o sistemas basados en Unix. Aquí te presentamos algunas de las principales razones por las que dominar bash.

- Automatizar tareas repetitivas
- Crear programas Sencillos
- Mejorar la eficiencia

Terminal

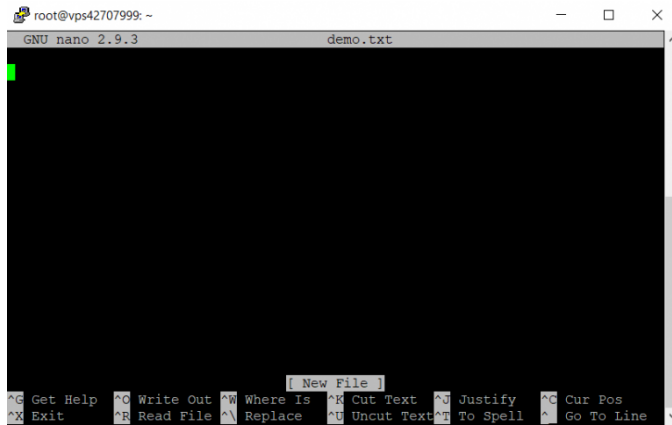
En la terminal es donde ejecutaras tus comandos y scripts. Puedes acceder a la terminal en la mayoría de los sistemas Linux presionando **ctrl+ALT+T** o buscando <<Terminal>> en el menú de tu sistema. Es aquí donde escribiras y probaras tus comandos Bash.



Esto es la Terminal donde haremos nuestros scripts de bash donde utilizaremos cualquier editor de texto como puede ser:

- **Nano:** Es un editor de texto Simple y fácil de usar.
- **Vim:** Es un editor más avanzado.
- **Gedit:** Es un editor gráfico que se asemeja a los editores de textos comunes como notepad.

Tipos de Editores de Texto en Linux



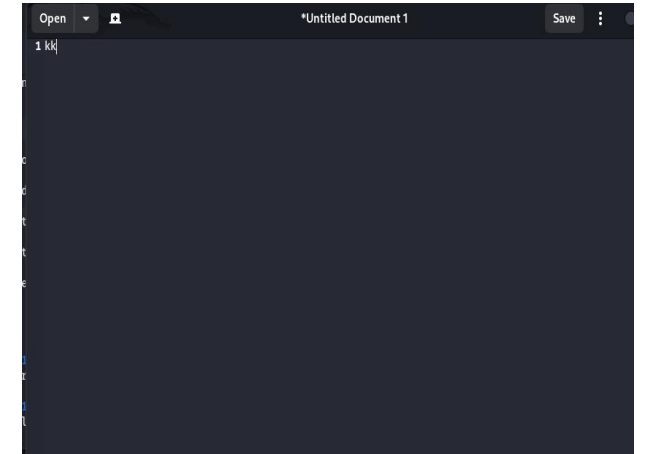
The screenshot shows the GNU nano 2.9.3 text editor. The title bar indicates the file is 'demo.txt'. The editor has a simple interface with a black background and white text. At the bottom, there is a menu bar with various options: Get Help, Write Out, Where Is, Cut Text, Justify, Cur Pos, Exit, Read File, Replace, Uncut Text, To Spell, and Go To Line. A 'New File' button is also visible in the center of the bottom bar.

Nano



The screenshot shows the VIM text editor. The interface is split into two panes. The left pane shows a list of files in a directory. The right pane shows the contents of a file, with some lines highlighted in yellow. The VIM status bar at the bottom indicates the current file and line number.

VIM

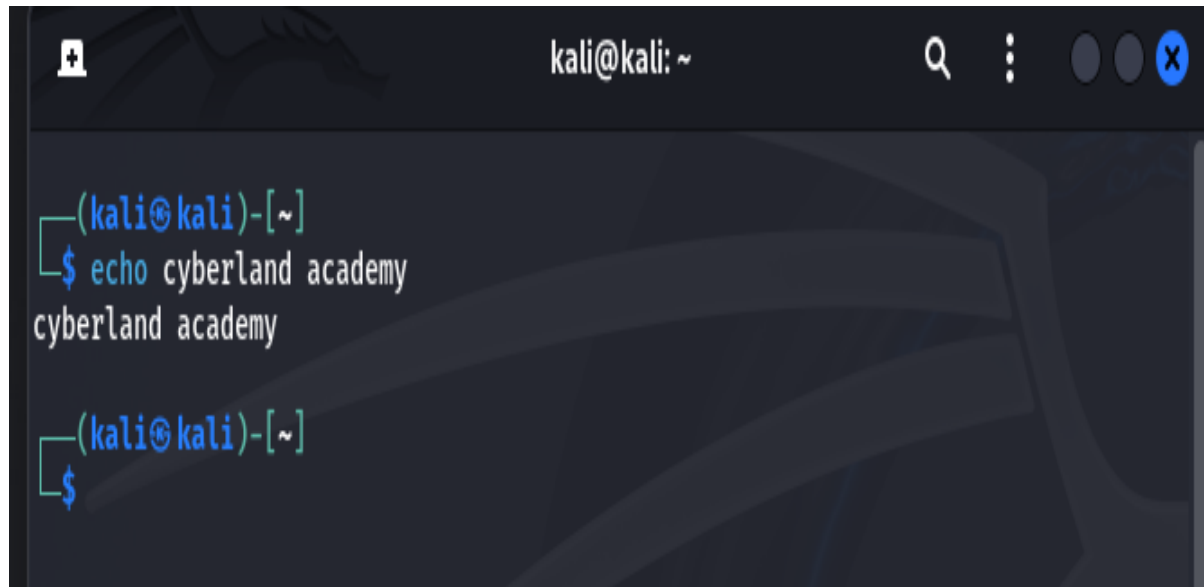


The screenshot shows the GEDIT text editor. The title bar indicates the file is 'Untitled Document 1'. The editor has a dark background and a light-colored text area. The status bar at the bottom shows the current line and column number.

GEDIT

Tu Primer Comando Bash

Abre la terminal en tu sistema Linux e intenta ejecutar el siguiente comando:

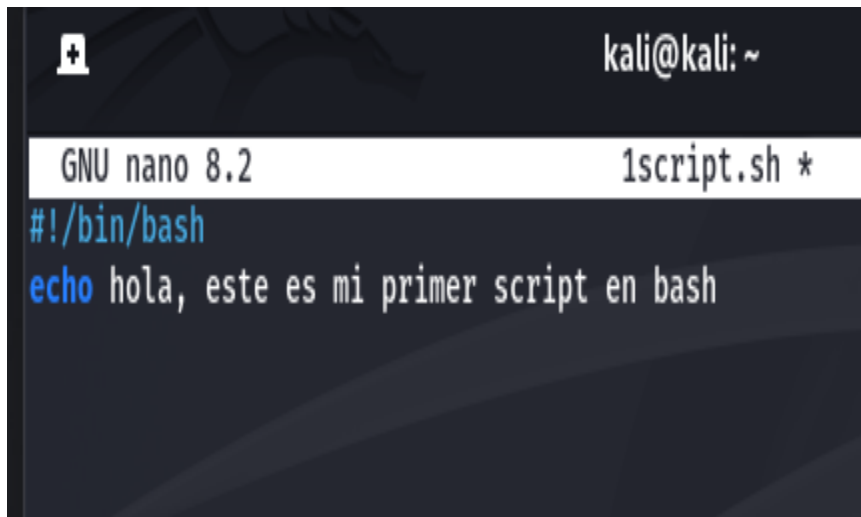
A screenshot of a Linux terminal window. The window title bar shows 'kali@kali: ~' and standard window controls. The terminal content shows a prompt '(kali@kali)-[~]' followed by the command '\$ echo cyberland academy' and its output 'cyberland academy'. A second prompt '(kali@kali)-[~]' is shown below with a '\$' character on the next line.

```
(kali@kali)-[~]  
$ echo cyberland academy  
cyberland academy  
  
(kali@kali)-[~]  
$
```

Como su nombre indica, el comando **echo** se hace eco del texto o de la cadena que quieres que se muestre.

¿Qué es un Script?

Un script en bash es simplemente un archivo de texto que contiene una serie de comandos que el interprete de bash ejecutará en orden. En lugar de escribir comandos uno por uno en la terminal, puedes escribirlos en un archivo y ejecutarlos todos de una vez. Esto es útil cuando necesitas realizar tareas repetitivas o complejas.



```
kali@kali: ~  
GNU nano 8.2 1script.sh *  
#!/bin/bash  
echo hola, este es mi primer script en bash
```

Para crear nuestro primer script abriremos la terminal y escribiremos el siguiente comando ***sudo nano 1script.sh***. Aquí en ***sudo*** le estamos primero que todo dando permiso de administrador para que nos abra el editor de texto que en este caso es ***Nano*** y ***1script.sh*** que es el nombre que tu quieres ponerles seguido de la extensión .sh.

Luego la primera Linea ***#!/bin/bash*** indica al sistema que el archivo debe ejecutarse usando BASH. Para guardar el archivo y salir del editor en Nano Ctrl +X.

Permisos en Linux

Imagina tu computadora como una casa. Cada habitación (directorio) y cada objeto dentro de ella (archivo) tiene una cerradura con diferentes llaves. Estas llaves son los permisos en linux. Ellos determinan quién (usuarios, grupos) puede hacer qué con cada archivo o directorio: leerlo, modificarlo o ejecutarlo.

❏ Tipo de Permisos:

- **Lectura (r):** Permite ver el contenido de un archivo.
- **Escritura (w):** Permite modificar el contenido de un archivo
- **Ejecución (x):** Permite ejecutar un programa

Permisos en Linux I

❑ Los permisos se asignan a 3 categorías:

- Propietario: La persona que creó el archivo.
- Grupo: Un grupo de usuario al que pertenece el archivo.
- Otros: todos los demás usuarios del sistema.

Cuando estamos editando los permisos en linux nos aparecen los 0 y los 1 estos significa 0 que no tiene ningun permiso y 1 si tiene permiso de ejecución.

Permiso de Linux II

Para darle permisos de ejecución a un archivo necesitas escribir en la terminal el siguiente comando `chmod 777 1script.sh`

```
(kali@kali)-[~]  
$ sudo chmod 777 1script.sh  
[sudo] password for kali:
```

Sudo Chmod 777 1script.sh aqui le estoy diciendo que todo el mundo tiene permisos para hacer cualquier cosa.

Estructura básica de un Script Bash

Los scripts Bash tienen una estructura sencilla.

1. Shebang (#! / bin /bash): El shebang es la primera línea de casi todos los scripts bash. Le dice al sistema qué interprete debe usar para ejecutar el script, en este caso, bash. Si no incluyes el shebang, el sistema podría no saber cómo ejecutar tu archivo.

2. Comandos Bash: Cada línea del script es un comando que el sistema ejecutará. Puedes usar cualquier comando que normalmente escribirías en la terminal:

- Echo: comando para la impresión de un texto en pantalla
- ls: muestra todo el contenido de una carpeta.
- pwd: muestra el directorio de trabajo actual
- cp: Copia un archivo

```
(kali㉿kali)-[~]  
$ ls  
18199296_143955  
1script.sh  
Ciclismo.jpg  
Desktop  
Documents  
Downloads
```

```
(kali㉿kali)-[~]  
$ echo cyberland academy  
cyberland academy
```

```
(kali㉿kali)-[~]  
$ pwd  
/home/kali
```

```
(kali㉿kali)-[~]  
$ cp 1script.sh Desktop
```