Trabajo Práctico 2

Programación Estructurada

Estructuras Condicionales:

1. Verificación de Año Bisiesto.
   Escribe un programa en Java que solicite al usuario un año y determine si es bisiesto. Un año es bisiesto si es divisible por 4, pero no por 100, salvo que sea divisible por 400.

   Ejemplo de entrada/salida:
   Ingrese un año: 2024
   El año 2024 es bisiesto.

   ```
   run:
   Ingrese un anio: 2024
   El anio 2024 es bisiesto.
   BUILD SUCCESSFUL (total time: 5 seconds)
   ```

   Ingrese un año: 1900
   El año 1900 no es bisiesto.

   ```
   run:
   Ingrese un anio: 1900
   El anio 1900 no es bisiesto.
   BUILD SUCCESSFUL (total time: 4 seconds)
   ```

2. Determinar el Mayor de Tres Números.
   Escribe un programa en Java que pida al usuario tres números enteros y determine cuál es el mayor.

   Ejemplo de entrada/salida:
   Ingrese el primer número: 8
   Ingrese el segundo número: 12
   Ingrese el tercer número: 5
   El mayor es: 12

   ```
   run:
   Ingrese el primer numero: 8
   Ingrese el segundo numero: 12
   Ingrese el tercer numero: 5
   El mayor es: 12
   BUILD SUCCESSFUL (total time: 8 seconds)
   ```

3. Clasificación de Edad.
   Escribe un programa en Java que solicite al usuario su edad y clasifique su etapa de vida según la siguiente tabla:
   Menor de 12 años: "Niño"
   Entre 12 y 17 años: "Adolescente"
   Entre 18 y 59 años: "Adulto"
   60 años o más: "Adulto mayor"

   Ejemplo de entrada/salida:
   Ingrese su edad: 25
   Eres un Adulto.

```
run:
Ingrese su edad: 25
Eres un Adulto.
BUILD SUCCESSFUL (total time: 3 seconds)
```

   Ingrese su edad: 10
   Eres un Niño.

```
run:
Ingrese su edad: 10
Eres un Ninio.
BUILD SUCCESSFUL (total time: 2 seconds)
```

4. Calculadora de Descuento según categoría.
   Escribe un programa que solicite al usuario el precio de un producto y su categoría (A, B o C).
   Luego, aplique los siguientes descuentos:
   Categoría A: 10% de descuento
   Categoría B: 15% de descuento
   Categoría C: 20% de descuento

   El programa debe mostrar el precio original, el descuento aplicado y el precio final

   Ejemplo de entrada/salida:
   Ingrese el precio del producto: 1000
   Ingrese la categoría del producto (A, B o C): B
   Descuento aplicado: 15%
   Precio final: 850.0

```
run:
Ingrese el precio del producto: 1000
Ingrese la categoria del producto (A, B o C): B
Descuento aplicado: 15%
Precio final: 850.0
BUILD SUCCESSFUL (total time: 8 seconds)
```

Estructuras de Repetición:

5. Suma de Números Pares (while).
   Escribe un programa que solicite números al usuario y sume solo los números pares. El ciclo debe continuar hasta que el usuario ingrese el número 0, momento en el que se debe mostrar la suma total de los pares ingresados.

   Ejemplo de entrada/salida:
   Ingrese un número (0 para terminar): 4
   Ingrese un número (0 para terminar): 7
   Ingrese un número (0 para terminar): 2
   Ingrese un número (0 para terminar): 0
   La suma de los números pares es: 6

```
run:
Ingrese numeros (0 para terminar):
Ingrese un numero: 4
Ingrese un numero: 7
Ingrese un numero: 2
Ingrese un numero: 0
La suma de los numeros pares es: 6
BUILD SUCCESSFUL (total time: 9 seconds)
```

6. Contador de Positivos, Negativos y Ceros (for).
   Escribe un programa que pida al usuario ingresar 10 números enteros y cuente cuántos son positivos, negativos y cuántos son ceros.

   Ejemplo de entrada/salida:
   Ingrese el número 1: -5
   Ingrese el número 2: 3
   Ingrese el número 3: 0
   Ingrese el número 4: -1
   Ingrese el número 5: 6
   Ingrese el número 6: 0
   Ingrese el número 7: 9
   Ingrese el número 8: -3
   Ingrese el número 9: 4
   Ingrese el número 10: -8
   Resultados:
   Positivos: 4
   Negativos: 4
   Ceros: 2

```
run:
Ingrese el numero 1: -5
Ingrese el numero 2: 3
Ingrese el numero 3: 0
Ingrese el numero 4: -1
Ingrese el numero 5: 6
Ingrese el numero 6: 0
Ingrese el numero 7: 9
Ingrese el numero 8: -3
Ingrese el numero 9: 4
Ingrese el numero 10: -8
Resultados:
Positivos: 4
Negativos: 4
Ceros: 2
BUILD SUCCESSFUL (total time: 38 seconds)
```

7. Validación de Nota entre 0 y 10 (do-while).
   Escribe un programa que solicite al usuario una nota entre 0 y 10. Si el usuario ingresa un número fuera de este rango, debe seguir pidiéndole la nota hasta que ingrese un valor válido.

   Ejemplo de entrada/salida:
   Ingrese una nota (0-10): 15
   Error: Nota inválida. Ingrese una nota entre 0 y 10.
   Ingrese una nota (0-10): -2
   Error: Nota inválida. Ingrese una nota entre 0 y 10.
   Ingrese una nota (0-10): 8
   Nota guardada correctamente.

```
run:
Ingrese una nota (0-10): 15
Error: Nota invalida. Ingrese una nota entre 0 y 10.
Ingrese una nota (0-10): -2
Error: Nota invalida. Ingrese una nota entre 0 y 10.
Ingrese una nota (0-10): 8
Nota guardada correctamente.
BUILD SUCCESSFUL (total time: 10 seconds)
```

Funciones:

8. Cálculo del Precio Final con impuesto y descuento.
   Crea un método calcularPrecioFinal(double impuesto, double descuento) que calcule el precio final de un producto en un e-commerce.
   La fórmula es:
   PrecioFinal = PrecioBase + (PrecioBase×Impuesto) − (PrecioBase×Descuento) PrecioFinal = PrecioBase + (PrecioBase \times Impuesto) - (PrecioBase \times Descuento)

   Desde main(), solicita el precio base del producto, el porcentaje de impuesto y el

porcentaje de descuento, llama al método y muestra el precio final.

Ejemplo de entrada/salida:
Ingrese el precio base del producto: 100
Ingrese el impuesto en porcentaje (Ejemplo: 10 para 10%): 10
Ingrese el descuento en porcentaje (Ejemplo: 5 para 5%): 5
El precio final del producto es: 105.0

```
run:
Ingrese el precio base del producto: 100
Ingrese el impuesto en porcentaje (Ejemplo: 10 para 10%): 10
Ingrese el descuento en porcentaje (Ejemplo: 5 para 5%): 5
El precio final del producto es: 105.0
BUILD SUCCESSFUL (total time: 15 seconds)
```

9. Composición de funciones para calcular costo de envío y total de compra.

    a. calcularCostoEnvio(double peso, String zona):
    Calcula el costo de envío basado en la zona de envío (Nacional o Internacional) y el peso del paquete.
    Nacional: $5 por kg
    Internacional: $10 por kg
    b. calcularTotalCompra(double precioProducto, double costoEnvio):
    Usa calcularCostoEnvio para sumar el costo del producto con el costo de envío.

    Desde main(), solicita el peso del paquete, la zona de envío y el precio del producto. Luego, muestra el total a pagar.

    Ejemplo de entrada/salida:
    Ingrese el precio del producto: 50
    Ingrese el peso del paquete en kg: 2
    Ingrese la zona de envío (Nacional/Internacional): Nacional
    El costo de envío es: 10.0
    El total a pagar es: 60.0

```
run:
Ingrese el precio del producto: 50
Ingrese el peso del paquete en kg: 2
Ingrese la zona de envio (Nacional/Internacional): Nacional
El costo de envio es: 10.0
El total a pagar es: 60.0
BUILD SUCCESSFUL (total time: 23 seconds)
```

10. Actualización de stock a partir de venta y recepción de productos.

    Crea un método actualizarStock(int stockActual, int cantidadVendida, int cantidadRecibida), que calcule el nuevo stock después de una venta y recepción de productos:

    NuevoStock = StockActual − CantidadVendida + CantidadRecibida

NuevoStock = CantidadVendida + CantidadRecibida

Desde main(), solicita al usuario el stock actual, la cantidad vendida y la cantidad recibida, y muestra el stock actualizado.

Ejemplo de entrada/salida:
Ingrese el stock actual del producto: 50
Ingrese la cantidad vendida: 20
Ingrese la cantidad recibida: 30
El nuevo stock del producto es: 60

```
run:
Ingrese el stock actual del producto: 50
Ingrese la cantidad vendida: 20
Ingrese la cantidad recibida: 30
El nuevo stock del producto es: 60
BUILD SUCCESSFUL (total time: 14 seconds)
```

11. Cálculo de descuento especial usando variable global.

    Declara una variable global Ejemplo de entrada/salida: = 0.10. Luego, crea un método calcularDescuentoEspecial(double precio) que use la variable global para calcular el descuento especial del 10%.
    Dentro del método, declara una variable local descuentoAplicado, almacena el valor del descuento y muestra el precio final con descuento.

    Ejemplo de entrada/salida:
    Ingrese el precio del producto: 200
    El descuento especial aplicado es: 20.0
    El precio final con descuento es: 180.0

```
run:
Ingrese el precio del producto: 200
El descuento especial aplicado es: 20.0
El precio final con descuento es: 180.0
BUILD SUCCESSFUL (total time: 5 seconds)
```

Arrays y Recursividad:

12. Modificación de un array de precios y visualización de resultados.

    Crea un programa que:
    a. Declare e inicialice un array con los precios de algunos productos.
    b. Muestre los valores originales de los precios.
    c. Modifique el precio de un producto específico.
    d. Muestre los valores modificados.

Salida esperada:
Precios originales:
Precio: $199.99
Precio: $299.5
Precio: $149.75
Precio: $399.0
Precio: $89.99
Precios modificados:
Precio: $199.99
Precio: $299.5
Precio: $129.99
Precio: $399.0
Precio: $89.99

```
run:
Precios originales:
Precio: $199.99
Precio: $299.5
Precio: $149.75
Precio: $399.0
Precio: $89.99

Precios modificados:
Precio: $199.99
Precio: $299.5
Precio: $129.99
Precio: $399.0
Precio: $89.99
BUILD SUCCESSFUL (total time: 0 seconds)
```

Conceptos Clave Aplicados:
✔ Uso de arrays (double[]) para almacenar valores.
✔ Recorrido del array con for-each para mostrar valores.
✔ Modificación de un valor en un array mediante un índice.
✔ Reimpresión del array después de la modificación.

13. Impresión recursiva de arrays antes y después de modificar un elemento.

Crea un programa que:
   a. Declare e inicialice un array con los precios de algunos productos.
   b. Use una función recursiva para mostrar los precios originales.
   c. Modifique el precio de un producto específico.
   d. Use otra función recursiva para mostrar los valores modificados.

Salida esperada:
Precios originales:
Precio: $199.99
Precio: $299.5
Precio: $149.75

Precio: $399.0
Precio: $89.99
Precios modificados:
Precio: $199.99
Precio: $299.5
Precio: $129.99
Precio: $399.0
Precio: $89.99

```
run:
Precios originales:
Precio: $199.99
Precio: $299.5
Precio: $149.75
Precio: $399.0
Precio: $89.99

Precios modificados:
Precio: $199.99
Precio: $299.5
Precio: $129.99
Precio: $399.0
Precio: $89.99
BUILD SUCCESSFUL (total time: 0 seconds)
```

Conceptos Clave Aplicados:
✔ Uso de arrays (double[]) para almacenar valores.
✔ Recorrido del array con una función recursiva en lugar de un bucle.
✔ Modificación de un valor en un array mediante un índice.
✔ Uso de un índice como parámetro en la recursión para recorrer el array.