

## Encadenando comandos: operadores de control 9/21

### RECURSOS

### MARCADORES

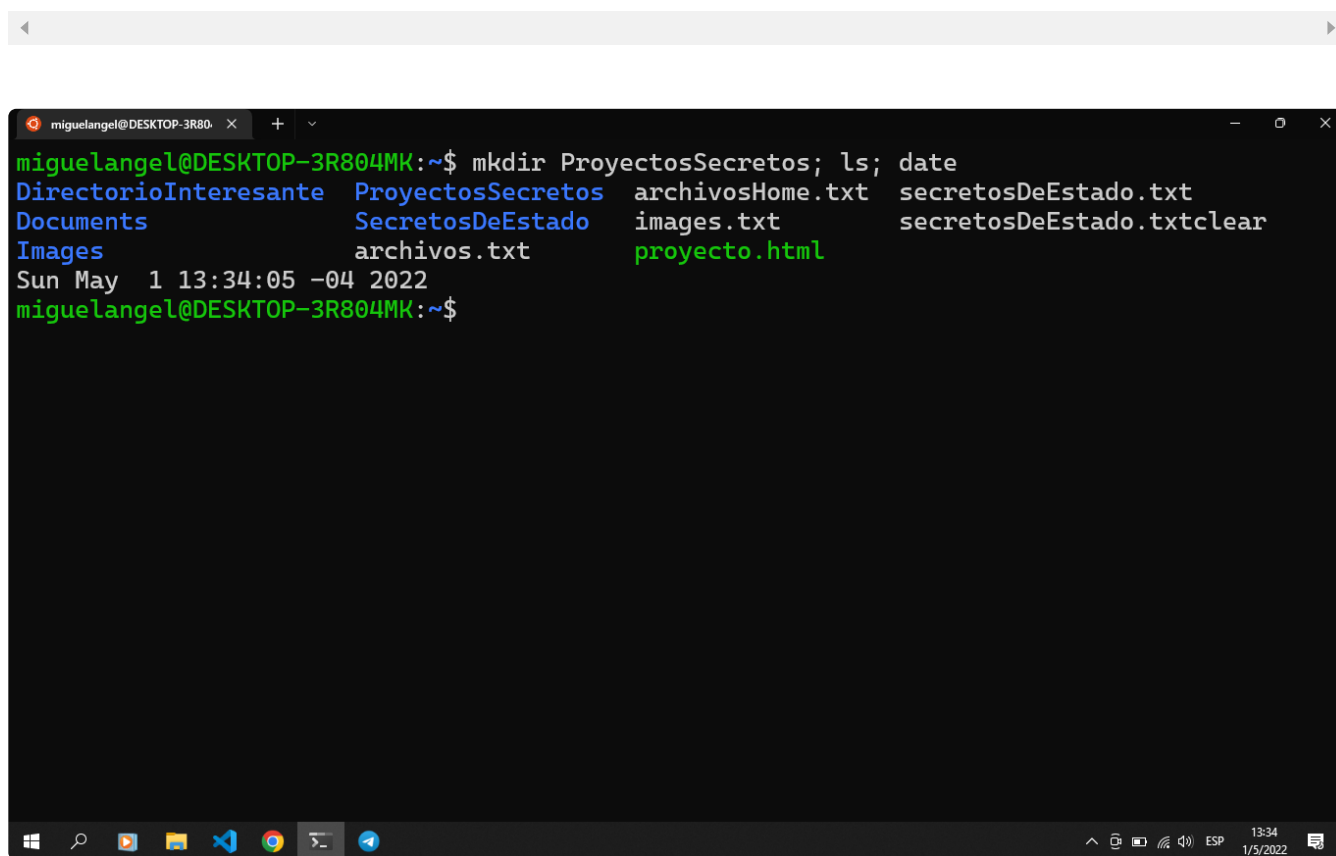
Los operadores de control son símbolos reservados por la terminal que nos permiten encadenar comandos.

Si usas constantemente la tecla enter para ejecutar varios comandos, puedes evitarlo si usas el operador `;` que separa los comandos que estamos ejecutando.

## Comandos en la misma línea ( ; )

Solo necesitas escribir los comandos que quieres ejecutar separados por `;` y luego presionar enter.

```
mkdir ProyectosSecretos; ls; date
```



```
miguelangel@DESKTOP-3R804MK:~$ mkdir ProyectosSecretos; ls; date
DirectorioInteresante  ProyectosSecretos  archivosHome.txt  secretosDeEstado.txt
Documents             SecretosDeEstado  images.txt        secretosDeEstado.txt
Images               archivos.txt      proyecto.html
Sun May  1 13:34:05 -04 2022
miguelangel@DESKTOP-3R804MK:~$
```

Lo que sucedió es:

- Se creó un directorio llamado “Proyectos Secretos”
- Se listaron los archivos

- Se imprimió la fecha actual

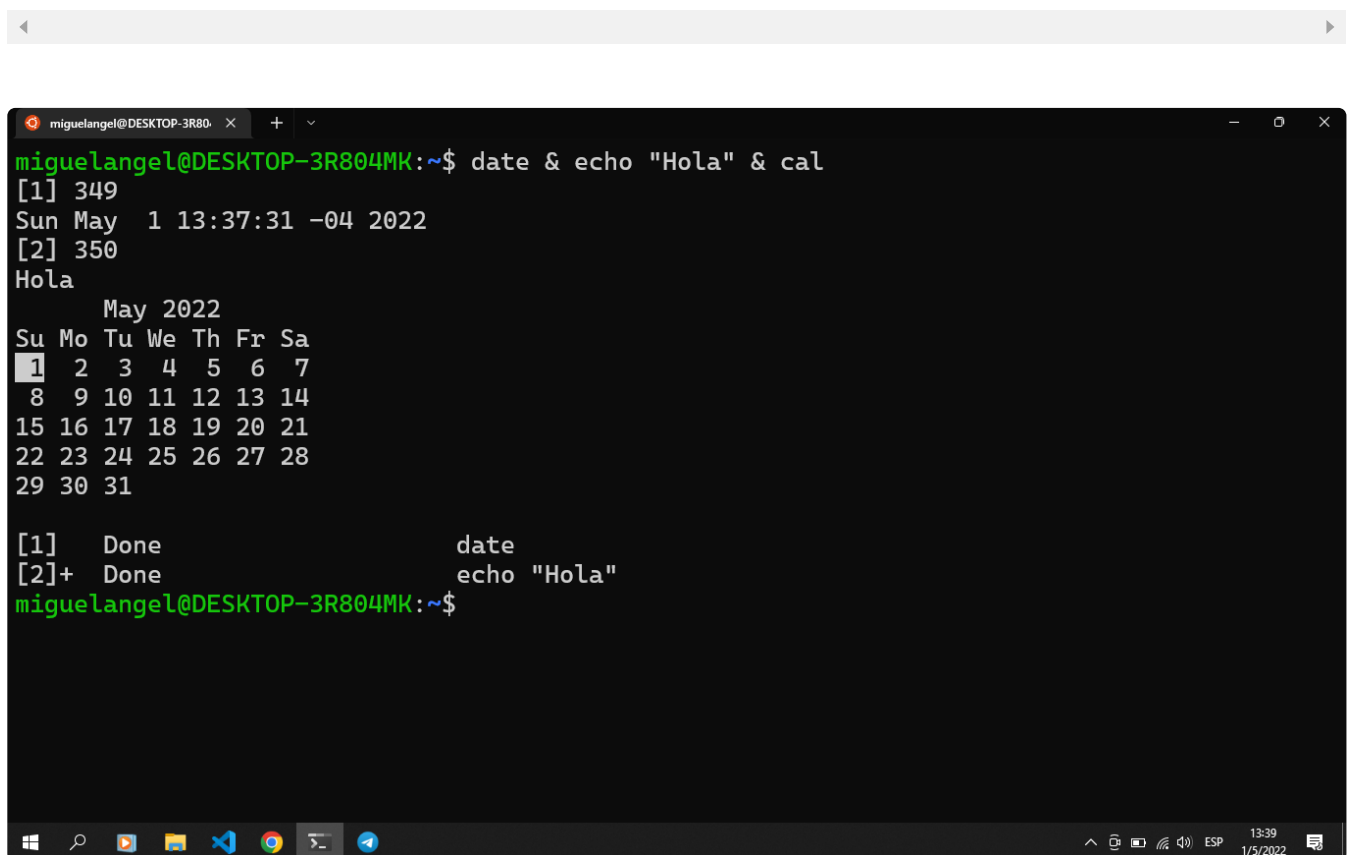
El comando `date` imprime por consola la fecha actual.

## Comandos asíncronos (&)

Cuando queremos ser más eficientes podemos ejecutar varios comandos al mismo tiempo, de modo que no tenemos que esperar a que uno se ejecute para luego ejecutar el que sigue.

Para llevar a cabo varios comandos, al mismo tiempo, usamos el operador **&** entre cada comando que queremos ejecutar.

```
date & echo "Hola" & cal
```



```
miguelangel@DESKTOP-3R80- x + v
miguelangel@DESKTOP-3R804MK:~$ date & echo "Hola" & cal
[1] 349
Sun May  1 13:37:31 -04 2022
[2] 350
Hola
      May 2022
Su Mo Tu We Th Fr Sa
 1  2  3  4  5  6  7
 8  9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 31

[1] Done
[2]+ Done
      date
      echo "Hola"
miguelangel@DESKTOP-3R804MK:~$
```

El comando `cal` imprime un pequeño calendario de la fecha actual y el comando `echo` imprime por el texto que le pases.

En la salida podemos ver que en la primera línea dice `[1] 349` y en la tercera dice `[2] 350`, esto significa que se crearon dos hilos para ejecutar los 3 comandos que se le dieron.

El primero, con el id 349, se usó para ejecutar el comando `date` y el segundo, con el id 350 se usó para ejecutar los comandos `echo` y `cal`.

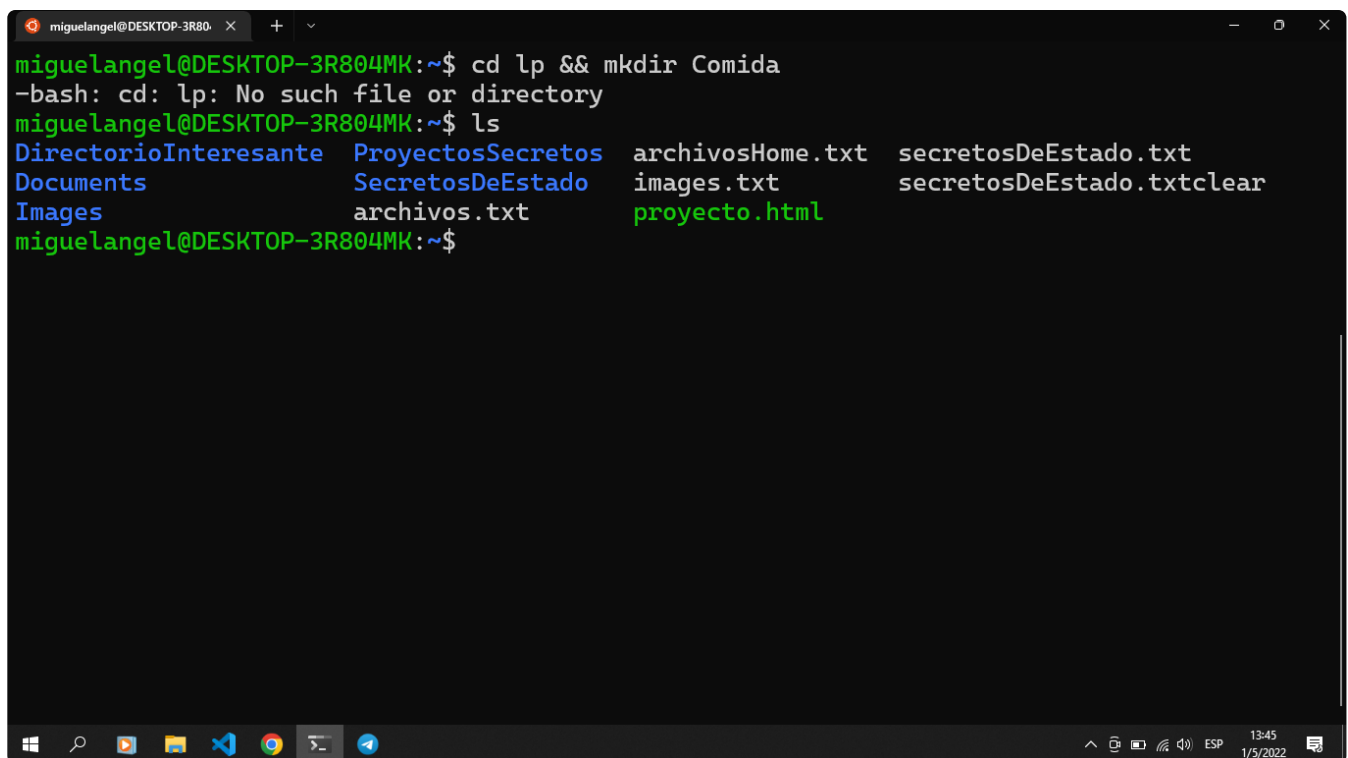
## Comandos con condicionales

Podemos ejecutar comandos dependiendo de condiciones.

### Condición and (&&)

Si escribimos varios comandos separados por el operador **&&** estamos indicando que para que estos se ejecuten, el comando anterior tuvo que ejecutarse correctamente.

```
cd lp && mkdir Comida
```



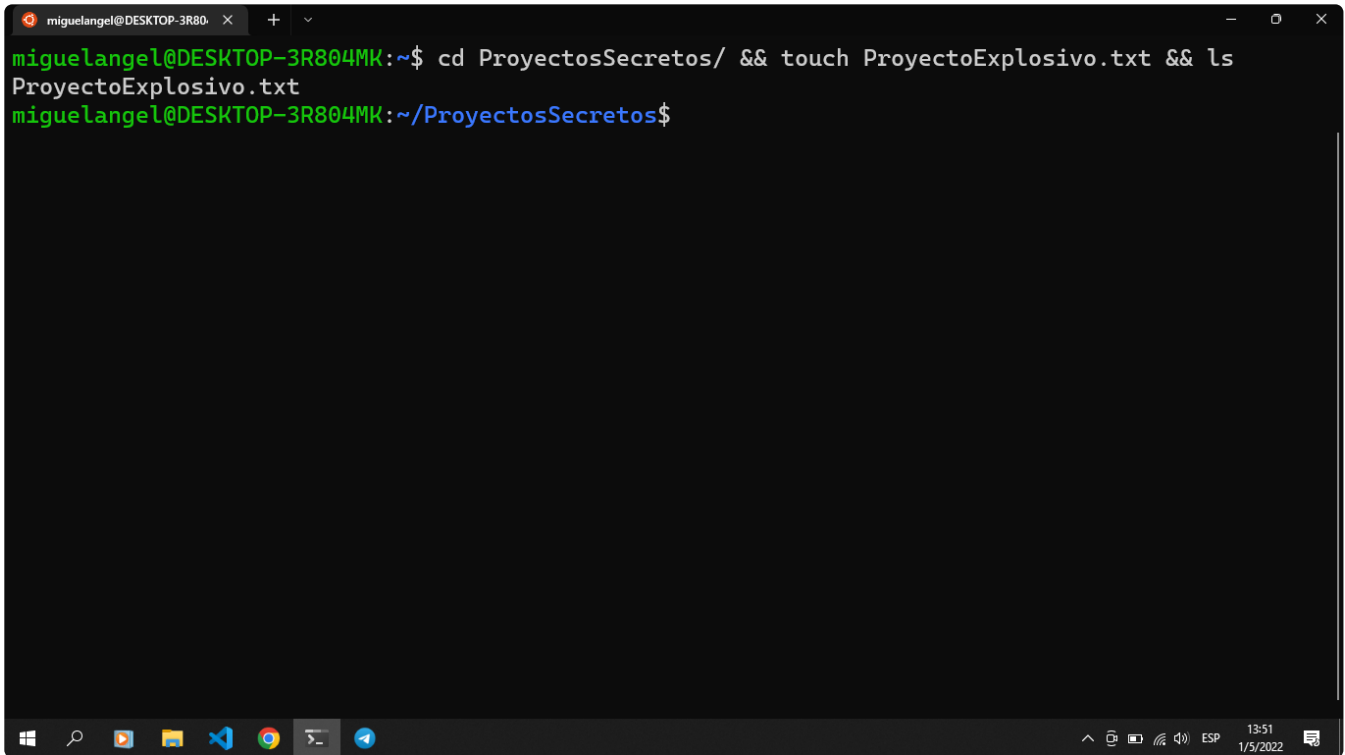
```
miguelangel@DESKTOP-3R804MK:~$ cd lp && mkdir Comida
-bash: cd: lp: No such file or directory
miguelangel@DESKTOP-3R804MK:~$ ls
DirectorioInteresante  ProyectosSecretos  archivosHome.txt  secretosDeEstado.txt
Documents              SecretosDeEstado  images.txt         secretosDeEstado.txtclear
Images                 archivos.txt       proyecto.html
miguelangel@DESKTOP-3R804MK:~$
```

(Recuerda que el comando `cd`, **Change Directory** te cambia al directorio que le indiques).

En este ejemplo, intentamos cambiar al directorio “lp” pero ese directorio no existe, así que el comando falla y no se ejecuta el siguiente.

En caso de que el primer comando se haya ejecutado correctamente pasará al siguiente, y después verá si ese se puede ejecutar.

```
cd ProyectosSecretos/ && touch ProyectoExplosivo.txt && ls
```

A terminal window titled 'miguelangel@DESKTOP-3R804MK' with a dark background. The prompt is 'miguelangel@DESKTOP-3R804MK:~\$'. The user enters the command 'cd ProyectosSecretos/ && touch ProyectoExplosivo.txt && ls'. The output shows 'ProyectoExplosivo.txt' on the next line, followed by the prompt 'miguelangel@DESKTOP-3R804MK:~/ProyectosSecretos\$'. The Windows taskbar is visible at the bottom with various icons and a system tray showing the time as 13:51 on 1/5/2022.

```
miguelangel@DESKTOP-3R804MK:~$ cd ProyectosSecretos/ && touch ProyectoExplosivo.txt && ls
ProyectoExplosivo.txt
miguelangel@DESKTOP-3R804MK:~/ProyectosSecretos$
```

Como en este caso todos los comandos se ejecutaron correctamente sucede esto:

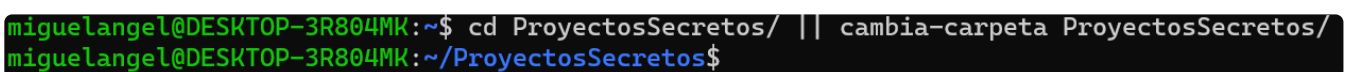
- Cambia el directorio “ProyectosSecretos”
- Crea el archivo “ProyectoExplosivo.txt”
- Lista el contenido del directorio con `ls`

## Condicional or (||)

Al condicional or no le importa si el comando anterior se ejecutó o no, simplemente va probando todos los comandos a ver cuál se ejecuta.

Por un momento vamos a suponer que no sabemos muy bien cuál es el comando para cambiar de directorio si es `cd` o `cambia-carpeta`, entonces para evitar un error escribimos:

```
cd ProyectosSecretos/ || cambia-carpeta ProyectosSecretos/
```

A terminal window showing the command 'cd ProyectosSecretos/ || cambia-carpeta ProyectosSecretos/' being executed. The prompt changes to 'miguelangel@DESKTOP-3R804MK:~/ProyectosSecretos\$' after the command runs.

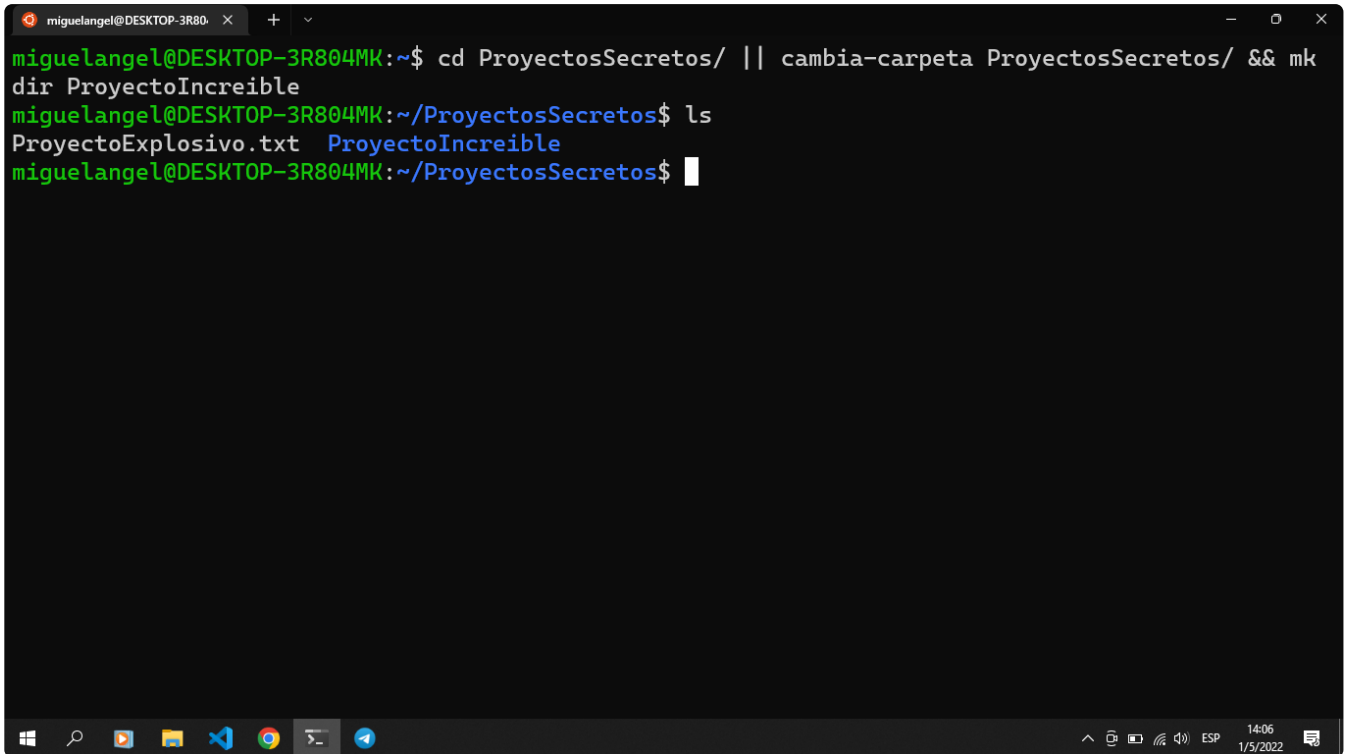
```
miguelangel@DESKTOP-3R804MK:~$ cd ProyectosSecretos/ || cambia-carpeta ProyectosSecretos/
miguelangel@DESKTOP-3R804MK:~/ProyectosSecretos$
```

Aquí vemos que alguno de los dos comandos está mal, pero igual cambió la carpeta porque uno de esos funcionó.

## Combinando operadores de control

Siguiendo con el ejemplo anterior, vamos a cambiar de directorio. Si se logra cambiar de directorio creamos una carpeta adentro.

```
cd ProyectosSecretos/ || cambia-carpeta ProyectosSecretos/ && mkdir  
ProyectoIncreible
```



```
miguelangel@DESKTOP-3R804MK:~$ cd ProyectosSecretos/ || cambia-carpeta ProyectosSecretos/ && mkdir ProyectoIncreible
miguelangel@DESKTOP-3R804MK:~/.ProyectosSecretos$ ls
ProyectoExplosivo.txt  ProyectoIncreible
miguelangel@DESKTOP-3R804MK:~/.ProyectosSecretos$
```

Esto funcionó porque el operador **or** devolvió verdadero al tener un comando que funcionara, por lo tanto, el operador **and** lo interpretó como un comando que funcionó correctamente.

Si esto último se te es un poco complicado te invito a que tomes el [Curso de Pensamiento Lógico](#).

## Tabla de operadores

Operador	Función
;	Ejecuta de forma <b>síncrona</b> los comandos especificados
&	Ejecuta de forma <b>asíncrona</b> los comandos especificados
&&	Ejecuta el comando si el anterior se ejecutó correctamente
	Ejecuta el comando si el anterior o la combinación de los anteriores resultaron en verdadero
Comando	Función

Comando	Función
echo	Imprime el mensaje que le indiqués
cal	De <b>cal</b> endar imprime un calendario con la fecha actual
date	Imprime la fecha actual