Marina Cordero

PROGRAMACIÓN II Trabajo Práctico 2:

Programación Estructurada

1. Verificación de Año Bisiesto.

```java
import java.util.Scanner;

public class AñoBisiesto {

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

        //Pedir al usuario que ingrese un año
        System.out.print("Introduce un año: ");
        int anio = Integer.parseInt(input.nextLine());

        //determinar si es bisiesto llamando a la función esbisiesto
        if (esbisiesto(anio)){
            System.out.println(anio + " es bisiesto");
        }
        else
            System.out.println(anio + " no es bisisesto.");

    }
    //función para determinar si un año es bisiesto
    public static boolean esbisiesto(int anio){
        return (anio % 4 == 0 && anio % 100 != 0) || (anio % 400 == 0);
    }
```

añobisiesto.AñoBisiesto > esbisiesto >

otifications | Output - AñoBisiesto (run) ×

```
run:
Introduce un a□o: 2024
2024 es bisiesto
BUILD SUCCESSFUL (total time: 2 seconds)
```

```
run:
Introduce un a□o: 1900
1900 no es bisisesto.
BUILD SUCCESSFUL (total time: 2 seconds)
```

Enlace repositorio Git Hub: https://github.com/Marigi84/TP-II-JavaEstructurada-ProgramacionII/blob/master/a%C3%B1oBisiesto/src/a%C3%B1obisiesto/A%C3%B1oBisiesto.java

2. Determinar el Mayor de Tres Números.

```java
public class Mayor {

    public static void main(String[] args) {
        Scanner input = new Scanner (System.in);
        ArrayList<Integer> numeros = new ArrayList<>();// Se crea una lista vacía para almacenar los números.

        System.out.print("Ingrese 3 número enteros: ");
        // Bucle para pedir y guardar tres números del usuario
        for(int i = 0; i < 3; i++){
            System.out.print("Introduce el número " + (i + 1) + ":");
            numeros.add(input.nextInt());// Se lee el número y se añade al final de la lista
        }  //se llama a la función encontrarMayor para lugo imprimir por pantalla en número mayor
        System.out.println("El número mayor de la lista es: " + encontrarMayor(numeros));
    }
    // se crea función encuentra el número más grande en una lista de números enteros.
    public static int encontrarMayor(ArrayList<Integer>numeros){ //recibe por parametro una lista de números
        if(numeros == null || numeros.isEmpty()){
            System.out.println("La lista no puede estar vacía, ingresa una que contenga numeros enteros:");
        }//se inicializa una variable mayor con el primer elemento de la lista
        int mayor = numeros.get(0);
        //civlo for para comparar cada elemento de la lista y guardarlo en la variabe mayor
        for (int i = 1; i < numeros.size(); i++){
            if (numeros.get(i) > mayor){
                mayor = numeros.get(i);
            }
        }return mayor;
    }
```

mayor.Mayor › encontrarMayor › for (int i = 1; i < numeros.size(); i++) › if (numeros.get(i) > mayor) ›

utput - Mayor (run) #2 ×

```
run:
Ingrese 3 número enteros: Introduce el número 1:8
Introduce el número 2:12
Introduce el número 3:5
El número mayor de la lista es: 12
```

Enlace repositorio Git Hub: https://github.com/Marigi84/TP-II-JavaEstructurada-ProgramacionII/blob/master/Mayor/src/mayor/Mayor.java

3. Clasificación de Edad.

```java
import java.util.Scanner;

public class ClasificarEdad {
    public static void main(String[] args) {
        //Creamos un objeto scanner para leer la entrada del usuario
        Scanner input = new Scanner(System.in);

        //solicitamos al usuari su edad
        System.out.print("Por favor ingrese su edad: ");
        int edad = input.nextInt();

        //inicializamos variable para la etapa de la vida
        String etapaVida;

        if (edad < 12){
            etapaVida = "Niño.";
        }else if (edad >= 12 && edad <= 17){
            etapaVida = "Adolecente.";
        }else if (edad >=18 && edad <= 59){
            etapaVida = "Adulto.";
        }else
            etapaVida = "Adulto Mayor.";

        //mostramos por consola
        System.out.println("Usted se encuentra en la etapa de la vida: " + etapaVida);

        input.close();
```

clasificaredad.ClasificarEdad > main > if (edad < 12) else if (edad >= 12 && edad <= 17) else if (edad >= 18 &

utput ×

Mayor (run) #2 × | ClasificarEdad (run) ×

```
run:
Por favor ingrese su edad: 15
Usted se encuentra en la etapa de la vida: Adolecente.
BUILD SUCCESSFUL (total time: 2 seconds)
```

Enlace repositorio Git Hub: https://github.com/Marigi84/TP-II-JavaEstructurada-ProgramacionII/blob/master/ClasificarEdad/src/clasificaredad/ClasificarEdad.java

4. Calculadora de Descuento según categoría.

```java
import java.util.Scanner;

public class CalculadoraDescuento {
    public static void main(String[] args) {
        //creamos objeto scanner para leer la entrada del usuario
        Scanner input = new Scanner(System.in);

        //declaramos variables
        double precio, descuento = 0, precioFinal;
        char categoria;

        //pedimos al usuario los datos y los guardamos en variables
        System.out.print("Ingrese el precio de producto:");
        precio = input.nextInt();

        System.out.println("Ingrese la categoría del producto (A, B o C): ");
        categoria = input.next().toUpperCase().charAt(0);

        //Aplicamos descuento seguon cada caso con un swich
        switch(categoria){
            case 'A':
                descuento = precio * 0.10;
                break;
            case 'B':
                descuento = precio * 0.15;
                break;
            case 'C':
                descuento = precio * 0.20;
                break;
            default:
                System.out.print("Categoría no válida para descuento.");
        }

         precioFinal = precio - descuento;

         //mostrar por consola
         System.out.println("Precio Original: $ " + precio );
         System.out.println("Decuento: $ " + descuento);
         System.out.println("El total a abonar es: $ " + precioFinal);
```

```
run:
Ingrese el precio de producto:1000
Ingrese la categoría del producto (A, B o C): a
Precio Original: $ 1000.0
Decuento: $ 100.0
El total a abonar es: $ 900.0
BUILD SUCCESSFUL (total time: 5 seconds)
```

Enlace repositorio Git Hub: https://github.com/Marigi84/TP-II-JavaEstructurada-ProgramacionII/blob/master/CalculadoraDescuento/src/calculadoradescuento/CalculadoraDescuento.java

5. Suma de Números Pares (while).

```java
import java.util.Scanner;

public class SumaPares {
    public static void main(String[] args) {
         // Inicializamos un objeto Scanner para leer la entrada del usuario
        Scanner input = new Scanner(System.in);
        //Definimos e inicializamos variables
        int sumaPares = 0, numero;

        // pedimos el primer número antes de entrar al bucle
        System.out.print("Ingrese un número (0 para terminar): ");
        numero = input.nextInt();

        // Bucle while para procesar los números
        while (numero != 0) {
            // Verificar si el número es par
            if (numero % 2 == 0) {
                // Si es par, lo sumamos a la variable sumaPares
                sumaPares += numero;
            }
            // Solicita el siguiente número dentro del bucle
            System.out.print("Ingrese un número (0 para terminar): ");
            numero = input.nextInt();
        }

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

        input.close();
```

```
ut - SumaPares (run) #2 ×
Ingrese un número (0 para terminar): 5
Ingrese un número (0 para terminar): 9
Ingrese un número (0 para terminar): 2
Ingrese un número (0 para terminar): 4
Ingrese un número (0 para terminar): 6
Ingrese un número (0 para terminar): 0
La suma de los números pares es: 12
```

Enlace repositorio Git Hub: https://github.com/Marigi84/TP-II-JavaEstructurada-ProgramacionII/blob/master/SumaPares/src/sumapares/SumaPares.java

6. Contador de Positivos, Negativos y Ceros (for).

```java
public class ContadorNumeros {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
       // declaramos e nicializamos los contadores
        int positivos = 0;
        int negativos = 0;
        int ceros = 0;

        // Usamos un bucle for para pedir 10 números
        for (int i = 0; i < 10; i++) {
            System.out.print("Ingrese el número " + (i + 1) + ": ");
            int numero = input.nextInt();

            // Verificamos si el número es positivo, negativo o cero
            if (numero > 0) {
                positivos++;
            } else if (numero < 0) {
                negativos++;
            } else {
                ceros++;
            }
        }
        input.close();

        // Imprimimos los resultados
        System.out.println("\nResultados:");
        System.out.println("Positivos: " + positivos);
        System.out.println("Negativos: " + negativos);
        System.out.println("Ceros: " + ceros);
    }
```

```
Ingrese el número 1: 56
Ingrese el número 2: 32
Ingrese el número 3: 8
Ingrese el número 4: 4
Ingrese el número 5: -25
Ingrese el número 6: 0
Ingrese el número 7: 6
Ingrese el número 8: 4
Ingrese el número 9: 8
Ingrese el número 10: 2

Resultados:
Positivos: 8
Negativos: 1
Ceros: 1
```

Enlace repositorio Git Hub: https://github.com/Marigi84/TP-II-JavaEstructurada-ProgramacionII/blob/master/ContadorNumeros/src/contadornumeros/ContadorNumeros.java

7. Validación de Nota entre 0 y 10 (do-while).

```java
public static void main(String[] args) {
    Scanner input = new Scanner(System.in);
    //declaramos variable nota
    int nota;

    do {
        System.out.print("Ingrese una nota (0-10): ");

        // Verifica si la entrada del usuario es un número entero
        while (!input.hasNextInt()) {
            // si la entrada no es un numero entero, aparecerá un mensaje de eeror y se
            //pedirá otro número.
            System.out.println("Error: Entrada inválida. Por favor, ingrese un número.");
            System.out.print("Ingrese una nota (0-10): ");
            input.next(); // Descarta la entrada no válida
        }
        nota = input.nextInt();

        if (nota < 0 || nota > 10) {
            System.out.println("Error: Nota inválida. Ingrese una nota entre 0 y 10.");
        }

    } while (nota < 0 || nota > 10);

    System.out.println("Nota guardada correctamente.");

    input.close();
```

```
t - ValidacionNota (run) ×
run:
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.
```

Enlace repositorio Git Hub: https://github.com/Marigi84/TP-II-JavaEstructurada-ProgramacionII/blob/master/ValidacionNota/src/validacionnota/ValidacionNota.java

8. Cálculo del Precio Final con impuesto y descuento.

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

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

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

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

    // Llamamos al método para calcular el precio final
    double precioFinal = calcularPrecioFinal(precioBase, impuesto, descuento);

    // Mostramos el resultado
    System.out.println("El precio final del producto es: " + precioFinal);

    input.close();
}
public static double calcularPrecioFinal(double precioBase, double impuesto, double descuento) {
    // La fórmula es: PrecioFinal = PrecioBase + (PrecioBase * Impuesto) - (PrecioBase * Descuento)
    // Convertimos los porcentajes a su valor decimal
    double impuestoDecimal = impuesto / 100.0;
    double descuentoDecimal = descuento / 100.0;

    // Calculamos el impuesto y el descuento
    double montoImpuesto = precioBase * impuestoDecimal;
    double montoDescuento = precioBase * descuentoDecimal;

    // Aplicamos la fórmula para obtener el precio final
    return precioBase + montoImpuesto - montoDescuento;
}
```

```
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: 16 seconds)
```

Enlace repositorio Git Hub: https://github.com/Marigi84/TP-II-JavaEstructurada-ProgramacionII/blob/master/CalculadoraPrecio/src/calculadoraprecio/CalculadoraPrecio.java

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

```java
import java.util.Scanner;

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

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

        System.out.print("Ingrese el peso del paquete en kg: ");
        double pesoPaquete = input.nextDouble();
        input.nextLine(); // Consumir el enter de línea

        //bandera para salir del ciclo while
        String zonaEnvio;
        boolean zonaValida = false;

        // Bucle para asegurar que el usuario ingrese una zona válida
        do {
            System.out.print("Ingrese la zona de envío (Nacional/Internacional): ");
            zonaEnvio = input.nextLine();
            if (zonaEnvio.equalsIgnoreCase("Nacional") || zonaEnvio.equalsIgnoreCase("Internacional")) {
                zonaValida = true;
            } else {
                System.out.println("Zona de envío no válida. Por favor, intente de nuevo.");
```

```java
    } while (!zonaValida);

    // Calcula el costo de envío y el total de la compra
    double costoEnvio = calcularCostoEnvio(pesoPaquete, zonaEnvio);
    double totalPagar = calcularTotalCompra(precioProducto, costoEnvio);

    // Muestra los resultados
    System.out.println("El costo de envío es: " + costoEnvio);
    System.out.println("El total a pagar es: " + totalPagar);

    input.close();
}
//método calculador precio de envio, recibe el pezo y la zona
public static double calcularCostoEnvio(double peso, String zona) {
    //equalsIgnoreCase, para no distinguir entre mayusculas y minusculas
    if (zona.equalsIgnoreCase("Nacional")) {
        return peso * 5;
    } else {
        return peso * 10;
```

```
Ingrese el precio del producto: 50
Ingrese el peso del paquete en kg: 2
Ingrese la zona de envío (Nacional/Internacional): naci
Zona de envío no válida. Por favor, intente de nuevo.
Ingrese la zona de envío (Nacional/Internacional): nacional
El costo de envío es: 10.0
El total a pagar es: 60.0
```

Enlace repositorio Git Hub: https://github.com/Marigi84/TP-II-JavaEstructurada-ProgramacionII/blob/master/CalculadoraEnvio/src/calculadoraenvio/CalculadoraEnvio.java

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

```java
package gestionstock;

import java.util.Scanner;

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

        // Solicitar al usuario los datos
        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();

        // Calcular el nuevo stock utilizando el método
        int nuevoStock = actualizarStock(stockActual, cantidadVendida, cantidadRecibida);

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

        input.close();

    }
    //metodo para calcular el nuevo stock
    public static int actualizarStock(int stockActual, int cantidadVendida, int cantidadRecibida) {
        return stockActual - cantidadVendida + cantidadRecibida;
    }
```

```
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: 15 seconds)
```

Enlace repositorio Git Hub: https://github.com/Marigi84/TP-II-JavaEstructurada-ProgramacionII/blob/master/GestionStock/src/gestionstock/GestionStock.java

11. Cálculo de descuento especial usando variable global.

```java
public class VariableGlobal {

    // Variable global para el porcentaje de descuento (10%)
    public static final double DESCUENTO_ESPECIAL = 0.10;

     // Método main para probar la funcionalidad
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        // Solicitar al usuario que ingrese el precio
        System.out.print("Ingrese el precio del producto: ");
        double precioProducto = input.nextDouble();

        // Llamar al método para calcular el descuento
        double descuentoAplicado = calcularDescuento(precioProducto);

        // Calcular el precio final
        double precioFinal = precioProducto - descuentoAplicado;

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

        input.close();
    }

public static double calcularDescuento(double precio) {
        // La variable 'descuento' es local a este método
        double descuento = precio * DESCUENTO_ESPECIAL;
        return descuento;
}
```

```
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: 2 seconds)
```

Enlace repositorio Git Hub: https://github.com/Marigi84/TP-II-JavaEstructurada-ProgramacionII/blob/master/VariableGlobal/src/variableglobal/VariableGlobal.java

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

```java
*/
package modificarprecios;
public class ModificarPrecios {
    public static void main(String[] args) {
       // declarar e inicializar un array con los precios de algunos productos.
        double[] precios = {199.99, 299.50, 149.75, 399.00, 89.99};

        //Muestra los valores originales de los precios
        System.out.println("Precios originales:");
        for (int i = 0; i < precios.length; i++) {
            System.out.println("Precio: $" + precios [i]);
        }
        // Vamos a modificar el tercer precio (índice 2)
        precios[2] = 129.99;

        //Muestra los valores modificados.
        System.out.println("\nPrecios modificados:");
        for (int i = 0; i <precios.length; i++) {
            System.out.println("Precio: $" + precios[i]);
        }
    }
```

```
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)
```

Enlace repositorio Git Hub: https://github.com/Marigi84/TP-II-JavaEstructurada-ProgramacionII/blob/master/ModificarPrecios/src/modificarprecios/ModificarPrecios.java

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

```java
package preciosrecursivos;
public class PreciosRecursivos {
    public static void main(String[] args) {
        //Declarar e inicializar un array con precios
        double[] precios = {199.99, 299.50, 149.75, 399.00, 89.99};

        // Usar la función recursiva para mostrar los precios originales
        System.out.println("Precios originales:");
        imprimirPrecios(precios, 0);

        //Modificar el precio de un producto específico (ejemplo: el tercer producto)
        int indiceAModificar = 2; // El tercer elemento tiene el índice 2
        double nuevoPrecio = 129.99;
        precios[indiceAModificar] = nuevoPrecio;

        //Usar la función recursiva para mostrar los valores modificados
        System.out.println("\nPrecios modificados:");
        imprimirPrecios(precios, 0);
    }
    // Función recursiva para imprimir el array
    public static void imprimirPrecios(double[] precios, int indice) {
        // Caso base: si el índice es igual o mayor a la longitud del array, la recursión termina
        if (indice >= precios.length) {
            return;
        }
        //Caso recursivo: imprime el precio en el índice actual y llama a la función con el siguiente índice
        System.out.println("Precio: $" + precios[indice]);
        imprimirPrecios(precios, indice + 1);
    }
```

```
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)
```

Enlace repositorio Git Hub: https://github.com/Marigi84/TP-II-JavaEstructurada-ProgramacionII/blob/master/PreciosRecursivos/src/preciosrecursivos/PreciosRecursivos.java