## Ejercicio 1

Completar el siguiente script el cual simula un cajero automático, con las siguientes indicaciones:

1. El script inicia con un saldo de 1000.
2. Se muestra un mensaje de bienvenida junto con el saldo actual.
3. Dentro de un bucle *while*, se pide al usuario que elija una operación:
   - depositar**:** se le solicitará una cantidad y se sumará al saldo.
   - retirar**:** se le solicitará una cantidad y, si hay saldo suficiente, se restará del saldo. Si el saldo es insuficiente, se mostrará un mensaje de error.
   - salir**:** se termina el programa.
4. El bucle se repetirá hasta que el usuario introduzca la opción salir.
5. Al finalizar, se mostrará el saldo final.

```
#!/bin/bash
# Ejercicio: Simulación de cajero automático
# Instrucciones:
# 1. Inicia el saldo en 1000.
# 2. Pregunta al usuario la operación a realizar:
#    - "depositar": pedir cantidad y sumar al saldo.
#      - "retirar": pedir cantidad, verificar saldo suficiente y restar la
cantidad.
#    - "salir": terminar el programa.
# 3. Al salir, muestra el saldo final.
#
# Completar las partes indicadas para que el script funcione correctamente.

saldo=1000

echo "Bienvenido al Cajero Automático"
echo "Saldo inicial: $saldo"

while true; do
    read -p "¿Qué operación deseas realizar? (depositar/retirar/salir): " opcion

    case $opcion in
        depositar)

            ;;
        retirar)

            ;;
        salir)
            break
            ;;
        *)
            echo "Opción inválida. Por favor, intenta de nuevo."
            ;;
    esac
done
```

```
echo "Saldo final: $saldo"
```

## Ejercicio 2

Completar el siguiente programa el cual gestiona una lista de estudiantes y muestra el siguiente menú de opciones:

- Agregar un estudiante al array.
- Mostrar el número total de estudiantes.
- Imprimir los nombres de los estudiantes en orden alfabético.

```
#!/bin/bash
# Ejercicio: Gestión de una lista de estudiantes con arrays
# Dado un array inicial de estudiantes, se deben completar las siguientes funciones:
#   1. añadir_estudiante: Recibe un nombre y lo agrega al array 'estudiantes'.
#   2. contar_estudiantes: Muestra el número total de estudiantes en el array.
#      3. ordenar_estudiantes: Imprime el array de estudiantes ordenado alfabéticamente.
#

# Array inicial de estudiantes
estudiantes=("Juan" "Marta" "Laura" "Miguel" "Sofia")

# Función para agregar un estudiante
function añadir_estudiante () {

}

# Función para contar el número de estudiantes
function contar_estudiantes () {

}

# Función para mostrar los estudiantes ordenados alfabéticamente
function ordenar_estudiantes () {

}

# Menú interactivo
while true; do
    echo "Menú de opciones:"
    echo "1) Agregar estudiante"
    echo "2) Mostrar número de estudiantes"
    echo "3) Mostrar estudiantes en orden alfabético"
    echo "4) Salir"
    read -p "Seleccione una opción: " option
    case $option in
```

```
        1)
            read -p "Ingrese el nombre del estudiante: " nombre
            añadir_estudiante "$nombre"
            ;;
        2)
            contar_estudiantes
            ;;
        3)
            ordenar_estudiantes
            ;;
        4)
            echo "Saliendo..."
            break
            ;;
        *)
            echo "Opción inválida. Intente nuevamente."
            ;;
    esac
    echo ""
done
```

## Ejercicio 3

Completar el siguiente programa consistente en procesar un archivo de texto línea por línea. Para cada línea, el script debe contar el número de palabras y, según la cantidad, escribir la información en uno de dos archivos de salida:

- Si la línea tiene más de 5 palabras, se debe escribir en el archivo *output.txt*.
- Si tiene 5 o menos palabras, se debe escribir en el archivo *líneas_cortas.txt*.

Además, el script debe mantener un contador global del total de líneas procesadas y del total de palabras contadas, imprimiendo un resumen al finalizar.

```
#!/bin/bash

# Este script procesa un archivo de texto.
# Uso: ./script.sh archivo.txt

if [ "$#" -ne 1 ]; then
    echo "Uso: $0 archivo.txt"
    exit 1
fi

archivo_entrada="$1"

if [ ! -f "$archivo_entrada" ]; then
    echo "Error: El archivo '$archivo_entrada' no existe."
    exit 1
fi

# Función para procesar cada línea.
# Completa la funcionalidad de esta función:
# - Contar el número de palabras en la línea.
# - Si la línea tiene más de 5 palabras, escribir la salida en "salida.txt".
# - Si tiene 5 o menos palabras, escribir en "lineas_cortas.txt".
# - Actualizar una variable global con el total de palabras contadas.
procesar_linea() {
    local linea="$1"

}

# Función para imprimir un resumen final.
# Completa la funcionalidad para mostrar:
# - El número total de líneas procesadas.
# - El total de palabras contadas.
imprimir_resumen() {


}

# Variables para el resumen.
total_lineas=0
```

```
total_palabras=0

# Procesa cada línea del archivo.
while read -r linea; do
    total_lineas=$((total_lineas + 1))
    procesar_linea "$linea"
done < "$archivo_entrada"

# Imprime el resumen final.
imprimir_resumen
```

## Ejercicio 4

Completar el siguiente script, el cual muestra un menú interactivo con las siguientes opciones:

- Mostrar Particiones (fdisk -l): Ejecuta el comando fdisk -l para mostrar la información de las particiones del sistema.
- Mostrar UUID y Etiquetas (blkid): Ejecuta el comando blkid para mostrar los identificadores y etiquetas de las particiones.
- Listar Discos y Particiones (lsblk): Ejecuta el comando lsblk para listar la estructura de discos y particiones.
- Exportar Información a Archivo: El script combinará la información obtenida de los tres comandos en un solo reporte y la guardará en un archivo cuyo nombre será proporcionado por el usuario.
- Salir: Finaliza la ejecución del script.

```
#!/bin/bash

echo "Gestor avanzado de información de particiones"

select opcion in "Mostrar Particiones (fdisk -l)" "Mostrar UUID y Etiquetas
(blkid)" "Listar Discos y Particiones (lsblk)" "Exportar Información a
archivo" "Salir"
do
    case $opcion in
        "Mostrar Particiones (fdisk -l)")

            ;;
        "Mostrar UUID y Etiquetas (blkid)")

            ;;
        "Listar Discos y Particiones (lsblk)")

            ;;

        "Exportar Información a archivo")

            ;;
        "Salir")
```

```
            echo "Saliendo..."
            break
            ;;
        *)
            echo "Opción inválida, intente nuevamente."
            ;;
    esac
done
```