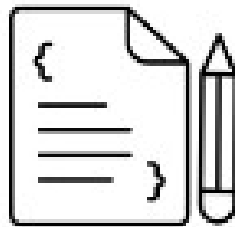


SCRIPTING



FUNCIONES

Conceptos básicos

1. Shebang (#!)
2. Variables
3. Operadores
4. Estructuras de control
- 5. Funciones**





Las funciones son bloques de código que realizan una **tarea específica** y pueden ser **reutilizadas** en diferentes partes de un script

- Modularizar
- Organizar



- ✓ Reutilización
- ✓ Facilita mantenimiento del código



Definición de funciones

Sintaxis

```
nombre_de_la_funcion() {  
    # Código de la función  
}
```

Ejemplo

```
saludar() {  
    echo "¡Hola, mundo!"  
}
```



Llamada a funciones

Llamada

saludar



Pasar argumentos a funciones

Se pasan al llamar a la función

Dentro del cuerpo de la función, estos argumentos están disponibles a través de las variables posicionales **\$1**, **\$2**, etc.

```
saludar() {  
    echo "¡Hola, $1 y $2!"  
}  
  
saludar Juan María
```

The diagram illustrates the argument passing process. The function `saludar()` is defined with a body that uses positional variables `$1` and `$2` to print a greeting. When the function is called as `saludar Juan María`, the arguments `Juan` and `María` are passed to the function. The diagram shows arrows pointing from `Juan` to `$1` and from `María` to `$2`, indicating that the arguments are substituted into the variables within the function's scope.



Valor de retorno de funciones

Pueden devolver un valor utilizando la palabra clave **return**

El valor devuelto
puede ser
almacenado en una
variable al llamar a
la función.

```
sumar() {  
    local resultado=$(( $1 + $2 ))  
    return $resultado  
}  
  
sumar 5 3  
resultado=$?  
echo "La suma es: $resultado"
```

Variables locales en funciones



Por defecto, todas las variables en Bash **son globales**, lo que significa que están disponibles para todas las funciones y comandos en el script.

```
mi_funcion() {  
    local variable_local="Hola, mundo!"  
    echo $variable_local  
}
```

Puedes declarar variables locales dentro de una función utilizando el comando **local**

```
mi_funcion  
echo $variable_local # Esto dará un error, ya que la variable es local a la función
```




```
#!/bin/bash
```

```
saludar() {  
    echo "¡Hola, $1!"  
}
```

```
nombre="Juan"  
saludar $nombre
```

Actividad

P26 – Scripting IV while & funciones