TECNICATURA UNIVERSITARIA
EN PROGRAMACIÓN
A DISTANCIA

PROGRAMACIÓN II
Trabajo Práctico 2: Programación Estructurada

OBJETIVO GENERAL

Desarrollar habilidades en programación estructurada en Java, abordando desde conceptos básicos como operadores y estructuras de control hasta temas avanzados como funciones, recursividad y estructuras de datos. Se busca fortalecer la capacidad de análisis y solución de problemas mediante un enfoque práctico,

MARCO TEÓRICO

Estructuras condicionales	Clasificación de edad, verificación de año bisiesto
Ciclos (for, while, do-while)	Repetición de ingreso de datos y cálculos
Funciones	Cálculo modular de descuentos, envíos, stock
Arrays	Gestión de precios de productos
Recursividad	Impresión recursiva de arrays

Caso Práctico

Desarrollar los siguientes ejercicios en Java utilizando el paradigma de programación estructurada. Agrupados según el tipo de estructuras o conceptos aplicados:

TECNICATURA UNIVERSITARIA
EN PROGRAMACIÓN
A DISTANCIA

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.

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

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.

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

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

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

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.

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

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

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

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

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.

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.

CONCLUSIONES ESPERADAS

- Aplicar estructuras de control y decisión para resolver problemas.
- Diseñar soluciones usando estructuras iterativas y condicionales.
- Modularizar el código utilizando funciones con y sin retorno.
- Utilizar arrays para almacenamiento y manipulación de datos.
- Comprender y aplicar la recursividad en casos simples.
- Trabajar con variables locales y globales de forma adecuada.
- Fortalecer la capacidad de análisis lógico y la resolución de errores.
- Consolidar el uso del lenguaje Java mediante la práctica estructurada.

1. Verificación de Año Bisiesto

```
import java.util.Scanner;

public class AnioBisiesto {

    public static void main(String[] args) {

        // Crea un objeto Scanner para leer la entrada del usuario
        Scanner scanner = new Scanner(System.in);

        // Solicita el año al usuario
        System.out.print("Ingrese un año: ");
        int anio = scanner.nextInt();

        // Lógica para determinar si el año es bisiesto
        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

```
import java.util.Scanner;

public class MayorDeTres {

    public static void main(String[] args) {

        // Crea un objeto Scanner para leer la entrada del usuario
        Scanner scanner = new Scanner(System.in);

        // Solicita los tres números al usuario
        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();

        // Lógica para determinar el mayor
        int mayor = num1; // Asumimos que el primer número es el mayor

        if (num2 > mayor) {
            mayor = num2; // Si el segundo número es mayor, actualizamos la
variable 'mayor'
        }
```

```
        if (num3 > mayor) {
            mayor = num3; // Si el tercer número es mayor, actualizamos la
variable 'mayor'
        }

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

    }
}
```

3. Clasificación de Edad

```
import java.util.Scanner;

public class ClasificacionEdad {

    public static void main(String[] args) {

        // Crea un objeto Scanner para leer la entrada del usuario
        Scanner scanner = new Scanner(System.in);

        // Solicita la edad al usuario
        System.out.print("Ingrese su edad: ");
        int edad = scanner.nextInt();

        // Lógica para clasificar la edad
        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 { // Si no se cumple ninguna de las condiciones anteriores, la
edad es 60 o más
            System.out.println("Eres un Adulto mayor.");
        }
    }
}
```

4. Calculadora de Descuento según categoría

```
import java.util.Scanner;

public class CalculadoraDescuento {

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

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

        // Solicita la categoría del producto
        System.out.print("Ingrese la categoría del producto (A, B o C): ");
        String categoria = scanner.next();
```

```
        double descuento = 0.0;
        double porcentajeDescuento = 0.0;

        // Aplica el descuento según la categoría
        if (categoria.equalsIgnoreCase("A")) {
            porcentajeDescuento = 0.10; // 10%
        } else if (categoria.equalsIgnoreCase("B")) {
            porcentajeDescuento = 0.15; // 15%
        } else if (categoria.equalsIgnoreCase("C")) {
            porcentajeDescuento = 0.20; // 20%
        } else {
            System.out.println("Categoría no válida. No se aplicará
descuento.");
            porcentajeDescuento = 0.0;
        }

        descuento = precioOriginal * porcentajeDescuento;
        double precioFinal = precioOriginal - descuento;

        // Muestra los resultados
        System.out.println("Precio original: " + precioOriginal);
        System.out.println("Descuento aplicado: " +
(int)(porcentajeDescuento * 100) + "%");
        System.out.println("Precio final: " + precioFinal);

    }
}
```

5. Suma de Números Pares (while)

```
import java.util.Scanner;

public class SumaParesWhile {

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

        int sumaPares = 0;
        int numero;

        // Bucle while que se ejecuta al menos una vez (do-while)
        // o se puede hacer con un while infinito y un break
        // o con un while que comprueba la condición antes de la primera
iteración

        System.out.print("Ingrese un número (0 para terminar): ");
        numero = scanner.nextInt();

        while (numero != 0) {
            // Verifica si el número es par
            if (numero % 2 == 0) {
                sumaPares += numero; // Suma el número si es par
            }
            // Solicita el siguiente número dentro del bucle
            System.out.print("Ingrese un número (0 para terminar): ");
```

```
            numero = scanner.nextInt();
        }

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

    }
}
```

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

```
import java.util.Scanner;

public class ContadorNumeros {

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

        int positivos = 0;
        int negativos = 0;
        int ceros = 0;

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

            // Clasifica el número y actualiza los contadores
            if (numero > 0) {
                positivos++;
            } else if (numero < 0) {
                negativos++;
            } else {
                ceros++;
            }
        }

        // Muestra los resultados finales
        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)

```
import java.util.Scanner;

public class ValidacionNota {

    public static void main(String[] args) {
        // Crea un objeto Scanner para leer la entrada del usuario
```

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

        int nota;

        // Bucle do-while para solicitar y validar la nota
        do {
            System.out.print("Ingrese una nota (0-10): ");
            nota = scanner.nextInt();

            // Mensaje de error si la nota está fuera del rango
            if (nota < 0 || nota > 10) {
                System.out.println("Error: Nota inválida. Ingrese una nota entre
0 y 10.");
            }
        } while (nota < 0 || nota > 10); // La condición evalúa si la nota es
inválida

        // Si el bucle termina, la nota es válida
        System.out.println("Nota guardada correctamente.");

    }
}
```

8. Cálculo del Precio Final con Impuesto y Descuento

```
import java.util.Scanner;

public class CalculadoraPrecioFinal {

    // Método para calcular el precio final del producto
    public static double calcularPrecioFinal(double precioBase, double
impuesto, double descuento) {
        // Convertimos los porcentajes a decimales
        double impuestoDecimal = impuesto / 100;
        double descuentoDecimal = descuento / 100;

        // Fórmula para el precio final
        double precioFinal = precioBase + (precioBase * impuestoDecimal) -
(precioBase * descuentoDecimal);
        return precioFinal;
    }

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

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

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

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

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

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

    }
}
```

9. Composición de Funciones para Calcular Costo de Envío y Total de Compra

```
import java.util.Scanner;

public class CalculadoraCompra {

    // Método 1: Calcula el costo de envío
    public static double calcularCostoEnvio(double peso, String zona) {
        double costoEnvio = 0;
        if (zona.equalsIgnoreCase("Nacional")) {
            costoEnvio = peso * 5;
        } else if (zona.equalsIgnoreCase("Internacional")) {
            costoEnvio = peso * 10;
        } else {
            System.out.println("Zona no válida. El costo de envío se calculará
como 0.");
        }
        return costoEnvio;
    }

    // Método 2: Calcula el total de la compra usando el costo de envío
    public static double calcularTotalCompra(double precioProducto,
double costoEnvio) {
        return precioProducto + costoEnvio;
    }

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

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

        System.out.print("Ingrese el peso del paquete en kg: ");
        double peso = scanner.nextDouble();

        System.out.print("Ingrese la zona de envío (Nacional/Internacional):
");
        String zona = scanner.next();

        // Llama al primer método para obtener el costo de envío
        double costoEnvio = calcularCostoEnvio(peso, zona);

        // Muestra el costo de envío
```

```
        System.out.println("El costo de envío es: " + costoEnvio);

        // Llama al segundo método para calcular el total de la compra
        double totalPagar = calcularTotalCompra(precioProducto,
costoEnvio);

        // Muestra el resultado final
        System.out.println("El total a pagar es: " + totalPagar);

    }
}
```

10. Actualización de Stock a partir de Venta y Recepción

```
import java.util.Scanner;

public class GestionStock {

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

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

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

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

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

        // Llama al método para calcular el nuevo stock
        int nuevoStock = actualizarStock(stockActual, cantidadVendida,
cantidadRecibida);

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

    }
}
```

11. Cálculo de Descuento Especial Usando Variable Global

```
import java.util.Scanner;

public class CalculadoraDescuentoEspecial {

    // Variable global (de clase) para el descuento especial
    private static final double DESCUENTO_ESPECIAL = 0.10;
```

```
    // Método para calcular el descuento y el precio final
    public static void calcularDescuentoEspecial(double precio) {
        // Variable local dentro del método
        double descuentoAplicado = precio * DESCUENTO_ESPECIAL;
        double precioFinal = precio - descuentoAplicado;

        // Muestra los 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 scanner = new Scanner(System.in);

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

        // Llama al método para aplicar el descuento
        calcularDescuentoEspecial(precioProducto);

    }
}
```

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

```
public class ModificarArray {

    public static void main(String[] args) {
        // a. Declarar e inicializar un array con los precios
        double[] precios = {199.99, 299.50, 149.75, 399.00, 89.99};

        // b. Mostrar los valores originales de los precios
        System.out.println("Precios originales:");
        for (double precio : precios) {
            System.out.println("Precio: $" + precio);
        }

        // c. Modificar el precio de un producto específico (ejemplo: el tercer
producto)
        // El tercer elemento está en el índice 2
        precios[2] = 129.99;

        // d. Mostrar los valores modificados
        System.out.println("\nPrecios modificados:");
        for (double precio : precios) {
            System.out.println("Precio: $" + precio);
        }
    }
}
```

13. Impresión Recursiva de Arrays

```
public class ArrayRecursivo {

    // Método recursivo para imprimir los elementos del array
    public static void imprimirArrayRecursivo(double[] array, int indice) {
        // Caso base: si el índice es igual a la longitud del array, la recursión
termina
        if (indice < array.length) {
            System.out.println("Precio: $" + array[indice]);
            // Llamada recursiva para el siguiente elemento
            imprimirArrayRecursivo(array, indice + 1);
        }
    }

    public static void main(String[] args) {
        // a. Declarar e inicializar un array con los precios
        double[] precios = {199.99, 299.50, 149.75, 399.00, 89.99};

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

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

        System.out.println("\nPrecios modificados:");
        // d. Usar la misma función recursiva para mostrar los valores
modificados
        imprimirArrayRecursivo(precios, 0);
    }
}
```

Autor: Federico Iacono