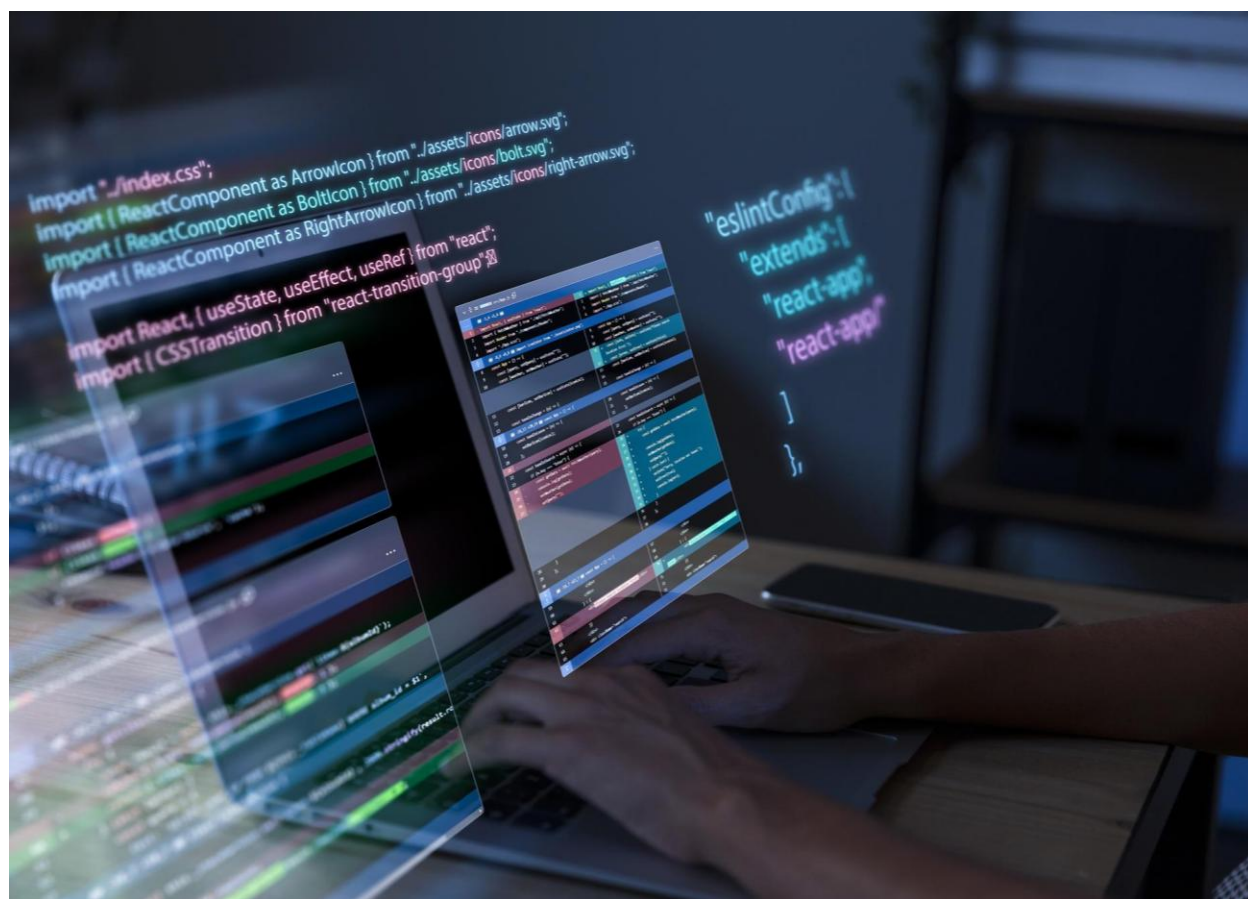


# Ejercicios

## POO (II)

---



Autor: Marcela Martín

Fecha: diciembre 2025

**Ejercicio 9.** Crea una clase llamada Estudiante con su constructor.

- Define los atributos nombre, nota1, nota2 y nota3.
- Crea un método para calcular el promedio de las tres notas.
- En la APP, crea un objeto de Estudiante, asigna valores a las notas y muestra su promedio.

**Ejercicio 10.** Añade un constructor sobrecargado a la clase Estudiante implementada en el ejercicio anterior, reutilizando el constructor anterior que tiene como parámetros de entrada todos los atributos de la clase.

**Ejercicio 11.** Diseñar la clase CuentaCorriente, que almacena los datos: DNI y nombre del titular, así como el saldo. Las operaciones típicas con una cuenta corriente son:

- Crear una cuenta: se necesita el DNI y nombre del titular. El saldo inicial será 0.
- Sacar dinero: el método debe indicar si ha sido posible llevar a cabo la operación, si existe saldo suficiente.
- Ingresar dinero: se incrementa el saldo.
- Mostrar información: muestra la información disponible de la cuenta corriente.

**Ejercicio 12.** En la clase CuentaCorriente del ejercicio anterior, sobrecarga los constructores para poder crear objetos:

- Con el DNI del titular de la cuenta y un saldo inicial.
- Con el DNI, nombre y el saldo inicial.

Utiliza el constructor genérico `this()`.

**Ejercicio 13.** Modificar la visibilidad de la clase CuentaCorriente para que sea visible desde clases externas y la visibilidad de sus atributos para que:

- saldo no sea visible para otras clases.
- nombre sea público para cualquier clase.

- dni solo sea visible por clases vecinas.

Añade en la APP la comprobación de la visibilidad de los atributos.

**Ejercicio 14.** Todas las cuentas corrientes con las que se va a trabajar pertenecen al mismo banco. Añade un atributo que almacene el nombre del banco (que es único) en la clase Cuenta-Corriente. Diseña un método que permita recuperar y modificar el nombre del banco (al que pertenecen todas las cuentas corrientes)

**Ejercicio 15.** Existen gestores que administran las cuentas bancarias y atienden a sus propietarios. Cada cuenta, en caso de tenerlo, cuenta con un único gestor. Diseña la clase Gestor de la que interesa guardar su nombre, teléfono y el importe máximo autorizado con el que pueden operar.

Con respecto a los gestores, existen las siguientes restricciones:

- Un gestor tendrá siempre un nombre y un teléfono.
- Si no se asigna, el importe máximo autorizado por operación será de 10000 euros.
- Un gestor, una vez asignado, no podrá cambiar su número de teléfono. Y todo el mundo podrá consultarlo.
- El nombre será público y el importe máximo solo será visible por clases vecinas. Modifica la clase CuentaCorriente para que pueda disponer de un objeto Gestor.

Escribe los métodos necesarios.

**Ejercicio 16.** Crea un proyecto/paquete que incluya los códigos fuente de los ejercicios anteriores. Cambia la visibilidad de los atributos de todas las clases a privado, y añade los métodos set/get para cada uno.

1. Adapta las clases a la nueva estructura de packages.