1.Saludo: Crea una función llamada `saludar` que reciba el nombre de una persona como argumento y muestre un saludo personalizado.

2.Calculadora simple: Desarrolla una función llamada `calculadora` que reciba dos números y una operación (suma, resta, multiplicación, división) como argumentos y muestre el resultado de la operación.

3.Factorial: Escribe una función llamada `factorial` que calcule el factorial de un número dado como argumento.

4.Tabla de multiplicar: Crea una función llamada `tabla_multiplicar` que genere la tabla de multiplicar de un número dado como argumento.

5.Conteo de palabras: Desarrolla una función llamada `contar_palabras` que cuente el número de palabras en un archivo de texto dado como argumento.

6.Ordenar números: Escribe una función llamada `ordenar_numeros` que ordene una lista de números dados como argumentos de menor a mayor.

7.Generador de contraseñas: Crea una función llamada `generar_contrasena` que genere una contraseña aleatoria con una longitud específica dada como argumento.

8.Fibonacci: Desarrolla una función llamada `fibonacci` que genere los primeros n términos de la secuencia de Fibonacci.

10.Buscar en archivos: Escribe una función llamada `buscar_archivos` que busque archivos con un nombre o patrón específico en un directorio dado como argumento.

11.Backup de archivos: Crea una función llamada `backup_archivos` que realice copias de seguridad de archivos o directorios específicos.

```
#!/bin/bash

# 1. Saludo
saludar() {
    echo "¡Hola, $1!"
}

# Llamada a la función
saludar "Juan"

# 2. Calculadora simple
calculadora() {
    local resultado=0
    case $3 in
        suma)
            resultado=$(($1 + $2))
            ;;
        resta)
            resultado=$(($1 - $2))
            ;;
        multiplicacion)
            resultado=$(($1 * $2))
            ;;
        division)
            resultado=$(($1 / $2))
            ;;
        *)
            echo "Operación no válida"
            return 1
            ;;
    esac
    echo "Resultado de $1 $3 $2 = $resultado"
}

# Llamada a la función
```

```
calculadora 10 5 suma
```

3. Factorial

```
factorial() {
    if [ $1 -eq 0 ] || [ $1 -eq 1 ]; then
        echo "Factorial de $1 = 1"
    else
        local resultado=1
        for ((i = 2; i <= $1; i++)); do
            resultado=$(($resultado * $i))
        done
        echo "Factorial de $1 = $resultado"
    fi
}

# Llamada a la función
factorial 5
```

4. Tabla de multiplicar

```
tabla_multiplicar() {
    for ((i = 1; i <= 10; i++)); do
        echo "$1 x $i = $(($1 * $i))"
    done
}

# Llamada a la función
tabla_multiplicar 7
```

5. Conteo de palabras

```
contar_palabras() {
    local archivo=$1
    local conteo=$(wc -w <"$archivo")
    echo "El archivo $archivo contiene $conteo palabras."
}

# Llamada a la función
contar_palabras archivo.txt
```

6. Ordenar números

```
ordenar_numeros() {
    local numeros=("$@")
    local sorted=($(printf '%s\n' "${numeros[@]}" | sort -n))
    echo "Números ordenados: ${sorted[*]}"
}

# Llamada a la función
ordenar_numeros 5 3 9 1 7
```

7. Generador de contraseñas

```
generar_contrasena() {
    local longitud=$1
    tr -dc '[:alnum:]' < /dev/urandom | head -c $longitud
    echo
}

# Llamada a la función
generar_contrasena 10
```

8. Fibonacci

```
fibonacci() {
    local n=$1
    local a=0
    local b=1
    local resultado="0 1"

    for ((i = 2; i < n; i++)); do
        local temp=$((a + b))
        resultado+=" $temp"
        a=$b
        b=$temp
    done

    echo "Secuencia de Fibonacci ($n términos): $resultado"
}

# Llamada a la función
fibonacci 10
```

9. Buscar en archivos

```
buscar_archivos() {
    local patron=$1
    local directorio=$2
    echo "Archivos que coinciden con '$patron' en '$directorio':"
    find "$directorio" -type f -name "$patron"
}

# Llamada a la función
buscar_archivos "*.txt" "/ruta/a/directorio"
```

10. Backup de archivos

```
backup_archivos() {
    local origen=$1
    local destino=$2
    tar -czvf "$destino/backup_$(date +%Y%m%d).tar.gz" "$origen"
```

```
    echo "Copia de seguridad creada en $destino"
}

# Llamada a la función
backup_archivos "/ruta/a/archivos" "/ruta/a/destino"
```