

Sesión Sincrónica Nº 2

Fundamentos de Programación (PRY2201)

Profesor: Miguel Puebla

Experiencia 1: Creando Mi Primer Algoritmo

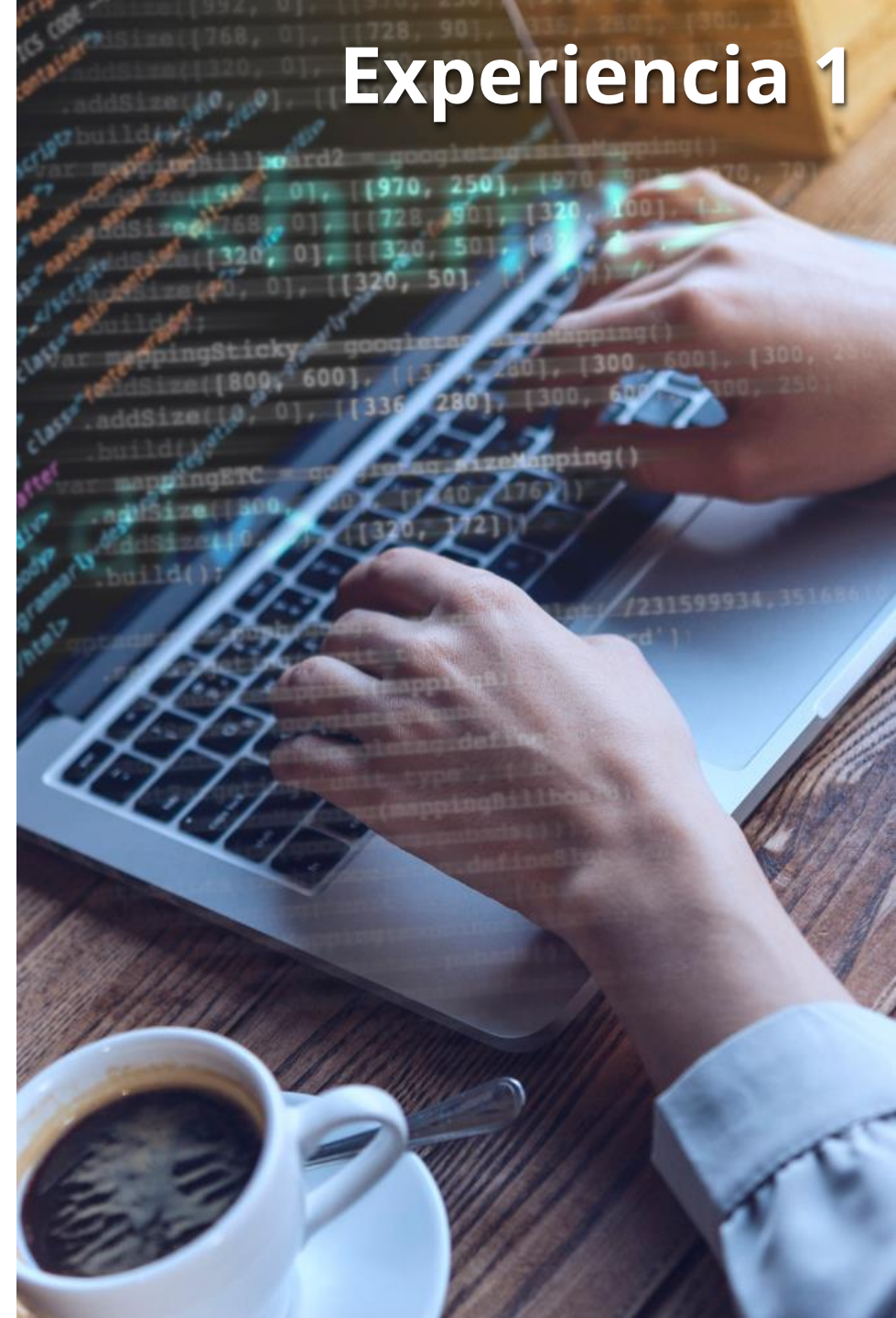
RA1. Utiliza estrategias de abstracción para la construcción de algoritmo, aplicando pseudocódigo con el objetivo de dar solución a problemáticas planteadas.

Semana 2: Diseñando y modelando Algoritmos



IL1. Identifica estrategias de abstracción y tipos de algoritmos aplicables en la construcción de algoritmos.

IL2. Aplica pseudocódigo en la implementación de algoritmos, brindando una solución a las problemáticas planteadas.

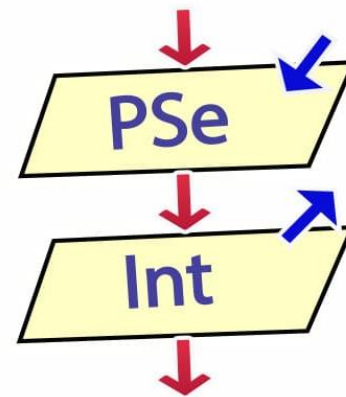


Semana 2



Conocimientos Generales

Algoritmos Estructurados	Estructuras de Control	Pseudocódigo
Resolución de Problemas	Sintaxis de Java	Eficiencia de Algoritmos



Algoritmos Estructurados

Definición

Un algoritmo estructurado organiza el flujo de control en una secuencia clara y lógica, utilizando estructuras como secuencias, decisiones y repeticiones.

Características:

- **Secuencialidad:** las instrucciones se ejecutan en el orden en que aparecen.
- **Modularidad:** se pueden dividir en subalgoritmos o funciones.
- **Claridad:** facilitan la lectura y el mantenimiento.

Ejemplo en PSeint: Problema: Calcular el promedio de tres números:

Algoritmo CalcularPromedioTresNumeros

// Declaración de variables

Definir num1, num2, num3, promedio **Como** Real;

// Solicitar los tres números

Escribir "Ingrese el primer número:";

Leer num1;

Escribir "Ingrese el segundo número:";

Leer num2;

Escribir "Ingrese el tercer número:";

Leer num3;

// Calcular el promedio

promedio = (num1 + num2 + num3) / 3;

// Mostrar el promedio

Escribir "El promedio de los tres números es:", promedio;

FinAlgoritmo



Algoritmos Estructurados

Ejemplo Completo en Pseint con secuencia, decisiones y repeticiones : Calcular Promedio de Calificaciones

```
1  Algoritmo CalculoPromedio
2      // Declaración de variables
3      Definir n, i, calificacion, suma, promedio, calif_referencia, contador Como Real;
4
5      // Secuencia: Solicitar el número de calificaciones
6      Escribir "Ingrese el número de calificaciones:";
7      Leer n;
8
9      suma = 0;
10     contador = 0;
11     calif_referencia = 7;
12
13     // Repetición: Leer cada calificación y sumarla
14     Para i = 1 Hasta n Con Paso 1 Hacer
15         Escribir "Ingrese la calificación ", i, ":";
16         Leer calificacion;
17         suma = suma + calificacion;
18
19         // Decisión: Contar cuántas calificaciones están por encima de la calificación de referencia
20         Si calificacion > calif_referencia Entonces
21             contador = contador + 1;
22         FinSi
23     FinPara
24
25     // Secuencia: Calcular el promedio
26     promedio = suma / n;
27
28     // Decisión: Determinar si el promedio es aprobatorio
29     Escribir "Ingrese la calificación de referencia para comparación (Debe ser menos de 7):";
30     Leer calif_referencia;
31
32     Si promedio ≥ calif_referencia Entonces
33         Escribir "El promedio es: ", promedio, ". Aprobado.";
34     SiNo
35         Escribir "El promedio es: ", promedio, ". Reprobado.";
36     FinSi
37
38     // Secuencia: Mostrar el número de calificaciones por encima de la referencia
39     Escribir "Número de calificaciones por encima de ", calif_referencia, ": ", contador;
40
41 FinAlgoritmo
```




Estructuras de Control

Definición

Son mecanismos que permiten alterar el flujo de ejecución de un algoritmo.

Tipos:


- **Condicionales:** Permiten ejecutar código basado en una condición (Si...Entonces...SiNo...FinSi en PSeint, if...else en Java).
- **Bucles:** Permiten repetir bloques de código (Mientras...Hacer, Para...Hasta en PSeint; while, for en Java).
 - Bucle **Para** (For): Se utiliza cuando se conoce el número exacto de iteraciones.
 - Bucle **Mientras** (While): Se utiliza cuando el número de iteraciones depende de una condición que se evalúa antes de cada repetición.

Estructuras de Control

- Ejemplo Bucle Para (For)

PSeint:

pseint

 Copiar código

```
Algoritmo BuclePara
    // Declarar la variable
    Definir i Como Entero;

    // Imprimir los números del 1 al 5
    Para i = 1 Hasta 5 Con Paso 1 Hacer
        Escribir "Número:", i;
    FinPara
FinAlgoritmo
```

Java:

java

 Copiar código


```
public class BuclePara {
    public static void main(String[] args) {
        // Imprimir los números del 1 al 5
        for (int i = 1; i <= 5; i++) {
            System.out.println("Número: " + i);
        }
    }
}
```

Estructuras de Control

- Ejemplo Bucle **Mientras** (While)

PSeint:

pseint


 Copiar código

```
Algoritmo BucleMientras
    // Imprimir los números del 1 al 5
    Definir i Como Entero;
    i = 1;

    Mientras i <= 5 Hacer
        Escribir "Número:", i;
        i = i + 1;
    FinMientras
FinAlgoritmo
```

Java:

java

 Copiar código

```
public class BucleMientras {
    public static void main(String[] args) {
        // Imprimir los números del 1 al 5
        int i = 1;

        while (i <= 5) {
            System.out.println("Número: " + i);
            i++;
        }
    }
}
```


Pseudocódigo

Definición:

Es una descripción detallada y abstracta de un algoritmo, que no requiere una sintaxis estricta de un lenguaje de programación.

Ejemplo en PSeint: Venta de Entradas de un Teatro: En el ejemplo de la venta de entradas, Real es la elección adecuada porque estamos tratando con precios, que pueden tener decimales. Si utilizáramos Entero, perderíamos la capacidad de representar fracciones de unidades monetarias.

```
1  Algoritmo VentaEntradasTeatro
2      // Declaración de variables
3      Definir cantidadEntradas, precioEntrada, totalVenta Como Real;
4      Definir i Como Entero;
5
6      // Solicitar el número de entradas
7      Escribir "Ingrese la cantidad de entradas vendidas:";
8      Leer cantidadEntradas;
9
10     // Solicitar el precio de una entrada
11     Escribir "Ingrese el precio de una entrada:";
12     Leer precioEntrada;
13
14     totalVenta = 0;
15
16     // Repetición: Calcular el total de la venta
17     Para i = 1 Hasta cantidadEntradas Con Paso 1 Hacer
18         totalVenta = totalVenta + precioEntrada;
19     FinPara
20
21     // Mostrar el total de la venta
22     Escribir "El total de la venta es:", totalVenta;
23
24 FinAlgoritmo
```

Resolución de Problemas

Definición:

Proceso de diseñar un algoritmo para resolver un problema específico, desde la comprensión del problema hasta la implementación y pruebas.

Problema: Calcular el máximo común divisor (MCD) de dos números.

Ejemplo En PSeint:

```
1  Algoritmo CalcularMCD
2      Definir a, b, temp Como Entero;
3      Escribir "Ingrese el primer número:";
4      Leer a;
5      Escribir "Ingrese el segundo número:";
6      Leer b;
7
8      Mientras b ≠ 0 Hacer
9          temp = b;
10         b = a % b;
11         a = temp;
12      FinMientras
13
14      Escribir "El MCD es: ", a;
15  FinAlgoritmo
16
```


Resolución de Problemas

Problema: Calcular el máximo común divisor (MCD) de dos números.

Ejemplo En Java:

java

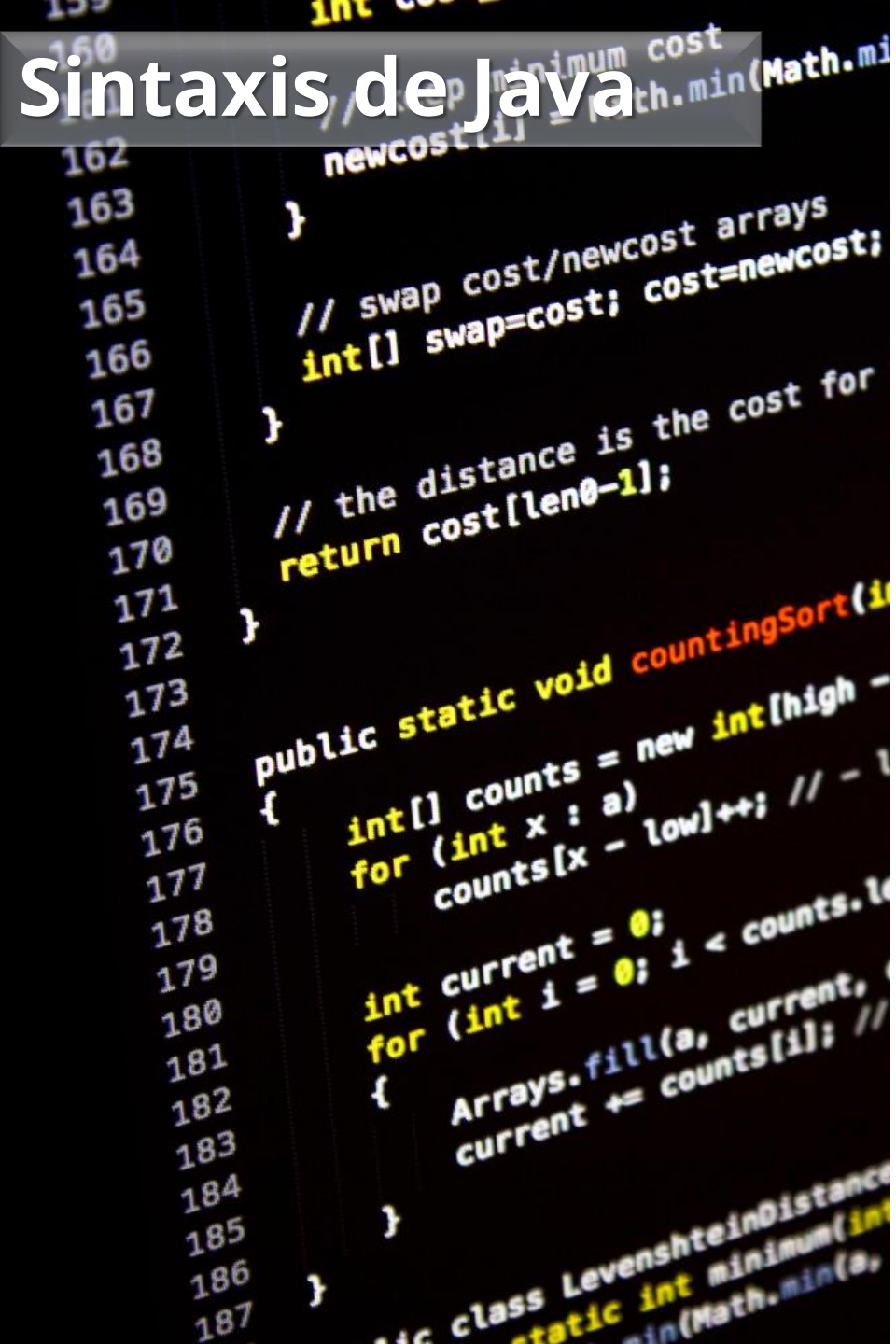
 Copiar código

```
import java.util.Scanner;

public class CalcularMCD {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Ingresa el primer número:");
        int a = sc.nextInt();
        System.out.println("Ingresa el segundo número:");
        int b = sc.nextInt();

        while (b != 0) {
            int temp = b;
            b = a % b;
            a = temp;
        }

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

Sintaxis de Java

Definición

La sintaxis en Java define cómo se deben estructurar y escribir los programas. Incluye:

Clases y Métodos:

Clase: Estructura básica del código (ej. `public class MiClase { }`).

Método main: Punto de entrada del programa (`public static void main(String[] args)`).

Variables:

Declaración: Especificar tipo y nombre (`int numero;`).

Inicialización: Asignar valores (`numero = 10;`).

Estructuras de Control:

Condicionales: `if`, `else if`, `else`.

Bucles: `for`, `while`, `do-while`.

Métodos:

Definición: Bloques de código que realizan tareas (`public int sumar(int a, int b) { return a + b; }`).

Modificadores de Acceso:

`public`, `private`, `protected`: Controlan la visibilidad de clases, métodos y atributos.

Comentarios:

En Línea: `// comentario`.

Multilínea: `/* comentario */`.

Sintaxis de Java

```

159 // keep minimum cost
160 newcost[i] = Math.min(Math.min(
161     // swap cost/newcost arrays
162     int[] swap=cost; cost=newcost;
163     // the distance is the cost for
164     return cost[len0-1];
165 }
166
167 public static void countingSort(int[] a)
168 {
169     int[] counts = new int[high - low + 1];
170     for (int x : a)
171         counts[x - low]++; // - low to get index
172     int current = 0;
173     for (int i = 0; i < counts.length; i++)
174     {
175         Arrays.fill(a, current, current + counts[i], current);
176         current += counts[i];
177     }
178 }
179
180 public class LevenshteinDistance
181 {
182     static int minimum(int a, int b)
183     {
184         return Math.min(a, b);
185     }
186 }
187

```

java

 Copiar código

```
// Clase principal que contiene el método principal (main) para ejecutar el programa
public class CalculadoraPromedio {

    // Método principal, punto de entrada del programa
    public static void main(String[] args) {
        // Crear una instancia de la clase PromedioCalculator
        PromedioCalculator calculadora = new PromedioCalculator(85.5, 90.0, 78.0);

        // Calcular el promedio y mostrarlo
        System.out.println("El promedio de los números es: " + calculadora.calcularPromedio());
    }
}

// Clase que realiza el cálculo del promedio
class PromedioCalculator {
    // Atributos para almacenar los números
    private double numero1;
    private double numero2;
    private double numero3;

    // Constructor para inicializar los números
    public PromedioCalculator(double numero1, double numero2, double numero3) {
        this.numero1 = numero1;
        this.numero2 = numero2;
        this.numero3 = numero3;
    }

    // Método para calcular el promedio de los tres números
    public double calcularPromedio() {
        return (numero1 + numero2 + numero3) / 3;
    }
}
```

Eficiencia de Algoritmos



Definición

La **eficiencia de un algoritmo** se refiere a cuán bien utiliza los recursos computacionales, principalmente el tiempo y la memoria, para resolver un problema

Objetivo: Elegir algoritmos que resuelvan problemas de manera eficiente para grandes volúmenes de datos y optimizar el rendimiento general del sistema.

Eficiencia de Algoritmos



```
1  Algoritmo BusquedaLineal
2      Definir arr, i, n, encontrado Como Entero;
3      Dimensionar arr[5];
4
5      encontrado = 0;
6
7      Escribir "Ingrese el número a buscar:";
8      Leer n;
9
10     arr[0]=2;
11     arr[1]=4;
12     arr[2]=5;
13     arr[3]=8;
14     arr[4]=6;
15
16     Para i = 0 Hasta 4 Con Paso 1 Hacer
17         Si arr[i] = n Entonces
18             Escribir i;
19             encontrado = 1;
20         FinSi
21     FinPara
22
23     Si encontrado = 1 Entonces
24         Escribir "Número encontrado en el arreglo.";
25     SiNo
26         Escribir "Número no encontrado.";
27     FinSi
28 FinAlgoritmo
29
```

Eficiencia de Algoritmos



```
public class BusquedaLineal {  
    public static void main(String[] args) {  
        int[] arr = {3, 1, 4, 1, 5};  
        boolean encontrado = false;  
  
        Scanner sc = new Scanner(System.in);  
        System.out.println("Ingrese el número a buscar:");  
        int n = sc.nextInt();  
  
        for (int i = 0; i < arr.length; i++) {  
            if (arr[i] == n) {  
                encontrado = true;  
                break;  
            }  
        }  
  
        if (encontrado) {  
            System.out.println("Número encontrado en el arreglo.");  
        } else {  
            System.out.println("Número no encontrado.");  
        }  
    }  
}
```


Lecturas



- Capítulo 4: Making the most variables and their values

Burd, B. (2022). Java for Dummies. New Jersey: John Wiley & Sons.

https://webezproxy.duoc.cl/login?url=http://biblioteca.duoc.cl/bdigital/elibros/a50163-Java_fordummies/68/ Páginas 57 a 95

- Capítulo 1: Introducción a Java

Vegas Gertrudix, J. M. (2022). Java 17: Fundamentos prácticos de programación. Bogotá:

Ediciones de la U.

https://webezproxy.duoc.cl/login?url=http://biblioteca.duoc.cl/bdigital/elibros/a50229-Java_17/36/

Páginas 40 a 57

¡Muchas Gracias!



DuocUC[®] 
ONLINE