Análisis Cinemax

Sofia Morato

202321074

CineMax es una cadena de cines que busca implementar un sistema web para la venta de boletos de cine. Este sistema tiene como objetivo simplificar el proceso de reserva de boletos, permitiendo que los clientes exploren películas, elijan funciones, seleccionen asientos, y realicen compras de manera fácil. Además, busca optimizar la gestión de las operaciones de cine, incluyendo la supervisión de horarios y ventas. El sistema debe atender a varios tipos de usuarios: clientes, empleados y gerentes. El sistema también incluye un **programa de lealtad** que otorga puntos por compras de boletos y permite a los clientes avanzar en niveles de membresía con beneficios exclusivos.

**2. Entidades y sus Atributos**

**a. Película (abstracta)**

* **Atributos**
  + titulo: Título de la película.
  + genero: Género de la película (acción, comedia, etc.).
  + duracion: Duración de la película en minutos.
  + clasificacionPorEdad: Clasificación de la película por edad (A, B, C, etc.).
  + fechaEstreno: Fecha de estreno de la película.
  + director: Nombre del director de la película.
  + actoresPrincipales: Lista de los actores principales.
* **Métodos**
  + **agregarFuncion(Funcion funcion): void**

Agrega una nueva función de la película.  
Descripción: Este método permite asociar una función a la película específica.

* + **mostrarDetalles(): String**

Muestra detalles de la película (título, género, clasificación, director, actores).  
Descripción: Devuelve un string con la información detallada de la película.

**b. Película Animada (hereda de Película)**

* **Atributos**
  + tecnicaAnimacion: Técnica de animación utilizada (2D, 3D, stop-motion).
* **Métodos:**
  + **mostrarDetalles(): String**

Muestra los detalles específicos de una película animada, incluyendo la técnica de animación.

Descripción: Sobrescribe el método de la clase Película para mostrar detalles adicionales sobre la técnica de animación utilizada.

**c. Largometraje (hereda de Película)**

* **Atributos**
  + narrativa: Detalles narrativos de la película.
* **Métodos:**
  + **mostrarDetalles(): String**

Muestra detalles adicionales del largometraje.

Descripción: Sobrescribe el método de la clase Película para incluir detalles narrativos adicionales si los hay.

**d. Documental (hereda de Película)**

* **Atributos**
  + narrador: Persona que narra el documental.
  + tematica: Tema del documental.
* **Métodos:**
  + **mostrarDetalles(): String**

Muestra detalles adicionales del documental, incluyendo el narrador y la temática.

Descripción: Sobrescribe el método de la clase Película para incluir información específica del narrador y la temática.

**e. Cine**

* **Atributos**
  + nombreCine: Nombre del cine.
  + direccionCine: Dirección física del cine.
  + salas: Mapa de salas en el cine (clave: número de sala, valor: objeto Sala).
* **Métodos:**
  + **getSalas(): HashMap<int, Sala>**

Devuelve un mapa de las salas del cine.

Descripción: Este método devuelve un diccionario de las salas disponibles, con el número de sala como clave.

* + **agregarSala(Sala sala): void**

Agrega una nueva sala al cine.

Descripción: Permite añadir nuevas salas al cine.

* + **getNombre(): String**

Devuelve el nombre del cine.

Descripción: Retorna el nombre del cine para ser mostrado o utilizado en el sistema.

* + **getDireccion(): String**

Devuelve la dirección del cine.

Descripción: Retorna la dirección del cine para consultas o verificaciones.

**f. Sala**

* **Atributos**
  + numeroSala: Número que identifica la sala.
  + capacidad: Capacidad total de asientos en la sala.
  + tipoTecnologia: Tipo de tecnología (IMAX, 3D, etc.).
  + asientos: Lista de objetos Asiento.
  + funcionActual: Función actual programada en la sala.
* **Métodos:**
  + **agregarAsiento(Asiento asiento): void**

Agrega un asiento a la sala.

Descripción: Permite agregar asientos a la sala.

* + **reservarFuncion(Funcion funcion): void**

Asigna una función a la sala.

Descripción: Programa una función en la sala.

* + **getNumeroSala(): int**

Devuelve el número de la sala.

Descripción: Devuelve el número de la sala para propósitos de identificación.

* + **getCapacidad(): int**

Devuelve la capacidad total de la sala.

Descripción: Retorna la cantidad máxima de asientos que puede contener la sala.

* + **getTipoTecnologia(): String**

Devuelve el tipo de tecnología de la sala (IMAX, 3D, etc.).

Descripción: Retorna la tecnología de proyección disponible en la sala.

**g. Asiento**

* **Atributos:**
  + numeroAsiento: Número que identifica el asiento.
  + numeroFila: Número de la fila del asiento.
  + tipoAsiento: Tipo de asiento (regular, premium).
  + disponible: Indica si el asiento está disponible para reservas.
* **Métodos:**
  + **marcarReservado(Compra compra): void**

Marca el asiento como reservado.

Descripción: Cambia el estado de disponibilidad del asiento a no disponible cuando un cliente lo reserva.

* + **esDisponible(): boolean**

Verifica si el asiento está disponible.

Descripción: Devuelve true si el asiento está disponible para reservas, false si ya está ocupado.

* + **getNumeroAsiento(): int**

Devuelve el número del asiento.

Descripción: Retorna el número del asiento para propósitos de identificación.

* + **getNumeroFila(): int**

Devuelve el número de fila del asiento.

Descripción: Devuelve el número de fila para identificar la ubicación del asiento dentro de la sala.

* + **getTipoAsiento(): int**

Devuelve el tipo de asiento (regular, premium).

Descripción: Retorna el tipo del asiento para diferenciar entre asientos estándar y premium.

**h. Función**

* **Atributos**
  + fecha: Fecha de la función.
  + hora: Hora de la función.
  + pelicula: Objeto Película que se proyecta en la función.
  + sala: Objeto Sala donde se proyecta la función.
  + compras: Lista de compras asociadas a la función.
* **Métodos:**
  + **mostrarFuncion(): String**

Muestra los detalles de la función (fecha, hora, sala, película).

Descripción: Devuelve una cadena con la información de la función para los clientes.

* + **getPelicula(): Pelicula**

Devuelve la película asociada a la función.

Descripción: Retorna la película que se proyectará durante la función.

* + **getSala(): Sala**

Devuelve la sala asignada para la función.

Descripción: Retorna la sala donde se realizará la proyección.

* + **registrarCompra(Compra compra): void**

Registra una nueva compra asociada a la función.

Descripción: Permite asociar una compra de boletos con la función correspondiente.

**i. Cliente**

* **Atributos**
  + idCliente: Identificador único del cliente.
  + correo: Correo electrónico del cliente.
  + contraseña: Contraseña del cliente para acceder al sistema.
  + historialCompras: Lista de compras realizadas por el cliente.
  + lealtad: Programa de lealtad al que está asociado el cliente.
  + reserva: Función reservada por el cliente.
* **Métodos:**
  + **reservarFuncion(Funcion funcion): void**

Reserva una función para el cliente.

Descripción: Permite a un cliente reservar una función.

* + **realizarCompra(Compra compra): void**

Registra una compra de boletos por parte del cliente.

Descripción: El cliente completa la compra de boletos para una función.

* + **agregarCompraAlHistorial(Compra compra): void**

Añade una compra al historial del cliente.

Descripción: Guarda la compra en el historial del cliente para referencia futura.

* + **canjearLeal(ProgramaLealtad programaLealtad): String**

Canjea puntos de lealtad para obtener beneficios.

Descripción: Permite al cliente usar sus puntos acumulados en el programa de lealtad para obtener recompensas o descuentos.

**j. Compra**

* **Atributos**
  + monto: Monto total de la compra.
  + fechaTransaccion: Fecha de la transacción.
  + asientosComprados: Lista de objetos Asiento comprados en la transacción.
  + metodoPago: Método de pago (tarjeta de crédito, débito).
  + funcion: Función para la que se realizó la compra.
  + cliente: Cliente que realizó la compra.
* **Métodos:**
  + **registrarCompra(): void**

Registra la compra de boletos de una función.

Descripción: Finaliza la compra de boletos, confirmando la transacción.

* + **asociarAsientos(ArrayList<Asiento> asientos): void**

Asocia los asientos comprados con la compra.

Descripción: Guarda los asientos comprados por el cliente durante la transacción.

* + **getMonto(): double**

Devuelve el monto total de la compra.

Descripción: Calcula el costo total de la compra basado en el número de asientos y sus precios.

* + **getFechaTransaccion(): String**

Devuelve la fecha de la transacción.

Descripción: Retorna la fecha en la que se completó la compra.

* + **getMetodoPago(): String**

Devuelve el método de pago utilizado en la compra.

Descripción: Retorna si el cliente usó tarjeta de crédito o débito para pagar.

**k. Programa de Lealtad**

* **Atributos:**
  + puntosLealtad: Puntos acumulados por el cliente.
  + nivelMembresia: Nivel de membresía del cliente.
  + clientes: Lista de clientes asociados al programa.
* **Métodos:**
  + **getPuntosLeal(): int**

Devuelve los puntos acumulados en el programa de lealtad.

Descripción: Permite al cliente consultar cuántos puntos ha acumulado.

* + **getNivelMembresia(): String**

Devuelve el nivel de membresía basado en los puntos.

Descripción: Retorna el nivel de membresía del cliente (por ejemplo, básico, plata, oro).

* + **agregarCliente(Cliente cliente): void**

Asocia un cliente al programa de lealtad.

Descripción: Permite que los clientes se registren en el programa y acumulen puntos.

**3. Relaciones y Cardinalidades**

**a. Cine - Sala (Cine tiene salas) Composición**

* **Relación**: Un cine contiene varias salas, y cada sala pertenece a un cine.
* **Cardinalidad**: 1 a \* (Un cine tiene múltiples salas).
* **Descripción**: Cada cine contiene varias salas donde se proyectan las películas.

**b. Sala - Asiento (Sala contiene asientos) Composición**

* **Relación**: Una sala contiene varios asientos, y cada asiento está asociado con una sala.
* **Cardinalidad**: 1 a \* (Una sala tiene múltiples asientos).
* **Descripción**: Los asientos están dispuestos dentro de una sala específica, y cada asiento tiene un estado de disponibilidad.

**c. Película - Función (Película tiene funciones) Asociación**

* **Relación**: Una película puede tener múltiples funciones programadas.
* **Cardinalidad**: 1 a N (Una película tiene varias funciones).
* **Descripción**: Cada película puede ser proyectada varias veces en diferentes horarios (funciones).

**d. Función - Sala (Función se proyecta en Sala)**

* **Relación**: Una función se proyecta en una sala específica.
* **Cardinalidad**: 1 a 1 (Una función se realiza en una sala a la vez).
* **Descripción**: Cada función tiene lugar en una sala específica, en un horario y fecha determinados.

**e. Función - Compra (Función tiene compras)**

* **Relación**: Varias compras pueden estar asociadas con una función.
* **Cardinalidad**: 1 a N (Una función puede tener múltiples compras).
* **Descripción**: Los clientes pueden comprar boletos para una función específica, y cada compra se asocia con una función.

**f. Cliente - Compra (Cliente realiza compras)**

* **Relación**: Un cliente puede realizar varias compras.
* **Cardinalidad**: 1 a N (Un cliente puede tener múltiples compras).
* **Descripción**: Los clientes pueden comprar múltiples boletos a lo largo del tiempo.

**g. Cliente - Programa de Lealtad (Cliente tiene lealtad)**

* **Relación**: Un cliente puede estar asociado a un programa de lealtad.
* **Cardinalidad**: 1 a 1 (Un cliente tiene un programa de lealtad).
* **Descripción**: Cada cliente puede acumular puntos y beneficios dentro de un programa de lealtad.

h. Función Cliente:

* **Relación:** Un Cliente realiza varias Compras, y cada Compra está asociado a un único Cliente.
* **Cardinalidad:** Pago (1) <-> Cliente (\*).
* **Descripción:** Un cliente puede realizar múltiples pagos a lo largo del tiempo (reservando diferentes funciones o asientos), pero cada pago está asociado a un solo cliente. Esto garantiza que los pagos se puedan rastrear individualmente para cada cliente.

i**. Película – Herencia:**

* Largometraje, Documental, y Animación son subclases que heredan de Película.
* Cada subclase puede tener atributos y métodos adicionales que son específicos para ese tipo de película.

**4. Reglas del Dominio**

**Películas:**

* Una película solo puede tener funciones programadas después de su fecha de estreno.
* Los detalles que se muestran varían según el tipo de película.

**Funciones:**

* Una función solo puede ser programada en una sala si esta no tiene otra función en ese momento.
* Los clientes no pueden comprar boletos para funciones donde no hay asientos disponibles.

**Clientes y Compras:**

* Los clientes solo pueden reservar funciones si hay asientos disponibles.
* Las compras deben asociarse a asientos disponibles y a un método de pago válido.
* Los clientes solo pueden canjear puntos de lealtad si tienen el saldo necesario.

**Cine y Salas:**

* Cada cine debe tener al menos una sala disponible.
* Las salas no pueden tener asientos duplicados (mismo número y fila).

**Asientos:**

* Un asiento solo puede ser marcado como reservado si está disponible.
* No se permite sobrepasar la capacidad de la sala con asientos reservados.

**Programa de Lealtad:**

* Los clientes solo pueden canjear puntos si tienen suficientes acumulados.
* Los niveles de membresía dependen de la cantidad de puntos acumulados.