**Nombre Alumno/s**: Fernandez Alejandro Facundo **DNI:** 40047821

Gomez Angela Rebeca **DNI:** 40488634

**Nombre Profesor:** Aldo Mettini, Andrea Airaldi

**Grupo Laboratorio:** 3 **TP:** 2 **Fecha de entrega:** 25/08/2025

**Clases:** Una clase es una representación de un concepto del mundo real, que permite modelar atributos y comportamientos. En este TP definimos clases como Persona, Laboratorio, Cliente, Alumno y Empleado, cada una con sus respectivos atributos y métodos.

**Abstracción:** Permite representar lo esencial de una entidad, omitiendo los detalles innecesarios. Cada clase creada abstrae un objeto o concepto real, definiendo sus características relevantes.

**Encapsulamiento:** Es la técnica que permite proteger los datos internos del objeto. Se realiza declarando atributos como private y permitiendo el acceso mediante métodos públicos (getters y setters). Esto garantiza que los datos no puedan ser modificados directamente desde el exterior.

**Ocultamiento de la información:** Relacionado al encapsulamiento, permite que la implementación interna de un objeto no sea visible desde fuera. Solo se puede acceder a la interfaz pública de la clase. En este TP, seguimos los diagramas UML que indicaban qué atributos y métodos debían ser públicos o privados.

**Visibilidad:** Se implementaron atributos y métodos con distintos niveles de acceso (public y private), según lo indicado en los diagramas UML.

**Métodos constructores:** Son métodos especiales que permiten inicializar objetos. Se implementaron constructores simples y también constructores sobrecargados, con distintos parámetros para permitir múltiples formas de instanciar un objeto.

**Instanciación de objetos:** Los objetos se crean con el operador new. Se utilizaron clases ejecutables donde se instancian objetos con datos constantes, por teclado (Scanner) o como parámetros del método main.

**Métodos de acceso (getters y setters):** Permiten consultar y modificar los valores de los atributos encapsulados. Los métodos get devuelven el valor de un atributo, mientras que los set lo modifican.

**Interacción entre objetos:** Los objetos se comunican entre sí mediante el envío de mensajes (llamadas a métodos). En los ejercicios, esto se observa cuando una clase ejecutable instancia un objeto y le solicita que realice una acción.

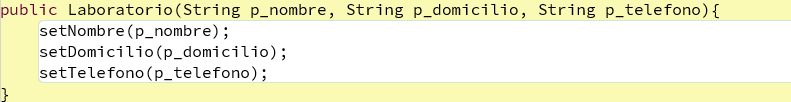
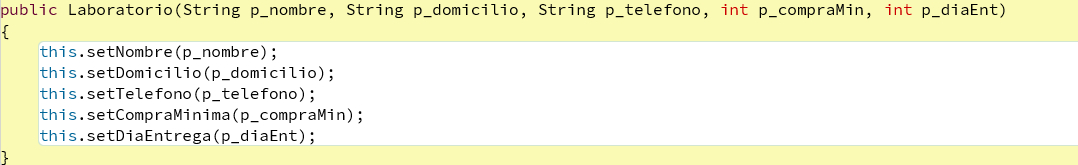
**Sobrecarga de métodos:** Un mismo método puede tener diferentes definiciones, siempre que se diferencien por tipo o cantidad de parámetros. Se utilizó la sobrecarga de constructores, especialmente en la clase Laboratorio.

**Uso de UML:** Los diagramas de clases en UML sirvieron como guía para definir la estructura de las clases, los tipos de atributos, los métodos, su visibilidad y el comportamiento esperado. Se respetó la firma indicada y se completaron los métodos get y set aunque no estuvieran explícitos.

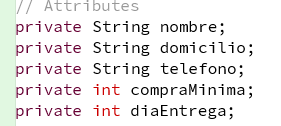
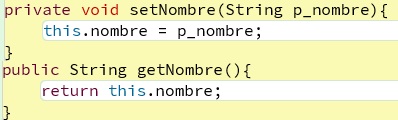
**Documentación (javadoc):** Se comenzó a documentar el código utilizando la herramienta javadoc para mejorar la legibilidad, comprensión y mantenibilidad del mismo.

**Ejemplos:**

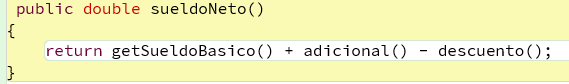
Constructores y sobrecarga:

****

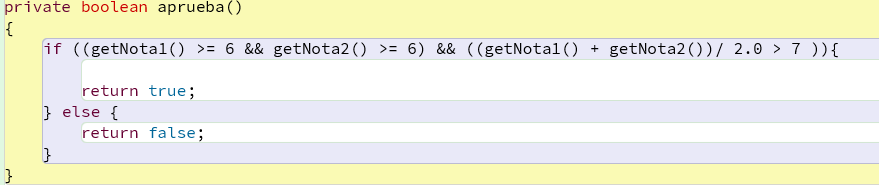
Encapsulamiento (atributos privados + getters/setters):

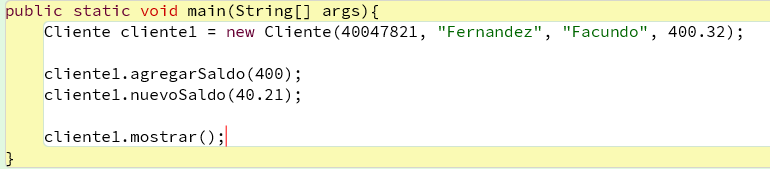
Métodos (comportamiento de objetos):



**Abstracción:**

****

Instanciación y uso de objetos:

****

Sobrecarga de constructores y comportamiento:

