🎓 Supuesto Didáctico

1.º DAM – Módulo: Programación

“Diseña, enlaza y da vida a tus datos”

2.1 ¿Qué supuesto queremos trabajar?

Diseñar e implementar estructuras de datos dinámicas en C, con uso modular, gestión explícita de memoria, punteros y punteros a funciones.

> Se validará técnicamente con herramientas como `assert`, `Valgrind` y, opcionalmente, se añadirá visualización gráfica básica (consola o GUI).

2.2 Contextualización del alumnado

- Curso: 1.º DAM
- Módulo: Programación
- Etapa del curso: intermedio – ya conocen arrays, funciones y estructuras
- Grupo de nivel medio con buena predisposición práctica
- Familiarizados con lógica, pero con poca experiencia en C

2.3 Conocimientos previos requeridos

- Uso de `struct`, `typedef`, funciones
- Declaración y uso básico de punteros
- Entrada y salida estándar (`fgets`, `printf`)
- Compilación con `gcc` y uso de `make`
- Conceptos básicos de depuración

2.4 Objetivos de aprendizaje

- Comprender y utilizar punteros en estructuras dinámicas
- Modularizar código: separación en `.h` y `.c`
- Aplicar punteros a funciones en menús y diseño extensible
- Validar y depurar con herramientas profesionales
- Expresar gráficamente estructuras dinámicas (opcional)

2.5 Metodología

- Activa y práctica, centrada en el desarrollo incremental
- Aprendizaje basado en proyectos
- Evaluación continua a través del desarrollo funcional
- Trabajo en parejas o pequeños grupos
- Reflexión crítica sobre diseño y eficiencia

2.6 Material didáctico (DUA)

- Código base comentado
- Diagramas de estructuras y llamadas
- Ejemplos interactivos guiados
- Alternativas visuales: consola (`ncurses`) o GUI (`SDL` , `Qt`)
- Recursos accesibles: repositorio en GitHub, plantillas de Makefile

2.7 Secuencia de acciones formativas

1. Introducción a punteros y referencias
2. Implementación de lista/pila con `malloc` y `free`
3. Separación modular del código
4. Menú interactivo con array de punteros a funciones
5. Persistencia básica: guardar/cargar desde fichero
6. Validación con `assert`, `Valgrind`, `gdb`
7. Preparación de presentación técnica
8. Defensa y evaluación final

2.8 Actividad principal

🧠 “Misión: estructura viva”

🔧 Objetivo:
Construir una biblioteca funcional y modular de estructuras dinámicas (listas, pilas, árboles).

💡 Requisitos:

- Modularidad: `.c` + `.h`
- Menú dinámico con punteros a funciones
- Operaciones: alta, baja, listar, buscar
- Persistencia de datos en fichero
- Validación con `assert`, depuración con `Valgrind`
- Documentación técnica y presentación

2.9 Evaluación – Instrumentos y criterios

Instrumentos

- Observación directa del desarrollo
- Presentación técnica
- Defensa oral
- Pruebas con casos reales
- Rúbrica detallada

Criterios

- Correcta gestión de memoria
- Uso adecuado de punteros y estructuras
- Modularidad, limpieza y organización del código
- Funcionalidad completa

2.10 Inclusión y atención a la diversidad

- Código base accesible para todos
- Apoyo visual (diagrama de memoria, árbol, nodos…)
- Retroalimentación frecuente y tutorías
- Alternativas a visualización gráfica si hay barreras técnicas

2.11 Actividades de ampliación

- Implementar estructura doblemente enlazada o árbol AVL
- Visualización animada de inserciones y recorridos
- Añadir exportación/importación en JSON o binario
- Integración con interfaz gráfica o consola avanzada
- Simulación visual tipo “debugger educativo”

✅ Conclusión

Este supuesto permite al alumnado **consolidar punteros**, mejorar su **modularidad** y **visualizar estructuras dinámicas**, aplicando programación estructurada con visión de eficiencia, mantenimiento y claridad técnica.