



**Universidad Autónoma del Estado de México**

**Centro Universitario UAEM Zumpango**

Ingeniería en Computación

**Sistemas Operativos**

**Tarea.**

Script Bash

**Presenta:**

Yesenia Martínez Galván

**Docente:**

Hazem Álvarez Rodríguez

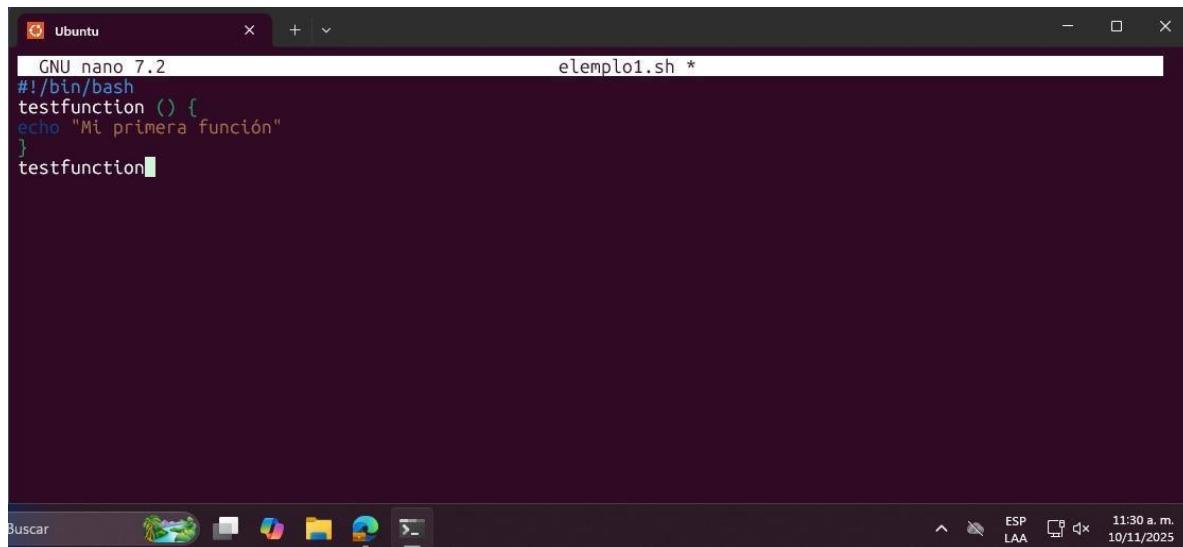
Zumpango, Estado de México a 11 de noviembre de 2025

# Funciones

## Ejemplo1: Comienza con una función echo sencilla

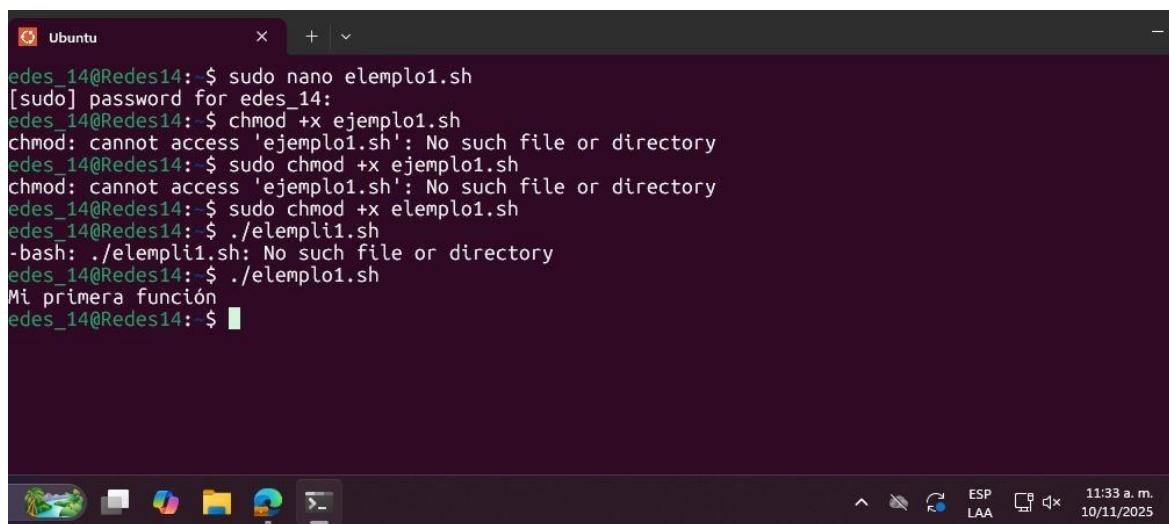
Cuando quieras escribir un archivo de script bash, utiliza el comando nano filename.sh para crear y abrir un archivo .sh, y comenzar a escribir tus funciones bash. No olvides salir y guardar el archivo cuando hayas terminado.

- Primero creamos el archivo del ejemplo1 con el comando **sudo nano ejemplo1.sh**
- Empecemos con una simple función echo. Empieza definiendo el nombre de la función seguido del comando echo en la línea siguiente



```
GNU nano 7.2
#!/bin/bash
testfunction () {
echo "Mi primera función"
}
testfunction
```

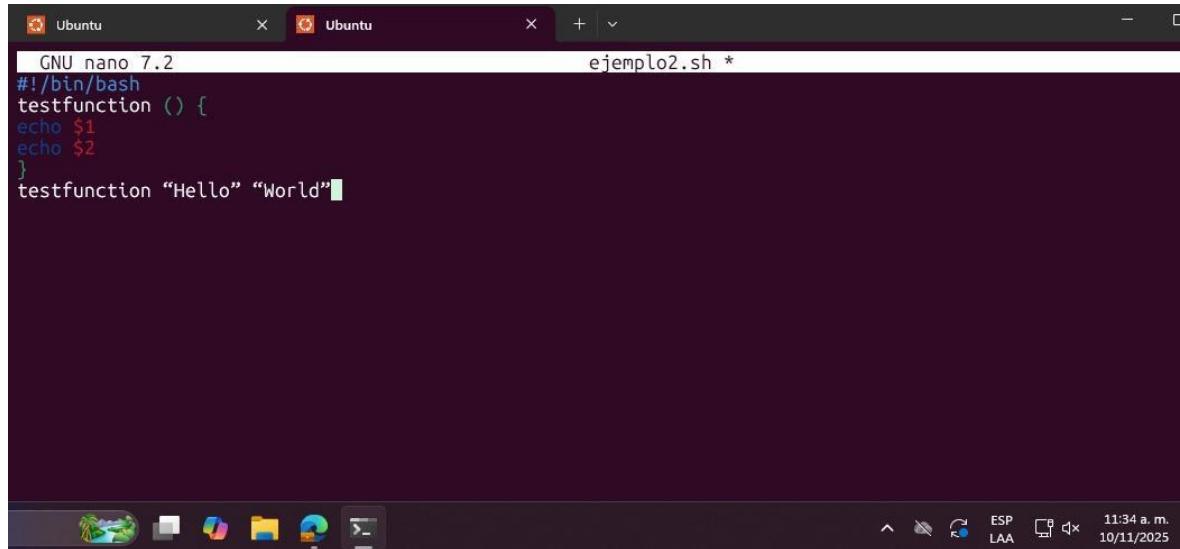
- Le damos permisos de ejecución con el comando **chmod +x ejemplo1.sh**
- Y agregamos el comando **./ejemplo1.sh** para que ejecute y como resultado se visualice la frase “Mi primera función”



```
edes_14@Redes14:~$ sudo nano ejemplo1.sh
[sudo] password for edes_14:
edes_14@Redes14:~$ chmod +x ejemplo1.sh
chmod: cannot access 'ejemplo1.sh': No such file or directory
edes_14@Redes14:~$ sudo chmod +x ejemplo1.sh
chmod: cannot access 'ejemplo1.sh': No such file or directory
edes_14@Redes14:~$ sudo chmod +x exemplo1.sh
edes_14@Redes14:~$ ./ejemplo1.sh
-bash: ./ejemplo1.sh: No such file or directory
edes_14@Redes14:~$ ./ejemplo1.sh
Mi primera función
edes_14@Redes14:~$
```

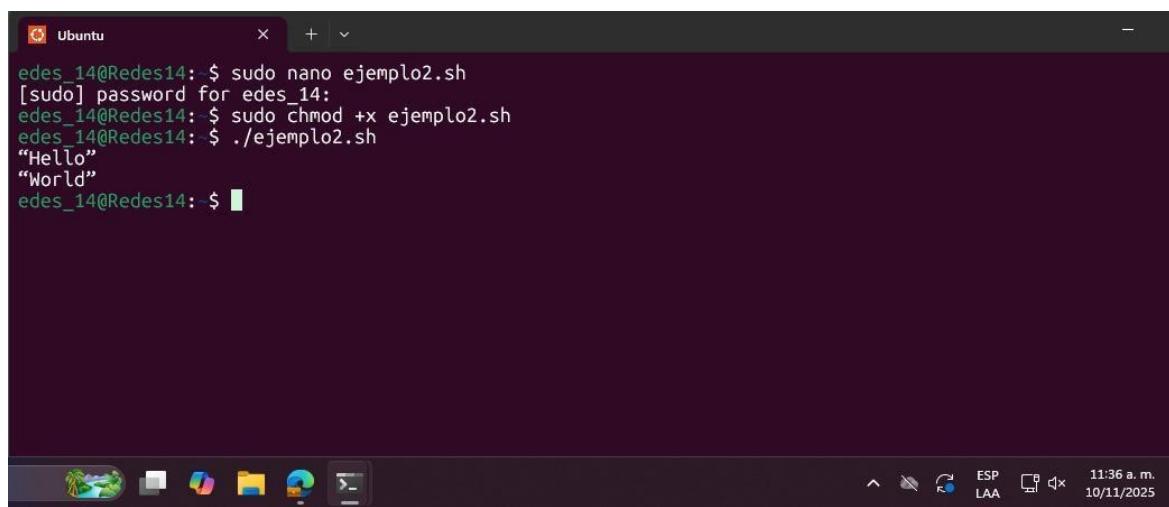
## Ejemplo2: Utiliza pocos parámetros

- Al igual que el ejemplo anterior se crea el archivo, dentro de la función bash y se guarda
- Las funciones bash aceptan cualquier número de parámetros. El siguiente ejemplo acepta dos parámetros



```
GNU nano 7.2
#!/bin/bash
testfunction () {
echo $1
echo $2
}
testfunction "Hello" "World"
```

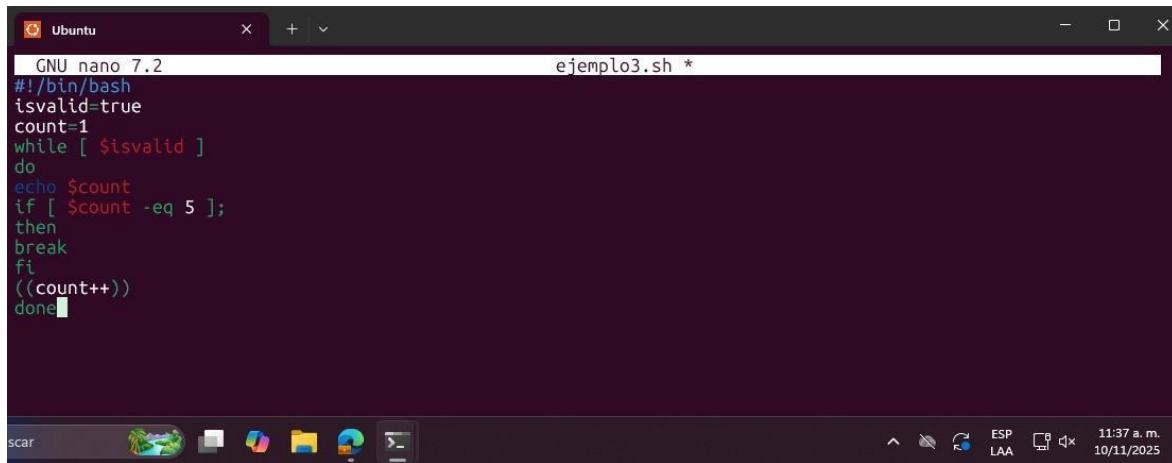
- **\$1** representa el primer argumento, mientras que **\$2** representa el segundo argumento en la línea de ejecución de la función. Como hemos utilizado “Hello” y “World” para los argumentos
- Una vez teniendo el documento con la función se le da permisos con **cmod +x ejemplo2.sh**
- Se ejecuta con **./ejemplo2.sh**



```
edes_14@Redes14: $ sudo nano ejemplo2.sh
[sudo] password for edes_14:
edes_14@Redes14: $ sudo chmod +x ejemplo2.sh
edes_14@Redes14: $ ./ejemplo2.sh
"Hello"
"World"
edes_14@Redes14: $
```

### **Ejemplo3: Combinar bucles y condicionales**

- Se crea el archivo **ejemplo3.sh**
- Los bucles y las sentencias condicionales también son populares en los scripts bash. Vamos a ver algunos casos de uso de ambos en el mismo script:



The screenshot shows a terminal window titled "Ubuntu" running the "nano" text editor. The file being edited is "ejemplo3.sh". The code inside the file is as follows:

```
GNU nano 7.2                                     exemplo3.sh *
#!/bin/bash
isvalid=true
count=1
while [ $isvalid ]
do
echo $count
if [ $count -eq 5 ];
then
break
fi
((count++))
done
```

- Se le otorgan permisos con con **chmod +x ejemplo3.sh**
- Y se ejecuta con **./ejemplo3.sh**



The screenshot shows a terminal window with the following session:

```
edes_14@Redes14:~$ 
edes_14@Redes14:~$ 
edes_14@Redes14:~$ sudo nano exemplo3.sh
edes_14@Redes14:~$ sudo chmod +x exemplo3.sh
edes_14@Redes14:~$ ./ejemplo3.sh
1
2
3
4
5
edes_14@Redes14:~$
```

### **Referencia:**

Hostinger. (26 de julio de 2024). *Bash Script: qué es, cómo escribir uno y ejemplos.* [https://www.hostinger.com/mx/tutoriales/bash-script-linux#4\\_funciones\\_sencillas-para-probar-en-tu-primer-bash-script](https://www.hostinger.com/mx/tutoriales/bash-script-linux#4_funciones_sencillas-para-probar-en-tu-primer-bash-script)