Práctica 3

Introducción a la Programación Orientada a Objetos con Java

Inicio: Semana del 26 de febrero.
Duración: 3 semanas.
Entrega: Semana del 19 de marzo.
Peso de la práctica: 20%

El objetivo de esta práctica es introducir al alumno en la programación orientada a objetos con el lenguaje Java, pidiéndole que desarrolle de forma incremental varias clases en Java, incluyendo sus pruebas y documentación, que implementen diversos componentes software que sirvan de base para aplicaciones cuyo objetivo sea la simulación y gestión de un cine.
En el desarrollo de esta práctica se utilizarán principalmente los siguientes conceptos de Java:

- *Tipos de datos primitivos, String, Array y tipos referencia* (objetos) definidos por el programador,
- *Clases* sencillas definidas por el programador para implementar tipos abstractos de datos mediante *variables de instancia, variables de clase, métodos de instancia, métodos de clase, y constructores*.
- *Herencia, sobrescritura de métodos*
- *Comentarios para documentación automática mediante javadoc.*

Apartado 0. Introducción

En esta práctica vamos a ir desarrollando progresivamente una aplicación muy sencilla para la simulación de un *cine* en el que se incluirá información sobre las películas, las salas, las sesiones en las que se proyectarán las películas y el precio de las entradas, que podrá cambiar dependiendo de si es el día del espectador. Obviamente, la aplicación obtenida al final de esta práctica será muy simple, pero debe desarrollarse siguiendo fielmente las especificaciones de diseño que se describen en el enunciado, pero a la vez pensando en que se faciliten posibles ampliaciones futuras.

Apartado 1. Las clases básicas

Clase enum *genero (0.5 puntos)*:
Esta clase enumera los géneros de las películas para la clase *Pelicula*.

Clase *Pelicula (1 punto)*:
Esta clase se compone de información de películas individuales. Solamente tiene los métodos típicos get y set además de un método encargado de mostrar información por pantalla. Los atributos de esta clase han de ser, al menos, *Titulo, Director, Anno, Sinopsis y Genero.*

Clase *Sesion (1.5 puntos)*:
Clase que incluye la información sobre las sesiones en las que se proyectan las películas. Debe tener los atributos *Fecha, Película, ButacasDisponibles y Butacas.* Existe una alternativa en la que se puede incluir un atributo del tipo *Sala* en esta clase. El alumno debe valorar si es conveniente y qué ventajas tendría en caso de hacerlo. A través de los métodos de esta clase, debe ser posible actualizar las butacas vendidas, que servirá para cambiar el valor del atributo *ButacasDisponibles* cuando se vendan entradas para la sesión correspondiente.
Es importante estar al tanto de que hará falta un control de errores externo para controlar el número de butacas totales y hacerlo coincidir con la Sala en la que se incluye el objeto. El método *actualizarButacasVendidas* tiene de salida una variable booleana que facilita la programación de métodos que lo llaman, además de que es un ejemplo de control de errores a bajo nivel (en esta función) y tratamiento de errores a alto nivel (en los métodos que lo llaman) pues emplear una variable booleana le da información al método superior para tomar distintas decisiones, lo cual no podría hacer si no tuviera información sobre si la operación se ha podido realizar (que es el caso de usar void).

Clase Sala (1 punto):

Esta clase contiene la información sobre las salas en las que se proyectan películas en el cine. Sus atributos son *Identificador, Butacas* y una lista de sesiones.
La primera particularidad de esta clase es que tiene un atributo, Identificador, que no se puede modificar una vez creada la sala.
Esta clase debe tener un control de errores al añadir sesiones a la sala en la que se comprueba que no haya más sesiones con la misma fecha. Este hecho, que parece bastante lógico, pues entendemos que cada sala solamente puede emitir una película a la vez, facilita mucho las búsquedas por fecha en el resto de clases, pues pueden tener la certeza de que la primera sesión que encuentren con cierta fecha será la única disponible. Este es un ejemplo de las ventajas del control de errores a bajo nivel.
En esta clase se deberá hacer hincapié en la forma correcta de programar orientado a objetos, donde se usan los métodos de otras clases siempre que sea posible en lugar de duplicar funcionalidades.

Clases Entradas y EntradaDiaEspectador (2 puntos):

Son dos clases relacionadas por herencia. El método *getPrecio*, tendrá asociado un descuento el día del espectador. Se debe elegir un formato en el que la única forma de hacer una entrada con precio reducido sea a través de los conceptos de herencia y polimorfismo, que es el objetivo de este apartado de la práctica.

Clase Cine (3 puntos):
Esta es la clase más complicada (y divertida de programar) de todo el proyecto, pues gestiona todo el cine y, por tanto, hace uso de todas las clases definidas hasta ahora.

Debe tener los atributos *Nombre* (del cine), *Direccion, ListaPeliculas, ListaSalas y ListaEntradas*. Así mismo, debe incluir métodos para poder crear películas y asignarlas a las salas y sesiones que se desee, vender entradas, dar información sobre la recaudación, cartelera, Sesiones y fechas de películas.

En la programación de esta clase, los métodos que realizan búsquedas, añaden o quitan elementos de los arraylist, etc, serán programados pensando en dar la mayor cantidad de información útil al usuario. Por ejemplo, el método *removePeliculaCartelera* informa de si la película se ha eliminado correctamente o no, y del número de sesiones que se han eliminado en cada sala.
El método *ventaEntradas* es de los más complejos de la práctica e incluye una gran cantidad de control de errores. También debe intentar mostrar la mayor cantidad posible de información al usuario, como la sala en la que se proyectará la película (no tiene sentido hacer una venta de entradas y no decir al cliente la sala a la que tiene que dirigirse!).

Clase Principal (Pruebas) (1 punto):
Esta clase sirve simplemente para implementar las pruebas de las distintas clases.

- Pruebas unitarias: Se deberán realizar pruebas unitarias rigurosas (incluyendo funcionamiento normal y casos en los que no se puede realizar la operación) de las clases Pelicula, Sesion y Sala.
- Pruebas de integración: Las pruebas de integración del proyecto realizarán a través de las pruebas unitarias de la clase cine. Esta clase, a través de sus distintos métodos, hace uso de la gran mayoría de métodos del resto de las clases, por lo que se pueden considerar pruebas de integración. Debido a que el correcto funcionamiento de esta clase implica también el correcto funcionamiento del resto, se pueden omitir pruebas unitarias de los módulos de Entradas y EntradaDiaEspectador.

Apartado opcional.
Clase *Cartelera* (0.5 puntos)

Crear una clase adicional llamada *Cartelera* que incluya la información de las películas y un método para mostrar esta información de forma agradable. La creación de esta clase modifica los atributos de la clase Cine y el diagrama de clases, de los que deberán entregarse las versiones modificadas para que se puntúe completamente este apartado.

Ampliación en descuentos (0.5 puntos)
Se debe diseñar un sistema de descuentos para estudiantes, personas mayores, días de fiesta, etc, que ajuste en función del cliente y el día de la semana un precio diferente a las entradas. Este precio debe adaptarse a través de los conceptos de herencia y polimorfismo.

Normas de Entrega:

- Se deberá entregar
  - un directorio **src** con todo el código Java en su **versión final** (Apartados 4, 5 y 6),
  - un directorio **doc** con la documentación generada
  - un directorio **txt** con todos los archivos utilizados y generados en las pruebas
  - un archivo PDF con una breve justificación de las decisiones que se hayan tomado en el desarrollo de la práctica, los problemas principales que se han abordado y cómo se han resuelto, así como los problemas pendientes de resolver; además se debe incluir el **diagrama de clases** y una explicación de las **pruebas realizadas**, cuyos ficheros de entrada y código Java deben de estar incluidos en los directorios antes descritos.
- Se debe entregar un único fichero ZIP con todo lo solicitado, que deberá llamarse de la siguiente manera: GR<numero_grupo>_<nombre_estudiantes>.zip. Por ejemplo Marisa y Pedro, del grupo 2261, entregarían el fichero: GR2261_MarisaPedro.zip.