

# UD 1 - 1.1.1

## Pseudocódigo



# Índice 1

- Introducción al Pseudocódigo
  - Definición
- 1. Características del Pseudocódigo
  - Rasgos principales
- 2. Elementos del Pseudocódigo
  - Instrucciones básicas

# Índice 2

- 2. Elementos del Pseudocódigo
  - Más instrucciones
- 2.1 Inicio y Fin
  - Estructura básica
- 2.2 Variables
  - Concepto y uso

# Índice 3

- 2.2 Variables
  - Ejemplo con variables
- 2.3 Estructuras de Control
  - Tipos principales
- 2.3.1 Secuenciales
  - Definición

# Índice 4

- 2.3.2 Condicionales
  - Condicional simple
  - Condicional doble
  - Condicional múltiple
  - Condicionales anidados

# Índice 5

- 2.3.3 Iterativas
  - Bucle Mientras
  - Ejemplo Mientras
  - Bucle Para
  - Ejemplo Para
- Actividades y Ejercicios

# Índice 6

- Actividades y Ejercicios
  - Actividad 1 - Comparar dos números
  - Actividad 2 - Relación entre dos números
  - Actividad 3 - Serie descendente
  - Actividad 4 - Serie descendente con Para
  - Ejercicio 1 - Validar rango



# Índice 7

- Actividades y Ejercicios
  - Solución
  - Ejercicio 2 - Serie entre dos números
  - Ejercicio 3 - Ordenar tres números
- ¡Gracias por su atención!

# Introducción al Pseudocódigo

# Definición

- Forma de representar algoritmos de forma simple.
- Fácilmente entendible por cualquier persona.
- No depende de la formación en programación.
- Lenguaje simplificado, con estructuras básicas.
- Se basa en la programación estructurada.

# 1. Características del Pseudocódigo

# Rasgos principales

- Lenguaje cercano a la programación.
- Objetivo: algoritmos fáciles de entender.
- Independiente del lenguaje que se usará después.
- Usa expresiones limitadas, sin sintaxis rígida.
- Debe comenzar en un único punto y terminar.
- Siempre debe tener solución finita.

## 2. Elementos del Pseudocódigo

# Instrucciones básicas

- Inicio
- Fin
- Escribe "texto"
- Lee variable
- Si ... entonces ...
- Si ... entonces ... Sino ...
- Según valor ... opciones ...

# Más instrucciones

- Mientras (condición) hacer ...
- Para X en (1...N) hacer ...
- Operadores matemáticos: +, -, \*, /, //, %
- Operadores relacionales: ==, >, <, >=, <=, !=
- Operadores lógicos: and, or, not
- Asignación con el símbolo =



# 2.1 Inicio y Fin

# Estructura básica

- Todo algoritmo empieza con `Inicio`.
- Todo algoritmo termina con `Fin`.
- Delimita el bloque principal de ejecución.
- Facilita la lectura y comprensión.

```
Inicio
  Escribe "¿Cómo te llamas?"
  Lee nombre
  Escribe "Hola, " + nombre
Fin
```

## 2.2 Variables

# Concepto y uso

- Contenedores de información con nombre.
- Tipos: cadenas ("texto"), números (5, 3.2), lógicos (verdadero/falso).
- Su valor puede cambiar en el algoritmo.
- El tipo se define implícitamente al asignar.
- Concatenación con + para unir textos y valores.

# Ejemplo con variables

```
Inicio
    num1 = 10
    Escribe "Introduce un número: "
    Lee num2

    suma = num1 + num2
    Escribe "La suma es " + suma

    iva = 0.21
    Escribe "Introduce un precio: "
    Lee precio

    Escribe "Precio con IVA: " + (precio * iva)
Fin
```

## 2.3 Estructuras de Control

# Tipos principales

- Secuenciales: instrucciones en orden.
- Condicionales: permiten tomar decisiones.
- Iterativas: repiten un bloque de código.
- Basadas en programación estructurada.
- Simples y claras en pseudocódigo.

## 2.3.1 Secuenciales



# Definición

- Ejecutan instrucciones en orden de aparición.
- Cada instrucción ocurre tras la anterior.
- Delimitadas entre `Inicio` y `Fin`.
- Base de cualquier programa.

```
Inicio
  Instrucción1
  Instrucción2
  Instrucción3
Fin
```

## 2.3.2 Condicionales

# Condicional simple

- Ejecuta un bloque solo si se cumple una condición.
- Si no se cumple, no ejecuta nada.

```
Si (condición) entonces  
    Instrucción1  
    Instrucción2  
Fin
```

# Condicional doble

- Añade bloque alternativo si la condición no se cumple.
- Permite elegir entre dos caminos.

```
Si (condición) entonces
    Instrucción1
Sino
    Instrucción2
```

# Condicional múltiple

- Evalúa un valor con varias opciones.
- Cada opción tiene sus instrucciones.

```
Según valor entonces
```

```
  opcion1: Instrucciones
```

```
  opcion2: Instrucciones
```

# Condicionales anidados

- Condiciones dentro de otras condiciones.
- Permiten mayor detalle y control.

```
Si (condición1) entonces
    Instrucciones1
Sino
    Si (condición2) entonces
        Instrucciones2
```

## 2.3.3 Iterativas

# Bucle Mientras

- Repite mientras la condición sea verdadera.
- Evalúa condición antes de ejecutar.
- Riesgo de bucles infinitos si no cambia la condición.

```
Mientras (condición) hacer  
    Instrucción1  
    Instrucción2  
Fin
```



# Ejemplo Mientras

```
Mientras (cont > 0) hacer  
    Escribe cont  
    cont = cont - 1
```

# Bucle Para

- Ejecuta bloque un número fijo de veces.
- Usa variable de control (i).
- Puede ser ascendente o descendente.

```
Para i en (1..N) hacer  
  Instrucciones
```

# Ejemplo Para

```
Inicio
  suma = 0
  Para i en (1..10) hacer
    suma = suma + i
  Escribe "La suma es " + suma
Fin
```

# Actividades y Ejercicios

# Actividad 1 - Comparar dos números

- Leer dos números.
- Mostrar cuál de los dos es mayor.

```
Inicio
  Lee num1
  Lee num2
  Si (num1 > num2) entonces
    Escribe num1 + " es mayor que " + num2
  Sino
    Escribe num2 + " es mayor que " + num1
Fin
```

# Actividad 2 - Relación entre dos números

- Leer dos números.
- Mostrar si son iguales, mayor o menor.

```
Inicio
  Lee num1
  Lee num2
  Si (num1 == num2) entonces
    Escribe "Son iguales"
  Sino
    Si (num1 > num2) entonces
      Escribe num1 + " es mayor"
    Sino
      Escribe num2 + " es mayor"
Fin
```

# Actividad 3 - Serie descendente

- Leer un número mayor que 0.
- Mostrar la serie decreciendo hasta 0.

```
Inicio
  Lee num
  Si (num > 0) entonces
    Mientras (num >= 0) hacer
      Escribe num
      num = num - 1
Fin
```

# Actividad 4 - Serie descendente con Para

- Repetir el ejercicio anterior.
- Usar el bucle `Para`.

```
Inicio
  Lee num
  Si (num > 0) entonces
    Para i en (num..0) hacer
      Escribe i
Fin
```



# Ejercicio 1 - Validar rango

- Leer un número entre 1 y 10.
- Repetir hasta que sea válido.

# Solución

```
Inicio
  Lee num
  Mientras (num < 1 or num > 10) hacer
    Escribe "Inténtalo otra vez!"
    Lee num
Escribe "Correcto!"
Fin
```

# Ejercicio 2 - Serie entre dos números

- Leer dos números.
- Crear la serie que los une.

Ejemplo:

```
4 y 8 → 4 5 6 7 8  
12 y 3 → 3 4 5 ... 12
```

Note: Buen ejercicio para usar condicionales y bucles.  
Permite pensar si el segundo número es mayor o menor.

# Ejercicio 3 - Ordenar tres números

- Leer tres números.
- Mostrar en orden ascendente

# Dudas



# ¡Gracias por su atención!



