**Análisis de Problemas con POO: Aplicación para un Cine**

Cristian Franco Raigosa, Santiago López Ramírez, Mariana Pineda Isaza

Facultad de Ingenierías, Universidad de Medellín

Algoritmos y Programación Orientada a Objetos

Jesús Andrés Hincapié Londoño

15 de septiembre de 2022

1. **Descripción del problema**

Se requiere hacer una aplicación para un cine que ofrece películas y comestibles. Si se desea hacer uso de la aplicación, el usuario se debe **registrar** en el sistema para poder ingresar. Los usuarios deberán **iniciar sesión** colocando su cédula y clave.

El cine ofrece diferentes películas que tienen un nombre, una duración, un género y una

sinopsis. Además, ofrece comestibles que tienen nombre, descripción, precio unitario y

unidades disponibles. El usuario podrá **visualizar estadísticas** acerca de cuál es la

película y el comestible más y menos: vendido, caro. También, podrá **reservar** la película a la que desea asistir, indicando la hora y la cantidad de boletas para luego elegir los asientos. Los usuarios podrán **agregar comestibles a una bolsa** indicando el producto(nombre) y la cantidad, cuando esto suceda, se debe verificar la disponibilidad. De igual manera, tendrán la opción de **eliminar un ítem de la bolsa**. Dicha bolsa, mostrará el total de cada ítem que contenga y el valor total a pagar.

La aplicación debe permitir al usuario **comprar los elementos presentes en la bolsa** y agregar el valor de la venta al total acumulado de la tienda.

Antes de realizar la compra, se deben **aplicar descuentos** al usuario si es posible:

* Descuento del 15% por compras mayores a 150.000
* Descuento del 10% por compras mayores a 90.000

(Los descuentos no son acumulables, se le aplicará el de más alto porcentaje).

Por otro lado, la aplicación debe permitir ingresar como administrador para **actualizar la agenda del día**, es decir, para programar las películas que se presentarán en una sala a lo largo del día, a esta sala se asocia una película a cierta hora con un precio de boleta. La **información** de los **comestibles** se debe **cargar de un archivo.**

1. **Análisis del problema**

* Definición de los requisitos funcionales

|  |  |
| --- | --- |
| **Nombre** | R01 – Registrar usuario |
| **Resumen** | Un usuario del cine puede acceder a los beneficios de la aplicación a través de un breve registro. |
| **Entradas** | * cedula * nombre * clave |
| **Resultado** | 1. Buscar usuario por su cedula:  * Si se encuentra el usuario:   1. Informar al usuario que ya está registrado, es decir, ya tiene una cuenta existente. * Si no encuentra el usuario:   1. Creación de la cuenta:      1. Ingresar al diccionario usuarios, como clave la cedula y como valor el nombre.      2. Informar al usuario que la creación de la cuenta fue exitosa. |

|  |  |
| --- | --- |
| **Nombre** | R02 – Iniciar sesión |
| **Resumen** | Un usuario del cine puede acceder a su cuenta ya existente a través de un breve ingreso donde se requiere la cedula y el nombre. |
| **Entradas** | * cedula * clave |
| **Resultado** | 1. Mostrar opciones de ingreso:  * Ingresar como administrador:   1. Se le pide al usuario la clave:      1. Si es correcta se le da ingreso a la cuenta como administrador.      2. Si no es correcta sale advertencia de contraseña invalida y devuelve al menú para ingresar opción de iniciar sesión. * Ingresar como usuario:   1. Buscar usuario por su cedula:      1. Si se encuentra el usuario:  1. Se le pide al usuario su cedula y contraseña. 2. Validar información:  * Si es correcto: * Ingresar al menú * Si no es correcto: * Informar que ingresó contraseña errónea. * Volver al menú de iniciar sesión   + 1. Si no lo encuentra:  1. Se informa sobre la cuenta no existente.  * Salir   1. Volver al menú inicial. |

|  |  |
| --- | --- |
| **Nombre** | R03 – Ver estadísticas |
| **Resumen** | Permite al usuario visualizar ciertas estadísticas arrojadas por las ventas. |
| **Entradas** | * comestibles * películas * salas de cine(funciones) |
| **Resultado** | (Comparativos)   1. Calcular película más vendida.  * Mostrar Película más vendida.  1. Calcular película menos vendida.  * Mostrar película menos vendida.  1. Calcular película con entradas más caras.  * Mostrar película con entradas más cara.  1. Calcular película con entradas más barata.  * Mostrar película con entradas más barata.  1. Calcular comestible más vendido.  * Mostrar comestible más vendido.  1. Calcular comestible menos vendido.  * Mostrar comestible menos vendido.  1. Calcular comestible más caro.  * Mostrar comestible más caro.  1. Calcular comestible más barato.  * Mostrar comestible más barato. |

|  |  |
| --- | --- |
| **Nombre** | R04 – Agregar comestible a bolsa |
| **Resumen** | Un usuario puede agregar los comestibles que desea a su bolsa indicando el nombre del producto y la cantidad. |
| **Entradas** | * nombre (producto) * cantidad |
| **Resultado** | * + - 1. Buscar comestible por nombre: * Si se encuentra:   1. Verificar si hay suficientes unidades.      1. Si hay suficientes unidades:         + - Agregar comestible a bolsa      2. Si no hay suficientes unidades:         + - Informar al usuario que no hay unidades suficientes. * Si no se encuentra:   1. Informar al usuario que no hay un producto con este nombre. |

|  |  |
| --- | --- |
| **Nombre** | R05 – Reservar |
| **Resumen** | Un usuario puede reservar la película a la que desea asistir. |
| **Entradas** | * nombre * selección asientos |
| **Resultado** | 1. Buscar película por su nombre en la agenda del día:  * Si se encuentra:   1. Verificar si hay asientos disponibles:      1. Si hay asientos disponibles: * Asignar reserva de los asientos a la bolsa del usuario.   + 1. Si no hay asientos disponibles: * Informar sobre ausencia de asientos en la sala. * Si no se encuentra:   1. Informar al usuario de que no hay una película con este nombre. |

|  |  |
| --- | --- |
| **Nombre** | R06 – Eliminar ítem de la bolsa |
| **Resumen** | Un usuario puede eliminar un ítem de la bolsa si así lo desea, seleccionando el ítem. |
| **Entradas** | * item |
| **Resultado** | 1. Seleccionar item 2. Eliminar el item seleccionado 3. Actualizar el total de la bolsa |

|  |  |
| --- | --- |
| **Nombre** | R07 – Aplicar descuentos |
| **Resumen** | Se aplicará descuentos a los usuarios por compras mayores a 90.000 y 150.000. |
| **Entradas** | * valor\_total |
| **Resultado** | 1. Calcular si el valor total de la bolsa es mayor o igual a 90.000:  * Si se cumple:   1. Ver si es menor s 150.000 y si sí lo es:      1. Al valor total descontarle el 10%      2. Informarle al usuario sobre el descuento adquirido.   2. Si no es menor a 150.000:      1. Al valor total descontarle el 15%      2. Informarle al usuario sobre el descuento adquirido. * Si no cumple:   1. No descontar nada y continuar. |

|  |  |
| --- | --- |
| **Nombre** | R08 – Comprar bolsa |
| **Resumen** | Permite al usuario comprar los productos que ha seleccionado. |
| **Entradas** |  |
| **Resultado** | 1. Verificar si la bolsa contiene items:  * Si sí contiene:  1. Calcular el total de la bolsa. 2. Acumular el total del cine. 3. Descontar las unidades vendidas de las unidades disponible de cada producto presente en la bolsa. 4. Limpiar la bolsa.  * Si no contiene:  1. Informar al usuario sobre la imposibilidad de realizar la compra. |

|  |  |
| --- | --- |
| **Nombre** | R09 – Actualizar agenda |
| **Resumen** | El administrador podrá actualizar la agenda, programando así las películas del día. |
| **Entradas** | * película |
| **Resultado** | 1. Abrir archivo donde se encuentra la programación del día. 2. Leerlos la información del archivo y crear los objetos de clase Película. 3. Agregar cada objeto de la clase Película a la lista de películas del cine. |

­

|  |  |
| --- | --- |
| **Nombre** | R10 – Cargar información de comestibles |
| **Resumen** | Permite subir la información de los comestibles que ofertará el cine desde un archivo. |
| **Entradas** | * Objetos comestibles |
| **Resultado** | 1. Creación del objeto comestible 2. Almacenarlo en un diccionario:  * Clave: nombre del producto. * Valor: objeto comestible. |

* Prototipos de la interfaz gráfica

Prototipo de interfaz de los requisitos funcionales realizado en Canva (adjuntamos PDF)

* Modelo del mundo del problema

Diagrama de clases realizado en Diagrams (adjuntamos un PDF para visualizarlo mejor).

1. **Implementación del modelo del mundo**

* Descomposición de los requisitos funcionales y asignación de responsabilidades

Los requisitos implementados son: Registrar usuario, iniciar sesión, agregar comestibles a la bolsa y eliminar item de la bolsa.

**Método: registrar\_usuario(cedula, nombre, clave)** Responsable: Cine

|  |  |  |
| --- | --- | --- |
| **Paso** | **Método (si aplica)** | **Responsable** |
| Buscar usuario por su cedula | buscar\_usuario(cedula) -> usuario | Cine |
| Creación de la cuenta |  | Cine |
| Ingresar al diccionario la cedula como clave y nombre como valor |  | Cine |

**Método: ingresar\_cuenta()** Responsable: Cine

|  |  |  |
| --- | --- | --- |
| **Paso** | **Método (si aplica)** | **Responsable** |
| Mostrar opciones |  | ui |
| Ingresar opción | iniciar\_sesion\_opcion(self,opcion: str) -> Optional[str]: | Cine |
| Ingresar como administrador | iniciar\_sesion\_admin(self, clave: str) -> bool: | Cine |
| Ingresar como usuario | iniciar\_sesion\_usuario(self,cedula: str, clave:str) -> Optional[int]: | Cine |

**Método: agregar\_comestibles\_bolsa(nombre,cantidad)** Responsable: Cine

|  |  |  |
| --- | --- | --- |
| **Paso** | **Método (si aplica)** | **Responsable** |
| Buscar comestible por nombre. | buscar\_comestible(nombre) | Cine |
| Verificar si la cantidad ingresada es menor a unidades disponibles. | unidades\_disponibles(cantidad) | Comestible |
| Agregar comestible a la bolsa | agregar\_comestible\_bolsa(comestible, cantidad) | Usuario |

**Método: reservar\_pelicula(nombre)** Responsable: Cine (no implementado)

|  |  |  |
| --- | --- | --- |
| **Paso** | **Método (si aplica)** | **Responsable** |
| Buscar película | buscar\_pelicula(nombre) | Cine |
| Verificar disponibilidad de asientos | hay\_asientos­\_disponibles() | Sala |
| Asignar la reserva la bolsa | agregar\_reserva\_a\_bolsa() | Cine |
| Informar si no hay asientos |  | Cine |

**Método: eliminar\_item(item)** Responsable: Cine

|  |  |  |
| --- | --- | --- |
| **Paso** | **Método (si aplica)** | **Responsable** |
| Buscar item | buscar\_item(item) | Cine |
| Eliminar item | eliminar\_item(indice) -> bool | Usuario |

**Método: actualizar\_agenda()**  Responsable: Cine (no implementado)

|  |  |  |
| --- | --- | --- |
| **Paso** | **Método (si aplica)** | **Responsable** |
| Abrir archivo con la programación | abrir\_archivo\_programacion() | Cine |
| Leer Info y crear objetos de la clase Película | pelicula = pelicula() | Pelicula |
| Agregar objetos a la lista de productos | Productos.append(producto) | Cine |