Sistema de Gestión de la Movilidad Sostenible de una Ciudad

Práctica de la asignatura Programación Orientada a Objetos
Escenario para el curso 2024-2025 - Febrero de 2025 - Versión 1.4

Departamento de Lenguajes y Sistemas Informáticos
Escuela Técnica Superior de Ingeniería Informática - UNED

1. Introducción

Los objetivos que se plantean en la realización de esta práctica son los siguientes:

- Familiarización con la Programación Orientada a Objetos (POO): definición de clases e instancias, uso de la herencia, definición/uso de métodos estáticos y abstractos.
- Realización del diseño orientado a objetos de un problema.
- Implementación de un programa sencillo donde se manejen conceptos relacionados con POO.

La práctica se va a implementar en Java 2 Estándar Edition (J2SE). El compilador de Java que se usará será BlueJ, tal y como se define en el programa de la asignatura.

2. Programación Orientada a Objetos en Java

El paradigma de programación orientada a objetos define un programa como una colección de entidades que se relacionan para resolver un problema. Estas entidades, que se conocen genéricamente como objetos, están definidas por un conjunto de propiedades y métodos, y están organizadas en torno a una jerarquía de clases.

En Java cada objeto puede tener variables y métodos privados y públicos. Se puede modificar dicha visibilidad de una clase usando los modificadores de acceso a miembros. Las dos maneras más habituales de especificar la accesibilidad son:

`private` – la variable o método está disponible solamente para esta clase.
`public` – la variable o método está disponible para todas las clases.

Una clase puede heredar los variables y métodos públicos de otra clase a través del mecanismo de herencia y la palabra clave `extends`. Por ejemplo:

```
//clase base que va a contener información sobre vehículos de nuestra
empresa:
public vehiculo {
	private int noPuertas;
	private int noRuedas;
	private String modelo;
	public vehiculo(){}
	public void setNoPuertas(int np) {
		noPuertas = np;
	}
	//etc.
}

//una clase para tratar a los coches en general...
public coche extends vehiculo {
	private boolean airbags;
	public coche(){}
	public void setAirbags(Boolean a) {
		airbags = a;
	}
	//etc.
}

//y, por fin, una clase para tratar a los coches deportivos
public final cocheDeportivo extends vehículo {
	private String capacidadMotor;
	private int maxVelocidad;
	public cocheDeportivo(){}
	public void setCapacidadMotor(String cm) {
		capacidadMotor = cm;
	}
	//etc.
	//se puede llamar a cualquier método en las superclases como
	//si estuvieran dentro
	//de esta misma clase, p.ej.:
	setNoPuertas(2);
}
```

Notas: Las clases que extienden otras clases tienen el nombre de subclases y las clases que son extendidas por otras clases tienen el nombre de superclases.

Hay que tener cuidado a la hora de planificar las relaciones de herencia entre clases en Java porque una clase solamente puede heredar variables y métodos de otra (y sus superclases). Es decir, que no hay herencia múltiple en Java como hay en lenguajes como C++ (aunque se puede reproducir la técnica de herencia múltiple usando interfaces…). De todas formas, la manera más habitual para tratar este tema es simplemente usar una clase dentro de otra, por ejemplo, si hay una clase para el aparcamiento de una empresa que ya es una extensión de una clase base aparcamiento, dicha clase no puede heredar ninguna otra clase, por lo tanto, se incluirán las clases de coches, camiones, motos, etc., así:

```
public aparcamientoEmpresa extends aparcamiento {
      private String nombreEmpresa;
      private cocheDirector = new cocheDeportivo(…);
      public aparcamientoEmpresa(){}
      //etc.
      //para llamar a algún método en una clase hay que especificar
      //la variable de la instancia…
      cocheDirector.setCapacidadMotor("4.5l");
}
```

3. Descripción de la práctica - Curso 2024-2025

La práctica del curso 2024-2025 consiste en implementar un **sistema integrado de gestión del plan de movilidad sostenible de una gran ciudad**. El Ayuntamiento desea poner en marcha una flota de diversos tipos de vehículos de alquiler para facilitar la movilidad de sus ciudadanos, por lo que se hace necesario un sistema para gestionar todo lo relacionado con la utilización y mantenimiento de dichos vehículos.

En general, el sistema solicitado ha de permitir gestionar todos los datos de usuarios y administradores del sistema, así como de los vehículos y las estaciones o bases en las que se inician y finalizan los alquileres. Además, se deberá manejar toda la información relativa a los viajes realizados, sus importes, etc. En cuanto a los vehículos, será necesario gestionar información acerca del estado de los mismos, carga de batería, etc.

En la aplicación que se desea desarrollar, es necesario almacenar dos tipos distintos de persona, para diferenciar si se trata de un usuario de vehículos compartidos, o un trabajador del Ayuntamiento. Existen dos tipos de usuarios: i) *usuario estándar*, al cuál se le aplicarán las tarifas completas en el alquiler de los vehículos; y ii) *usuario premium* que, gracias a su fidelidad en el uso de los vehículos, puede disfrutar de ofertas en las tarifas de uso de los vehículos de alquiler, así como de algunas ventajas adicionales que se especificarán posteriormente. Además, los trabajadores del Ayuntamiento pueden separarse en tres tipos diferentes: i) *mecánicos*, que se encargan de la reparación de los vehículos y de las bases defectuosas; ii) *personal de mantenimiento*, que se encarga de la recogida de vehículos defectuosos, y de la recarga de la batería de los vehículos; y, por último, iii) *administrador* del sistema.

Se considerarán tres tipos de vehículos de alquiler: motos eléctricas, bicicletas y patinetes. Además, existen dos tipos de motos, de mayor y menor cilindrada (se pueden considerar motos grandes y pequeñas). El alquiler de motos pequeñas será más económico, y la duración de su batería es menor. Se deberá almacenar información relativa al porcentaje de batería de cada vehículo para saber cuándo es necesario que se recarguen dichas baterías. Las bicicletas y los patinetes deberán iniciar y finalizar el viaje en unas bases distribuidas por toda la ciudad, mientras que las motos podrán iniciar y finalizar viaje en cualquier punto dentro del perímetro de la ciudad. Para ello, se deberá almacenar información relativa a las coordenadas en las que se encuentra cada base (junto con otros datos como su nombre, el número de espacios de que dispone, su ocupación, etc.). Así mismo, las coordenadas de

localización de las motos también deberán ser almacenadas. Para simplificar, se entenderá que la ciudad se sitúa sobre una cuadrícula en la que se utilizan únicamente coordenadas compuestas por dos números enteros (x, y). Se podrán definir los límites de la ciudad a través de vértices en la cuadrícula (por ejemplo, se puede considerar una ciudad cuadrada cuyos límites sean el eje X por el sur, el eje Y por el oeste, la recta (x, M) por el norte y la recta (N, y) por el este). De esta manera se asegurará, por una parte, que las bases de bicicletas y patinetes se encuentran todas dentro de la ciudad, y por otra, que las motos finalizan el viaje dentro de los límites de la ciudad.

En general, las funciones que tiene el sistema de gestión de la movilidad sostenible son las siguientes:

- **Gestión de usuarios**: altas, bajas, modificaciones de las personas que figuran en el sistema (empleados – administradores, mecánicos y encargados de mantenimiento / usuarios – estándar y premium). Los usuarios han de estar dados de alta en el sistema para poder utilizar los vehículos. Como no se solicita la gestión de las autenticaciones de los usuarios, se supondrá que el administrador del sistema tiene potestad para dar de alta a los usuarios.
- **Alquiler de un vehículo**: Un usuario podrá alquilar una bicicleta o un patinete indicando una base que disponga de vehículos disponibles (presentes en la base, sin averías y con batería suficiente). También podrá alquilar una moto seleccionándola en el listado de motos disponibles (sin averías y con batería suficiente). Para realizar el alquiler, el usuario deberá tener saldo positivo. Se almacenará el momento de inicio del alquiler y se calculará su duración hasta la finalización del alquiler. Para ello, se pueden utilizar los mecanismos de gestión horaria que se deseen (introducir manualmente la hora, introducirla automáticamente a partir de la hora del sistema, etc.). Para finalizar el alquiler, la base de finalización (que está dentro de los límites de la ciudad) deberá tener espacios libres en el caso de bicicletas y patinetes, y la moto deberá estar dentro de los límites de la ciudad. Al finalizar el alquiler se actualizará el saldo del usuario según el tiempo de alquiler y la tarifa, así como el estado de la batería del vehículo según el tiempo de utilización.
- **Usuarios premium**: Los usuarios que más utilizan la aplicación pueden ser promocionados a “usuarios premium” por el administrador. Para ello, un usuario ha de haber utilizado algún vehículo al menos 15 veces en el último mes, o 10 veces al mes durante 3 meses seguidos. Además, si un usuario utiliza los tres tipos de vehículos (moto, bicicleta y patinete) al menos una vez cada uno, durante 6 meses seguidos, también podrá ser promocionado a premium. Para realizar esta promoción, el administrador podrá consultar los usuarios que cumplen las condiciones para ser promocionados a usuarios premium y seleccionar aquéllos que son promocionados. Una vez hecho esto, los usuarios premium accederán a una rebaja de un porcentaje específico que se puede definir para la tarifa de cada tipo de vehículo. Además, existen dos ventajas adicionales para un usuario premium: puede realizar la reserva de un vehículo durante 20 minutos, por lo que el vehículo deberá estar en estado “reservado” y, por tanto, no será visible para el resto de usuarios durante ese tiempo. Por otro lado, los usuarios premium pueden utilizar vehículos cuya batería esté por debajo del 20% (aunque se les seguirá aplicando una penalización si agotan la batería completamente, como se explicará más adelante), aunque no por debajo del 10%.

- **Tarifas**: Las tarifas de utilización de los vehículos podrán ser definidas por el administrador y se aplicarán al alquiler de cada uno de los vehículos disponibles: motos, bicicletas y patinetes. Además, el administrador también definirá cuáles son los descuentos que se aplican sobre el uso de cada tipo de vehículo para los usuarios premium. Cuando se finalice un viaje, se actualizará el saldo del vehículo en función de las tarifas definidas. En caso de que el usuario acabe con un saldo negativo, no podrá alquilar un nuevo vehículo hasta que no realice una recarga de saldo (de la que se descontará el saldo negativo) y vuelva a tener saldo positivo.
- **Gestión de la localización de los vehículos**: Los vehículos han de estar siempre dentro de los límites de la ciudad cuando están estacionados. Para ello, las bases de bicicletas y patinetes se encuentran dentro de dichos límites. En el caso de las motos, se ha de comprobar siempre que, cuando se finaliza un viaje, las coordenadas de la moto se encuentran dentro de los límites de la ciudad. En el caso de una ciudad cuadrada, la coordenada *x* deberá estar entre el límite oeste y el límite este de la ciudad (en el ejemplo expuesto anteriormente, entre los valores 0 y *N*) y la coordenada *y* deberá estar entre el límite sur y el límite norte de la ciudad (en el ejemplo expuesto anteriormente, entre los valores 0 y *M*).
- **Gestión de la batería**: Es importante tener en cuenta los porcentajes de batería de los vehículos de alquiler. Con este objetivo, no se podrán iniciar viajes con vehículos por debajo de un 20% de batería, excepto los usuarios premium que podrán utilizar un vehículo cuya batería esté por encima del 10%. La batería se consume al ritmo de un 1% por minuto en el caso de las bicicletas, un 0,5% por minuto en el caso de los patinetes, un 0,4% por minuto en el caso de las motos pequeñas y un 0,25% por minuto en el caso de las motos grandes. Se deberá calcular y actualizar el porcentaje de batería cuando se finalice el viaje. Como medida de precaución, se aplicará una penalización de 1€ en caso de que la batería se acabe antes de llegar al destino (se asumirá que el usuario ha tenido que llevar la bicicleta o el patinete de forma “manual” hasta su base, o que ha tenido que dejar la moto en el punto en el que se terminó la batería). Cuando la batería está por debajo del 20%, es necesario que el administrador asigne el vehículo a un encargado de mantenimiento para que lo recoja y se encargue de la recarga de su batería. El empleado de mantenimiento deberá recoger el vehículo de su base o de las coordenadas en las que se encuentre, y devolverlo a una base o a unas coordenadas cuando la batería esté recargada.
- **Reparación de vehículos y bases**: Los usuarios pueden introducir avisos sobre un vehículo que tiene un fallo mecánico y no puede circular, así como fallos en el funcionamiento de una base de bicicletas o patinetes. En el momento en que el vehículo tiene un fallo registrado no podrá ser utilizado por ningún usuario. Igualmente, si la base está averiada, no se podrá utilizar para iniciar ni finalizar viajes de bicicletas o patinetes, aunque haya vehículos disponibles o espacios libres. El administrador puede visualizar todos los vehículos y bases con fallos para su reparación. Para ello, deberá asignar el vehículo a un encargado de mantenimiento para que retire el vehículo (siguiendo un proceso similar al de recarga de batería), y también asignar dicho vehículo a un mecánico que se encargará de la reparación. En caso de que el fallo sea en una base, el administrador únicamente avisará al mecánico para que acuda a repararlo. Los mecánicos emiten una factura con el importe relativo a la reparación del vehículo o de la base.

- **Ubicación del vehículo más cercano**: En el plano cartesiano, la distancia entre dos puntos A y B, con coordenadas (x_1, y_1) y (x_2, y_2), respectivamente, se representa como d(A, B) y se calcula mediante la fórmula $d(A,B) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$. Una de las funcionalidades a las que ha de tener acceso el usuario es encontrar el vehículo más cercano a su ubicación. Para ello, introducirá las coordenadas en las que se encuentra y el tipo de vehículo que quiere utilizar. La aplicación deberá indicarle la base más cercana, en el caso de bicicletas y patinetes, o las coordenadas de la moto más cercana en caso de que ese sea el vehículo elegido. Se recomienda no introducir una gran cantidad de vehículos en el sistema para evitar que el cálculo sea muy costoso (no se evaluará la escalabilidad del sistema).

Siguiendo las funcionalidades indicadas anteriormente, las acciones que podrá llevar a cabo cada tipo de usuario de la aplicación serán las siguientes:

- **Administrador**:
    - Gestión de usuarios: altas, bajas y modificaciones de todas las personas registradas en el sistema (usuarios, mecánicos y personal de mantenimiento).
    - Visualización del estado de batería de todos los vehículos.
    - Visualización de los avisos de problemas mecánicos en los vehículos.
    - Visualización del estado de las bases de bicicletas y patinetes (vehículos disponibles, huecos disponibles, fallos mecánicos).
    - Asignación de vehículos a mecánicos y a personal de mantenimiento y de bases directamente a mecánicos.
    - Visualización de datos de todas las personas registradas en el sistema.
    - Visualización de la utilización de los vehículos por parte de los usuarios y los importes asociados.
    - Visualización del listado de vehículos en uso en un momento o período de tiempo determinado.
    - Actualización de la flota de vehículos: altas, modificaciones y bajas de todos los vehículos que componen la flota.
    - Generación de estadísticas en relación a las estaciones (bases) con mayor/menor demanda, entre otras.
    - Obtener información sobre los usuarios que deberían ser promocionados a premium y promocionar usuarios a premium.

- **Usuario**:
    - Consulta de vehículos disponibles: ocupación de las bases de bicicletas/patinetes, y coordenadas de las motos. En todos los casos, visualización del nivel de batería de un vehículo.
    - Alquiler de un vehículo: inicio y final de un viaje, seleccionando el tipo de vehículo. En el caso de bicicletas y patinetes indicando la estación inicial y final del viaje. En el caso de motocicletas, indicando las coordenadas de inicio y de fin del viaje.
    - Informar de un problema con un vehículo o con una base de bicicletas o patinetes.

    - Visualización del historial de viajes realizados, con los importes asociados a los mismos.
    - Visualización del saldo disponible y recarga del mismo.
    - Generación de avisos de problemas mecánicos en los vehículos.
    - Consulta de ubicación de la moto más cercana.
    - Reserva de viaje desde 20 minutos antes (**sólo usuarios premium**).

- **Mecánico**:
    - Visualización de los vehículos y bases asignados para su reparación.
    - Recogida de un vehículo para su reparación.
    - Desplazamiento a una base de bicicletas o patinetes para su reparación.
    - Definición del período de inactividad de una base o vehículo.
    - Generación de facturas por la reparación de un vehículo o una base de bicicletas o patinetes.

- **Encargado de mantenimiento**:
    - Visualización de los vehículos asignados para su mantenimiento.
    - Recogida de un vehículo para su mantenimiento (recarga de baterías o reparación).
    - Definición del período de inactividad de un vehículo.
    - Devolución de un vehículo a una base o a unas coordenadas específicas.

4. Desarrollo de la práctica

En esta práctica se propondrán diferentes funcionalidades en función de la calificación a la que aspire el estudiante. De este modo, una mayor complejidad a desarrollar implica una calificación mayor en la evaluación de la práctica. Hay que tener en cuenta que **la nota mínima para aprobar** la práctica es 5.0.

Es importante considerar que para optar a la calificación de un nivel superior han de cumplirse todas y cada una de las funcionalidades especificadas en el nivel inmediatamente anterior. En caso de no ser así (no cumplir con todos los requerimientos de un nivel), no se podrá obtener una calificación superior a la marcada por el nivel cuyas restricciones no se cumplen en su totalidad. Del mismo modo, los niveles han de implementarse en el orden que se indican, no siendo posible implementar niveles no consecutivos para obtener calificaciones superiores.

Para cada uno de los niveles se van a indicar unos requisitos mínimos de cumplimiento. Esto quiere decir que para cualquier otro detalle de diseño que no se encuentre descrito expresamente en lo indicado en este enunciado, el alumno tiene libertad para tomar cuantas decisiones considere oportunas.

Para obtener la nota mínima para aprobar hay que desarrollar los primeros **dos niveles** de la práctica.

Nivel 1 - Puntuación total máxima a obtener: 3 puntos.

Lo que se pretende que el alumno desarrolle en este nivel son las relaciones de clase, herencia y demás que van asociadas al desarrollo de la práctica. Así, se pide realizar las siguientes tareas:

- Planteamiento del problema: actores participantes, relaciones entre actores, funcionalidad a cumplir por la práctica a desarrollar.
- Establecimiento de diferentes clases a intervenir en la práctica, relaciones de dependencia entre clases, identificar diferentes jerarquías de clases, etc.
- Elaboración de un documento escrito (memoria de la práctica) que contenga el primer punto y los correspondientes ficheros para BlueJ que implementen el segundo.

Es necesario contemplar las jerarquías de herencia necesarias para la implementación de lo solicitado en la práctica (tipos de personas, vehículos, etc.).

Nivel 2 - Puntuación total máxima a obtener: 7 puntos.

Los alumnos que implementen este nivel de finalización de la práctica recibirán una puntuación máxima de 7 puntos. Sólo se podrá optar a este Nivel si se ha implementado satisfactoriamente y en su totalidad los requerimientos especificados en el Nivel 1. Lo que se pretende que el alumno desarrolle en este nivel es la parte de gestión de datos del sistema usando una estructura de clases y métodos apropiados. De este modo, el sistema deberá permitir lo siguiente:

- Añadir y actualizar los datos de la flota de vehículos del sistema, así como de las tarifas aplicadas y los descuentos para usuarios premium.
- Dar de alta a usuarios del sistema con sus datos personales.
- Realizar el alquiler de vehículos con todas las características asociadas (gestión de la batería, gestión del saldo, introducción avisos por fallos mecánicos, etc.).
- Gestionar toda la funcionalidad relativa a los usuarios premium: consulta de potenciales usuarios premium por parte del administrador, asignación del estatus de premium a usuarios, gestión de los descuentos y de la posibilidad de realizar reservas. Gestión de la posibilidad por parte de un usuario premium de alquilar vehículos con un nivel de batería entre un 10% y un 20%.
- Asignar vehículos con baja batería o con fallos mecánicos a encargados de mantenimiento para su retirada, y gestionar.
- Permitir que cada encargado de mantenimiento y mecánico vea los vehículos que tiene asignados y realice sus funciones sobre los mismos, dejando constancia, en el caso de los mecánicos, del importe de las reparaciones realizadas.
- Realizar búsquedas sencillas sobre los usuarios y empleados del sistema.
- Realizar consultas y actualizaciones del stock de vehículos del sistema.

Nivel 3 - Puntuación total máxima a obtener: 10 puntos.

Los alumnos que implementen este nivel de finalización de la práctica recibirán una puntuación máxima de 10 puntos. Sólo se podrá optar a este Nivel si se han implementado satisfactoriamente y en su totalidad los requerimientos especificados en el Nivel 2. Lo que se pretende es que el alumno desarrolle en este nivel la interfaz textual del sistema para las

funciones identificadas en el nivel 2 más la consulta de vehículos más cercanos y la generación de listados. De este modo, el sistema deberá permitir lo siguiente:

- Gestionar las funciones identificadas en el nivel 2.
- Permitir la búsqueda de los vehículos más cercanos: que el usuario pueda consultar cuáles son las bases de bicicletas o patinetes, o las motos individuales, más cercanas a su posición.
- Producir diferentes listados y estadísticas del funcionamiento del sistema:
    - Listado de las bases de bicicletas y patinetes, ordenado según su demanda.
    - Listado de los vehículos que han requerido reparaciones en un período de tiempo determinado, incluyendo el número de reparaciones (si hubiera más de una) y el importe de cada una de ellas, así como el importe total.
    - Listado de los encargados de mantenimiento y mecánicos, ordenados de mayor a menor número de intervenciones realizadas (vehículos asignados).
    - Listado de vehículos en función de su tipo, ordenado por su tiempo de uso, de mayor a menor.
    - Listado de los usuarios del sistema, ordenados según sus gastos en alquileres de vehículos dado un período de tiempo determinado.

5. Plan de Trabajo

Para realizar la práctica se seguirá el siguiente método de trabajo:

- En primer lugar, se leerá detenidamente el enunciado de esta práctica.
- A continuación, hay que diseñar, utilizando un paradigma orientado a objetos, los elementos necesarios para cada nivel de la aplicación explicada en el apartado anterior. Debe hacerse uso de los mecanismos de herencia siempre que sea posible. Se valorará un buen diseño que favorezca la reutilización de código y facilite su mantenimiento.
- El código estará debidamente comentado.
- La clase principal que abre la aplicación deberá llamarse “**movilidad.class**”.

6. Control de plagio en las prácticas

Tal y como está indicado en el apartado 10 de este documento, las prácticas son esenciales en las titulaciones de Informática porque permiten a los alumnos adquirir conocimientos importantes sobre los aspectos más aplicados de una asignatura. Por lo tanto, dado el hecho de que la práctica de esta asignatura es un trabajo individual y obligatorio que cuenta para la nota final de la asignatura y que implica un esfuerzo por parte de los alumnos, es necesario garantizar la originalidad de dicho trabajo. Para evitar este problema, una vez terminado el plazo de entrega de la práctica (indicado en el curso virtual), el equipo docente usará un software de control de plagio para revisar las prácticas. En los casos donde haya plagio se informará al Servicio de Inspección de la UNED para que tome las medidas disciplinarias apropiadas.

7. Normas de realización de la práctica

1. La realización de la práctica es obligatoria. Sólo se evaluará el examen si la práctica ha sido previamente aprobada.
2. Si bien el desarrollo de aplicaciones Orientadas a Objetos usando el lenguaje de programación Java no requiere el uso concreto de ningún entorno de desarrollo, está práctica ha de desarrollarse íntegramente empleando el entorno de desarrollo BlueJ, que es el que se muestra en el libro de texto básico de la asignatura.
3. La práctica es un trabajo individual. Las prácticas cuyo código coincida total o parcialmente con el de otro alumno serán motivo de suspenso para todos los implicados (copiadores y copiados), no pudiéndose examinar ninguno de ellos en el presente curso académico (además de cualquier medida disciplinaria que aplicará el Servicio de Inspección).
4. Cada tutor será responsable de organizar las sesiones de control presenciales o virtuales de la realización de la práctica. Al menos una de dichas sesiones de control deberá ser obligatoria:
    a. Es el tutor el que marca la fecha de dicha sesión y no el equipo docente.
    b. El tutor puede organizar la sesión hacia el final del cuatrimestre para poder comprobar que los alumnos han hecho bien el trabajo y para ayudar al tutor a calificar el trabajo.
5. La única vía de entrega de la práctica es a través de la plataforma Ágora siguiendo las indicaciones del apartado 8.
6. El equipo docente tendrá en cuenta prácticas con notas altas para aquellos alumnos cuyo examen esté cercano al aprobado.
7. El alumno debería dirigirse a su tutor para cualquier duda que tenga sobre su práctica y solamente al equipo docente (por correo electrónico) en el caso de que su tutor no pueda resolver su problema. En este caso, pediremos al alumno que, además de sus datos personales, nos envíe el nombre del centro asociado en el que está matriculado y el de su tutor.
8. Evidentemente se pueden usar los foros para realizar consultas a los compañeros, pero nunca para intercambiar código.
9. La fecha límite de entrega de la práctica la establecerá **el tutor**, junto con el **procedimiento de seguimiento y entrega** que el tutor quiere seguir. Por lo tanto, es responsabilidad del **alumno** informarse de dicho procedimiento y de las fechas asociadas, que se comunican al inicio del curso.

   La fecha que figura en ÁGORA para la práctica, el **último domingo antes del inicio de la primera semana de pruebas presenciales de junio**, sirve exclusivamente para que **el tutor** pueda subir las notas y **no** para que un alumno entregue su práctica con la esperanza que sea corregida por el tutor o el equipo docente. Cualquier práctica que se suba a ÁGORA fuera del plazo **no será corregida**, y el alumno **no podría aprobar** el examen en la convocatoria de junio.

   Para la **convocatoria de septiembre**, la nota de la práctica tiene que estar incluida en el curso virtual por los tutores **antes del 31 de agosto de 2025**.

   Si el tutor en el centro asociado está dispuesto a corregir la práctica para la convocatoria de septiembre, el alumno debe seguir las indicaciones del tutor al respecto. En caso contrario, el alumno deberá enviar su práctica al tutor Antonio Sernandez (asernandez@ponferrada.uned.es), quien se encargará de su corrección.

Asimismo, el alumno deberá ponerse en contacto con él para gestionar la fecha de entrega de la práctica al mismo.

8. Entrega de la práctica

La práctica se entrega a través de la plataforma Ágora en el apartado “Entrega de trabajos”. El archivo que hay que subir a Ágora debe ser un archivo comprimido (rar o zip), que se puede preparar con el software de compresión que traen la mayoría de los sistemas operativos hoy en día o usando un software libre como 7zip (www.7-zip.org). **No se deben usar acentos** en los nombres de los archivos ni las carpetas. El archivo comprimido debe estar compuesto por una carpeta con el nombre del alumno que contendrá dos elementos:

1. **Memoria**: La memoria constará de los siguientes apartados:
    - Portada con título “Práctica de Programación Orientada a Objetos – Curso 2024-2025” y los datos del alumno: Nombre, Apellidos, dirección de correo electrónico y teléfono de contacto.
    - Análisis de la aplicación realizada, mostrando el funcionamiento del programa, estrategias implementadas, decisiones de diseño establecidas y, en general, toda aquella información que haga referencia a las diferentes decisiones tomadas a lo largo del desarrollo de la práctica, junto a una justificación de dichas decisiones.
    - Diagrama de clases, detallando claramente el tipo de relación entre ellas (uso, agregación, herencia, ...).
    - Un texto en el que se describa cada clase/objeto, justificación de su existencia, métodos públicos que contiene y funcionalidad que realizan.
    - Anexo con el código fuente de las clases implementadas.

2. **Una carpeta con el código**: incluyendo todos los ficheros *.java y *.class, así como la memoria en formato electrónico (preferiblemente html o pdf).

NOTAS:

- Al hacer la entrega del trabajo se acepta que tanto el código fuente Java como la memoria de la práctica es original. Aquellos aportes intelectuales de otros autores (como, por ejemplo, el tutor) deben estar referenciados debidamente en el texto de dicho trabajo.
- Si el archivo subido a ÁGORA por parte del alumno no sigue estas indicaciones, está infectado con algún virus, o no se puede descomprimir, el equipo docente no aceptará la práctica y se calificará con una nota de 0.

9. Normas para los tutores

Como se puede apreciar, el papel del tutor es fundamental en todos los aspectos de la práctica, tanto el planteamiento del problema, el diseño orientado a objetos del programa, su desarrollo y su depuración. Tratándose de una asignatura obligatoria, cada alumno debería tener acceso a un tutor. Los tutores deben seguir los siguientes pasos:

1. Ayudar a los alumnos al principio del curso con el planteamiento de la práctica y las normas que tienen que seguir.
2. Para explicar ciertos conceptos relacionados con la solución de la práctica, el tutor puede dar fragmentos de código fuente a los alumnos. Los pequeños fragmentos no tendrán importancia a la hora de llevar a cabo el control de plagio por parte del equipo docente. No obstante, si un alumno va a incluir un fragmento de código en su práctica, debe incluir un comentario al respecto directamente anterior al código y también una nota al respecto en su memoria.
3. Informar a los alumnos acerca de las sesiones de control presenciales o virtuales de la práctica. Al menos una de dichas sesiones de control deberá ser obligatoria.
4. Informar a los alumnos acerca de la fecha límite de entrega de la práctica
5. Comunicar la calificación a sus alumnos.

10. Centros Asociados vs. Prácticas en Asignaturas Obligatorias

Las prácticas son esenciales en las titulaciones de Informática porque, entre otras cosas, permiten a los alumnos adquirir conocimientos importantes sobre los aspectos más aplicados de ciertas asignaturas, lo cual resulta de gran relevancia e interés a la hora de acceder a un puesto laboral relacionado con la Informática. Para orientar y ayudar a los alumnos, así como para comprobar que realmente un alumno ha realizado su práctica de forma satisfactoria, ésta se debe realizar en un Centro Asociado bajo la supervisión de un tutor, quien decide, en última instancia, la forma en la cual se organiza el desarrollo de la misma en su Centro Asociado (existencia o no de sesiones presenciales obligatorias, forma de entrega, etc.).

De vez en cuando sucede que un alumno se pone en contacto con un Equipo Docente del Departamento de Lenguajes y Sistemas Informáticos (L.S.I.) porque se ha matriculado en una asignatura obligatoria en un Centro Asociado que no le proporciona un tutor para supervisar la práctica, aún cuando se le ha permitido matricularse. El alumno busca en el Equipo Docente que se le proporcione una solución a este problema, como por ejemplo, la posibilidad de asistir a unas sesiones extraordinarias de prácticas en la Sede Central de la U.N.E.D. en Madrid o la posibilidad de realizar la práctica por su cuenta en casa, enviándola a continuación al Equipo Docente para su corrección. Sin embargo, los Equipos Docentes de L.S.I. no disponen de recursos para poder llevar a cabo ninguna de estas dos alternativas.

Un Centro Asociado que ha permitido a un alumno matricularse en una asignatura obligatoria de una carrera de Informática debería ayudarle a encontrar una solución al problema de la realización de las prácticas. Si se trata de una asignatura donde no se han matriculado muchos alumnos, quizás el centro no cuente con recursos para proporcionar un

tutor específicamente para la asignatura. Si hay otro Centro Asociado cerca que dispone de tutor, quizás el alumno pueda realizar la práctica allí. Pero si no es así, el Centro Asociado debería proporcionar un tutor para supervisar y corregir las prácticas de sus alumnos. Lo más razonable sería que fuera un tutor de otra asignatura de Informática en el mismo Centro el que hiciera la sesión de prácticas para los alumnos de la asignatura en cuestión, y al final de la sesión evaluará los trabajos de los alumnos, según las pautas marcadas por el Equipo Docente, haciendo llegar a éste las calificaciones otorgadas.

Por lo tanto, un alumno que, tras haberse matriculado en una asignatura obligatoria en un Centro Asociado, se encuentre con que el centro no tiene tutor para dicha asignatura, debería dirigirse al Director del Centro Asociado, para solicitar de él una solución, tal como se ha presentado aquí, es decir, alguien que pueda supervisar y corregir su práctica con plenas garantías. En el caso de que el Director no le proporcione una solución, el alumno debería comunicárselo, por escrito, lo antes posible, a la Directora del Departamento de L.S.I., Dra. Raquel Martínez Unanue.