

Tarea extraordinaria

Entornos de desarrollo

Curso DAM 2025/26

Índice

Actividad 1: Ciclo de vida.....	2
Actividad 2: IDEs.....	2
Actividad 3: Casos de pruebas.....	2
Actividad 4: Refactorización.....	3
Actividad 5: Relaciones entre clases.....	3
Actividad 6: Caso de uso.....	4

Actividad 1: Ciclo de vida

(1,70 puntos)

Explica en qué consiste el ciclo de vida de un proyecto software según el modelo en cascada realimentado.

Debes desarrollar obligatoriamente los siguientes puntos:

- Nombra todas las fases del modelo en orden cronológico y explica qué se hace en cada una con tus palabras
- Indica qué producto o resultado se obtiene al finalizar cada fase (por ejemplo, documentación, código, etc.).
- Define qué se entiende por “realimentado” en este modelo y qué aporta frente al modelo clásico.
- Menciona dos situaciones concretas en las que tendría sentido volver a una fase anterior.

Actividad 2: IDEs

(1,70 puntos)

Compara los entornos de desarrollo NetBeans y Visual Studio en los siguientes aspectos. Rellena la tabla y explica lo que sea necesario:

- Tipo de licencia: ¿es libre, propietario o mixto? ¿Qué implicaciones tiene?
- Lenguajes principales que soporta por defecto
- Sistemas operativos en los que se puede instalar y usar sin problemas
- Ventajas concretas que ofrece cada uno al desarrollador
- Indica en qué tipo de proyecto recomendarías cada una:
 - Una aplicación de escritorio en Java
 - Una aplicación web basada en .NET y SQL Server para entorno Windows

Actividad 3: Casos de pruebas

(1,70 puntos)

Estás evaluando el siguiente código que forma parte de una función bancaria:

```
if (saldo >= cantidad) {  
    saldo = saldo - cantidad;  
    return "Operación completada";  
}else {  
    return "Fondos insuficientes";  
}
```

Responde lo siguiente:

- a) ¿Cuántos caminos de ejecución diferentes hay y qué condiciones los activan?
- b) Propón tres casos de prueba con valores de `saldo` y `cantidad` + el resultado esperado
- c) ¿Qué tipo de técnica de prueba estás utilizando (caja blanca o negra)?
- d) Justifica por qué es esa técnica y no la otra
- e) Proporciona un caso de prueba en el límite que podría fallar si la condición se hubiese programado como `saldo > cantidad` en vez de `>=`.

Actividad 4: Refactorización

(1,70 puntos)

Analiza las siguientes acciones de mejora en código Java. Di si son o no refactorización, qué técnica concreta se aplicar (si procede) y para qué sirve:

- a) Cambiar el nombre de una variable `p` por `precioConDescuento`.
- b) Sustituir un valor `0.21` por una constante `IVA_GENERAL`.
- c) Añadir comentarios línea por línea en el método `calcularTotal()`.
- d) Dividir el método `realizarCompra()` (60 líneas) en tres: `verificarStock()`, `generalPedido()` y `registrarPago()`.

Finalmente responde:

- e) ¿Qué ventaja general tiene la refactorización frente a simplemente documentar?

Actividad 5: Relaciones entre clases

(1,70 puntos)

Responde a las siguientes cuestiones explicando bien los conceptos antes de aplicarlos:

- a) ¿Qué es la herencia en programación orientada a objetos? Da un ejemplo realista de dos clases relacionadas por herencia
- b) ¿Qué es la composición? ¿En qué se diferencia de la agregación? Pon ejemplos claros.
- c) En una aplicación de tienda online, ¿qué tipo de relación existe entre las clases `Pedido` y `DetallePedido`? Justifica el tipo de relación (composición, agregación, etc.) con claridad.

Actividad 6: Caso de uso

(1,60 puntos)

A partir del siguiente escenario, completa la ficha de caso de uso y analiza su utilidad:

“Un usuario registrado accede al sistema, consulta sus pedidos, selecciona uno y descarga el albarán en PDF. El sistema registra la acción y marca el pedido como “completado” si era el último paso pendiente.

- a) Completa esta ficha del caso de uso:
 - i) Nombre del caso de uso
 - ii) Actor principal
 - iii) Precondición
 - iv) Flujo normal (en 4-6 pasos)
 - v) Postcondición
- b) Explica para qué sirve este tipo de modelado (caso de uso) durante el análisis de un sistema y qué ventajas aporta frente a una descripción en texto libre.