## 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.
Ingrese un año: 1900
El año 1900 no es bisiesto.
package tp2;
import java.util.Scanner;*/

```
package tp2;

import java.util.Scanner;



public class E1_AnioBisiesto {
   public static void main(String[] args) {
      Scanner input = new Scanner(System.in);

      System.out.print("Ingrese un año: ");
      int anio = input.nextInt();

      if ((anio % 4 == 0 && anio % 100 != 0) || (anio % 400 == 0)) {
         System.out.println("El año " + anio + " es bisiesto.");
      } else {
         System.out.println("El año " + anio + " no es bisiesto.");
      }


   }
}
```

/*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 */

```
package tp2;


import java.util.Scanner;
public class E2_MayorNumero{
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
```

```
        System.out.print("Ingrese el primer número: ");
        int num1 = scanner.nextInt();

        System.out.print("Ingrese el segundo número: ");
        int num2 = scanner.nextInt();

        System.out.print("Ingrese el tercer número: ");
        int num3 = scanner.nextInt();

        int mayor = num1; // Asumimos que el primer número es el mayor inicialmente

        if (num2 > mayor) {
            mayor = num2; // Actualizamos si el segundo número es mayor
        }
        if (num3 > mayor) {
            mayor = num3; // Actualizamos si el tercer número es mayor
        }

        System.out.println("El mayor es: " + mayor);
    }
}
```

/*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.
Ingrese su edad: 10
Eres un Niño.
*/

```
package tp2;

import java.util.Scanner;


public class E3_ClasificacionEdad {

    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        System.out.print("Ingrese su edad: ");
        int edad = input.nextInt();

        if (edad < 12) {
            System.out.println("Eres un Niño.");
        } else if (edad >= 12 && edad <= 17) {
            System.out.println("Eres un Adolescente.");
        } else if (edad >= 18 && edad <= 59) {
            System.out.println("Eres un Adulto.");
```

```
        } else {
            System.out.println("Eres un Adulto mayor.");
        }
    }
}
```

```
/*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
 */
package tp2;

import java.util.Scanner;


public class E4_CalculadoraDescuento {

    public static void main(String[] args) {
        // Creamos el objeto Scanner para leer entradas del usuario
        Scanner input = new Scanner(System.in);

        // Pedimos al usuario el precio del producto
        System.out.print("Ingrese el precio del producto: ");
        double precio = input.nextDouble();  // nextDouble() lee un número decimal
        input.nextLine(); // Consumimos el salto de línea

        // Pedimos al usuario la categoría
        System.out.print("Ingrese la categoría del producto (A, B o C): ");
        String categoria = input.nextLine().toUpperCase();
        // .toUpperCase() convierte la letra a mayúscula para no distinguir entre "a" o "A"

        // Variable donde guardamos el porcentaje de descuento
        double descuento = 0;

        // Usamos switch para decidir el porcentaje según la categoría
        switch (categoria) {
            case "A":
                descuento = 0.10; // 10% de descuento
                break;
            case "B":
                descuento = 0.15; // 15% de descuento
                break;
            case "C":
                descuento = 0.20; // 20% de descuento
```

```
            break;
        default:
            // Si el usuario ingresa algo distinto de A, B o C
            System.out.println("Categoría no válida.");
            return; // Salimos del programa
    }

    // Se calcula el monto de descuento y el precio final
    double montoDescuento = precio * descuento;
    double precioFinal = precio - montoDescuento;

    // Mostramos resultados
    System.out.printf("Descuento aplicado: %.0f%%\n", descuento * 100);
    System.out.printf("Precio final: %.2f\n", precioFinal);
    //%.0f%% muestra el descuento como entero ( 15%)
    //%.2f muestra el precio con 2 decimales (850.00)
  }
}
```

## 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
 */
package tp2;

import java.util.Scanner;

public class E5_SumaDeNumeros {


  public static void main(String[] args) {
    Scanner input = new Scanner(System.in); // Se crea el objeto Scanner para leer la entrada del usuario
    int numero; // Variable para almacenar el número ingresado
    int sumaPares = 0; // Acumulador para la suma de los números pares

    do {
      System.out.print("Ingrese un número (0 para terminar): ");
      numero = input.nextInt(); // Se lee el número ingresado por el usuario

      // Si el número es par y distinto de cero, se suma al acumulador
      if (numero % 2 == 0 && numero != 0) {
        sumaPares += numero;
```

```
        }
    } while (numero != 0); // El ciclo continúa hasta que el usuario ingresa 0

    // Se muestra la suma total de los números pares ingresados
    System.out.println("La suma de los números pares es: " + sumaPares);
  }

}
```

```
/*
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
 */
package tp2;

import java.util.Scanner;


public class E6_ContadorPosNegCero {

  public static void main(String[] args) {
    Scanner input = new Scanner(System.in); // Se crea el objeto Scanner para leer la entrada del usuario
    int positivos = 0, negativos = 0, ceros = 0; // Contadores para positivos, negativos y ceros

    // Bucle para pedir 10 números al usuario
    for (int i = 1; i <= 10; i++) {
      System.out.print("Ingrese el número " + i + ": ");
      int numero = input.nextInt(); // Se lee el número ingresado

      // Se verifica si el número es positivo, negativo o cero y se incrementa el contador correspondiente
      if (numero > 0) {
        positivos++;
      } else if (numero < 0) {
        negativos++;
      } else {
        ceros++;
      }
```

```
        }
        // Se muestran los resultados
        System.out.println("Resultados:");
        System.out.println("Positivos: " + positivos);
        System.out.println("Negativos: " + negativos);
        System.out.println("Ceros: " + ceros);


    }

}



/*
 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.
 */
package tp2;

import java.util.Scanner;

public class E7_ValidacionDeNota {

  public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int nota; // Variable para almacenar la nota ingresada

        // Bucle do-while para pedir la nota hasta que sea válida (entre 0 y 10)
        do {
            System.out.print("Ingrese una nota (0-10): ");
            nota = scanner.nextInt(); // Se lee la nota ingresada

            // Si la nota no está en el rango válido, se muestra un mensaje de error
            if (nota < 0 || nota > 10) {
                System.out.println("Error: Nota inválida. Ingrese una nota entre 0 y 10.");
            }
        } while (nota < 0 || nota > 10); // El ciclo continúa mientras la nota sea inválida

        // Mensaje de confirmación cuando la nota es válida
        System.out.println("Nota guardada correctamente.");

    }
}
```

## 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
 */
package tp2;

import java.util.Scanner;


public class E8_CalculoPrecio {

    // Método que calcula el precio final
    public static double calcularPrecioFinal(double precioBase, double impuesto, double descuento) {
        // Convertimos los porcentajes a decimales (ej: 10% → 0.10)
        double impuestoDecimal = impuesto / 100;
        double descuentoDecimal = descuento / 100;

        // Fórmula del precio final
        double precioFinal = precioBase + (precioBase * impuestoDecimal) - (precioBase * descuentoDecimal);

        return precioFinal;
    }

    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        // Pedimos el precio base
        System.out.print("Ingrese el precio base del producto: ");
        double precioBase = input.nextDouble();

        // Pedimos impuesto en %
        System.out.print("Ingrese el impuesto en porcentaje (Ej: 10 para 10%): ");
        double impuesto = input.nextDouble();

        // Pedimos descuento en %
        System.out.print("Ingrese el descuento en porcentaje (Ej: 5 para 5%): ");
        double descuento = input.nextDouble();

        // Llamamos al método para calcular
        double resultado = calcularPrecioFinal(precioBase, impuesto, descuento);
```

```
        // Mostramos el resultado
        System.out.printf("El precio final del producto es: %.2f\n", resultado);
    }
}
```

```
/*
 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
 */
package tp2;

import java.util.Scanner;


public class E9_CalculoEnvioCompra {


public static void main(String[] args) {
        Scanner input = new Scanner(System.in); // Se crea el objeto Scanner para leer la entrada del usuario

        // Solicita el precio del producto
        System.out.print("Ingrese el precio del producto: ");
        double precioProducto = input.nextDouble();

        // Solicita el peso del paquete
        System.out.print("Ingrese el peso del paquete en kg: ");
        double peso = input.nextDouble();
        input.nextLine(); // Consumir la nueva línea pendiente

        // Solicita la zona de envío
        System.out.print("Ingrese la zona de envío (Nacional/Internacional): ");
        String zona = input.nextLine();

        // Calcula el costo de envío usando la función calcularCostoEnvio
        double costoEnvio = calcularCostoEnvio(peso, zona);

        // Calcula el total de la compra sumando el precio del producto y el costo de
        //envío, utiliza funcion calcularTotalCompra
        double totalCompra = calcularTotalCompra(precioProducto, costoEnvio);

        // Muestra los resultados al usuario
        System.out.println("El costo de envío es: " + costoEnvio);
```

```
        System.out.println("El total a pagar es: " + totalCompra);


    }

    // Función que calcula el costo de envío según la zona y el peso
    public static double calcularCostoEnvio(double peso, String zona) {
        if (zona.equalsIgnoreCase("Nacional")) {
            return 5 * peso; // Nacional: $5 por kg
        } else if (zona.equalsIgnoreCase("Internacional")) {
            return 10 * peso; // Internacional: $10 por kg
        } else {
            // Si la zona no es válida, se informa y se asume costo 0
            System.out.println("Zona de envío no válida. Asumiendo costo de envío 0.");
            return 0;
        }
    }

    // Función que suma el precio del producto y el costo de envío
    public static double calcularTotalCompra(double precioProducto, double costoEnvio) {
        return precioProducto + costoEnvio;
    }
}




/*
 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
 */
package tp2;

import java.util.Scanner;


public class E10_Stock {

    // Funcion que calcula el stock actualizado
    public static int actualizarStock(int stockActual, int cantidadVendida, int cantidadRecibida) {
        int nuevoStock = stockActual - cantidadVendida + cantidadRecibida;
        return nuevoStock;
    }

    public static void main(String[] args) {
```

```
        Scanner input = new Scanner(System.in);

        // Pedimos los datos al usuario
        System.out.print("Ingrese el stock actual del producto: ");
        int stockActual = input.nextInt();

        System.out.print("Ingrese la cantidad vendida: ");
        int cantidadVendida = input.nextInt();

        System.out.print("Ingrese la cantidad recibida: ");
        int cantidadRecibida = input.nextInt();

        // Llamamos a la funcion para calcular
        int resultado = actualizarStock(stockActual, cantidadVendida, cantidadRecibida);

        // Mostramos el resultado
        System.out.println("El nuevo stock del producto es: " + resultado);

    }
}
```

```
/*
 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
 */
package tp2;

import java.util.Scanner;


public class E11_VariableGlobal {


//Variable global
    static final double DESCUENTO_ESPECIAL = 0.10;

    // Funcion que aplica el descuento especial
    public static void calcularDescuentoEspecial(double precio) {
        // Variable local dentro del método
        double descuentoAplicado = precio * DESCUENTO_ESPECIAL;
        double precioFinal = precio - descuentoAplicado;

        // Mostrar resultados
        System.out.println("El descuento especial aplicado es: " + descuentoAplicado);
        System.out.println("El precio final con descuento es: " + precioFinal);
```

```
    }

    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        // 3. Pedir precio al usuario
        System.out.print("Ingrese el precio del producto: ");
        double precio = input.nextDouble();

        // Llamar a la funcion
        calcularDescuentoEspecial(precio);


    }
}
```

## 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
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.
 */
package tp2;


public class E12_ModArray {

public static void main(String[] args) {
        // Declaramos e inicializamos un array de precios
        double[] precios = {199.99, 299.50, 149.75, 399.00, 89.99};
```

```
        // Mostramos los valores originales
        System.out.println("Precios originales:");
        for (double precio : precios) { // for-each recorre cada valor del array
            System.out.println("Precio: $" + precio);
        }

        // Modificamos el precio de un producto específico
        // Por ejemplo, cambiamos el precio en la posición 2 (tercer elemento)
        precios[2] = 129.99;

        // Mostramos los valores modificados
        System.out.println("Precios modificados:");
        for (double precio : precios) {
            System.out.println("Precio: $" + precio);
        }
    }
}
```

/*
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
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.
*/

```
package tp2;


public class E13_ImprRecursiva {

// Función recursiva que imprime "Precio" desde el índice i hasta el final
    public static void imprimirRecursivo(double[] precios, int i) {
        // Caso base: si i llegó al final, terminar
        if (i == precios.length) return;
```

```
        // Trabajo actual (imprimir el elemento i)
        System.out.println("Precio: $" + precios[i]);

        // Paso recursivo: avanzar al siguiente índice
        imprimirRecursivo(precios, i + 1);
    }

    public static void main(String[] args) {
        // a) Declarar e inicializar el array de precios
        double[] precios = {199.99, 299.5, 149.75, 399.0, 89.99};

        // b) Mostrar precios originales (recursivo)
        System.out.println("Precios originales:");
        imprimirRecursivo(precios, 0);

        // c) Modificar el precio de un producto específico (tercer elemento índice 2)
        precios[2] = 129.99;

        // d) Mostrar precios modificados (recursivo)
        System.out.println("Precios modificados:");
        imprimirRecursivo(precios, 0);
    }
}
```