Sistema Integrado de Gestión de una fabrica artesanal de muebles

Práctica de la asignatura de Programación Orientada a Objetos
Escenario para el Curso 2019-2020 – Febrero de 2020 – Versión 1.1

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
        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 está 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

La práctica del presente curso va a consistir en diseñar e implementar un sistema integrado de gestión de una fábrica artesanal de muebles. Hace años el responsable de una fábrica no haría más que apuntar en un libro una breve descripción del pedido de un cliente, con los muebles que quiere comprar y las características de cada uno, junto con algún número de contacto. Hoy en día, debido en parte a la competición entre fábricas y al deseo de aportar servicios de calidad para poder fidelizar al cliente, las fábricas usan sistemas de gestión para todo el proceso de fabricación, así como para la venta y la facturación.

Para diferenciarse de otras fábricas y grandes almacenes de muebles, el dueño de la fábrica ha decido que todos sus muebles estarán fabricados a mano por artesanos especialistas. Por ello, y por los costes de almacenaje de los muebles, la fabricación de todos los muebles se hace bajo demanda.

En general, son varias las funciones que tiene un sistema de gestión de una fábrica artesanal de muebles:

- Recepción de un pedido de un cliente con los muebles que quiere comprar y las características de cada uno.
- Asignación por parte del jefe de un pedido a un artesano en la fábrica. Para simplificar la gestión del proceso en esta práctica, se da por hecho que todos los artesanos en la fábrica son igual de capaces de fabricar todos los muebles.
- Proceso de fabricación: un artesano, al terminar la fabricación de un mueble, acude al sistema para ver los próximos trabajos que le tocan. Según la información en el sistema por el pedido actual, tiene que ponerse con la fabricación de un nuevo mueble y dejar constancia del proceso en el sistema, de manera que el jefe pueda inspeccionar los progresos de cada artesano y asignar los nuevos pedidos en consecuencia. Si por el motivo que sea (por ejemplo, falta de piezas), un artesano deja un trabajo en un estado sin completar, anota el motivo en el sistema y pasa al siguiente trabajo.
- Gestión de usuarios: altas, bajas, modificaciones de las personas que figuran en el sistema, tanto empleados como clientes.
- Gestión de clientes por parte del comercial: comunicar a los clientes el precio de un pedido, informarles de que su pedido está listo para recoger/entregar, etc.

Para estructurar los datos en el sistema se contemplarán dos jerarquías de objetos:

- **Mueble** con dos subtipos:
    - **Mesa** con sus subtipos: **mesa de café** con dos subtipos (**mesa de café de cristal** y **mesa de café de madera**), **mesa de dormitorio** y **mesa de comedor**.
    - **Silla** con sus subtipos (**silla de cocina**, **silla de oficina** con dos subtipos (**silla de oficina con ruedas** y **silla de oficina sin ruedas**) y **silla plegable**).
- **Persona** con dos subtipos:
    - **Empleado** con sus subtipos: **jefe**, **comercial** y **artesano** con dos subtipos (**contrato por hora** y **en plantilla**).
    - **Cliente** con sus subtipos: **persona** y **empresa.**

En estas dos jerarquías se contemplan un total de 23 tipos de objetos, los cuales deben estar explícitamente representados, cada uno en su clase particular.

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 implicará 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. 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 dos jerarquías de herencia que incluyan las características necesarias para poder diferenciar su posición en cada jerarquía: una para los muebles y otra para las personas.

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 de la fabrica usando una estructura de clases y métodos apropiados. De este modo, el sistema deberá permitir lo siguiente:

- Añadir nuevos muebles al sistema y actualizar los datos de muebles ya en el sistema.
- Dar de alta a empleados y clientes en el sistema con sus datos personales.
- Dar de alta un nuevo pedido de muebles con los datos del cliente correspondiente.
- Realizar búsquedas sencillas sobre los empleados, clientes y pedidos.

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 gestión de artesanos y el proceso de fabricación de los muebles. De este modo, el sistema deberá permitir lo siguiente:

- Gestionar las funciones identificadas en el nivel 2.
- Permitir la asignación de las fichas de los empleados, clientes y pedidos.
- Permitir que cada artesano vea las fichas que le toca gestionar y pueda editar los datos dejando constancia del trabajo realizado y el estado de fabricación (por ejemplo, pendiente, en proceso, parado [hace falta piezas, pendiente de confirmación del cliente], fase de prueba, terminado).
- Producir diferentes listados del funcionamiento de la fabrica: las piezas que hace falta pedir para las fabricaciones, las fichas procesadas por cada artesano, las confirmaciones que hay que solicitar a los clientes, las fichas en proceso y un historial de cada artesano y cada mueble.

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 “fabrica.class”.

6.- Control de plagio en las prácticas

Tal y como esta indicado en el apartado 10 de este anuncio, 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. Aunque 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 cualquiera medida disciplinaria que aplicará el Servicio de Inspección).
4. Cada tutor organizará una sesión de control de la realización de la práctica:
    a. Es el **tutor** el que marca la fecha de dicha sesión y no el equipo docente. Los tutores enviarán posteriormente listados de los alumnos que han asistido a dicha sesión.
    b. La asistencia a dicha sesión es **obligatoria** y se debería realizar antes de la entrega de la práctica en la plataforma aLF.
    c. 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.
    d. El tutor entrará en el espacio virtual de la asignatura dentro de aLF antes del 1 de junio, para meter las notas para sus alumnos.
    e. En el caso de que un alumno no pueda asistir a la sesión de control debido a una causa mayor (p.ej., por motivos médicos o de trabajo), se lo debería informar al equipo docente (juntando un certificado médico o carta de su empresa) antes del 1 de junio. Una vez empezado el periodo de exámenes no se aceptarán estos avisos.
5. La **única vía** de entrega de la práctica es a través de la plataforma aLF 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.

8.- La entrega de la práctica

La práctica se entrega a través de la plataforma aLF en el apartado "Entrega de trabajos". El archivo que hay que subir a aLF 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 contiene dos cosas:

1. **Memoria**: La memoria constará de los siguientes apartados:
   - Portada con título "Práctica de Programación Orientada a Objetos – Curso 2016-2017" 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 aLF por parte del alumno no sigue estas indicaciones, está infectado con algún virus, o que 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 respeto en su memoria.
3. Indicar a los alumnos que habrá **una sesión obligatoria** de seguimiento y evaluación de la práctica.

4. Una vez terminada y entregada la práctica, el tutor debe entrar en el espacio virtual de la asignatura dentro de aLF, antes del 1 de junio, para meter las notas de sus estudiantes.
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 evaluara 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, al Directora del Departamento de L.S.I., Dra. Lourdes Araujo.