




Supuesto Didáctico

CFGs DAM – Módulo: Programación de procesos y servicios

 Proyecto didáctico basado en la simulación por software de un microprocesador elemental.

2.1 ¿Qué supuesto queremos trabajar?

Simulación en Java de un **microprocesador elemental**, como herramienta para:

- Reforzar el uso de **bucles y condicionales**
- Diseñar y utilizar **funciones con paso de parámetros**
- Comprender el control de flujo de instrucciones tipo ensamblador

2.2 Contextualización del alumnado

- 1.º Curso de **Desarrollo de Aplicaciones Multiplataforma (DAM)**
- Grupo de 25 alumnos/as
- Nivel medio de programación estructurada
- En proceso de consolidar conceptos clave como:
 - Modularización
 - Variables y estructuras de control
 - Comprensión de lógica de ejecución

2.3 Conocimientos previos requeridos






✓ Para abordar el supuesto, el alumnado debe haber trabajado:

- Entrada y salida en consola (`Scanner` , `System.out`)
- Condicionales (`if` , `else` , `switch`)
- Bucles (`for` , `while` , `do-while`)
- Funciones/métodos en Java (definición, llamadas, retorno)

2.4 Objetivos de aprendizaje

- Comprender cómo simular instrucciones básicas con código
- Utilizar funciones de forma modular para dividir responsabilidades
- Aplicar correctamente estructuras de control (`while` , `if` , etc.)
- Analizar y representar la **ejecución paso a paso** de un proceso secuencial

2.5 Metodología

-  **Aprendizaje basado en retos:** construir un simulador paso a paso
-  Trabajo individual con apoyo en clase
-  División modular por fases
-  Evaluación continua y formativa
-  Enfoque funcional: se programa para que "haga algo real"

2.6 Material didáctico y diseño DUA

 Recursos aplicados con criterios de **Diseño Universal para el Aprendizaje (DUA)**:

- Código base inicial con ejemplos comentados
- Guía escrita + visual (diagramas)
- Plantilla de tabla de ejecución paso a paso
- Esquemas en papel y digital
- Rúbrica entregada desde el inicio
- Posibilidad de grabar vídeo explicativo como alternativa



2.7 Secuencia de acciones formativas

1. Introducción conceptual
2. Presentación del enunciado paso a paso
3. Análisis conjunto de un ejemplo
4. Resolución guiada de parte del código
5. Desarrollo individual con soporte del docente
6. Entrega final + presentación de resultados

2.8 Actividad principal

Objetivo:

Simular instrucciones tipo ensamblador (`LOAD` , `ADD` , `JMP` , `PRINT` , `HALT`) sobre una memoria muy simple.

Pasos a desarrollar:

- Crear una lista de instrucciones con sintaxis simplificada
- Usar `while` para recorrer instrucciones hasta `HALT`
- Definir funciones individuales (`load()` , `add()` , etc.)
- Usar un **registro acumulador** para los valores
- Imprimir el estado tras cada instrucción → análisis paso a paso

 Cada instrucción se interpreta como si se estuviera ejecutando en una CPU básica



2.9 Evaluación: Instrumentos y criterios




Criterios de evaluación

- Uso correcto de **bucles y estructuras de control**
- Diseño **modular y funcional**
- Lógica del simulador coherente
- Claridad en la representación paso a paso
- Presentación y limpieza del código

Instrumentos de evaluación

- Rúbrica detallada (por criterios)
- Revisión individual del código
- Ejecución en pantalla y análisis de resultados
- Autoevaluación o reflexión final (opcional)

2.10 Inclusión y atención a la diversidad

 Medidas aplicadas:

- Código base parcial para alumnado con más dificultades
- Comentarios orientativos y estructura inicial guiada
- Feedback individual durante el desarrollo
- Ejemplo base comentado en clase
- Alternativas de entrega: PDF con análisis + código o vídeo explicativo

 Flexibilidad sin reducir el nivel de exigencia cognitiva

2.11 Actividades de ampliación

Para alumnado avanzado:

- Añadir nuevas instrucciones: SUB , IFZERO , STORE , LOADM
- Simular una **memoria con celdas** (int[])
- Incluir control de errores (instrucciones mal escritas, saltos inválidos)
- Añadir opción de entrada desde archivo o menú interactivo

✅ Finalidad del supuesto

💡 Este supuesto permite:

- ✓ Consolidar el uso de bucles y funciones
 - ✓ Comprender la lógica interna de una CPU
 - ✓ Fomentar pensamiento algorítmico
 - ✓ Introducir nociones de arquitectura de computadoras de forma práctica
- 🎯 ¡Un simulador educativo que convierte código en concepto!