Icono

Descripción generada automáticamente

**Ejercicio 2**

**PC 3**

**Dilian Anabel HURTADO PONCE**

Semana: 15

**Integrantes:**

Roberto Agustín Mejía Collazos

21/11/24

**Docente:**

***Una Tienda de Libros***

***Libro.java***

***Texto

Descripción generada automáticamente***

***ItemCompra.java***

***Texto

Descripción generada automáticamente***

***CarroObserver.java (Interface)***

***Texto

Descripción generada automáticamente***

***CarroCompra.java***

***Texto

Descripción generada automáticamente***

***MostrarCarroObserver.java***

***Texto

Descripción generada automáticamente***

***MostrarTotalObserver.java***

***Texto

Descripción generada automáticamente***

***Main.java***

***Pantalla de computadora con letras

Descripción generada automáticamente con confianza media***

***Ejecutando el código***

***Texto

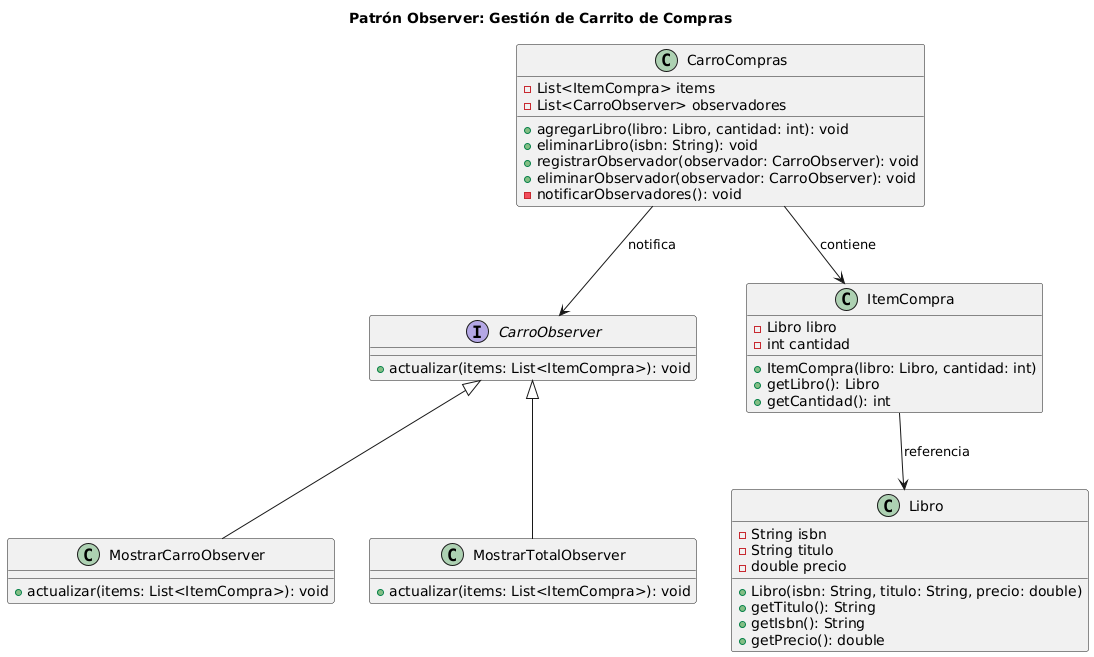
Descripción generada automáticamente***

***Explicación del Código del Patrón Observer***

El patrón Observer permite que el objeto sujeto (en este caso, el CarroCompras) notifique automáticamente a múltiples observadores cada vez que cambie su estado.

Componentes Clave del Código

1. Interfaz CarroObserver: Define el contrato para los observadores. Cualquier clase que implemente esta interfaz puede registrarse para recibir notificaciones del carrito.
2. Clase CarroCompras (Sujeto):
   * Gestiona la lista de observadores que quieren ser notificados.
   * Tiene métodos para agregar y eliminar libros del carrito.
   * Cada vez que cambia su estado, llama al método actualizar de todos los observadores registrados.
3. Observadores Concretos:
   * MostrarCarroObserver: Muestra el contenido del carrito actualizado.
   * MostrarTotalObserver: Calcula y muestra el total de la compra.
4. Clase Principal (Main):
   * Crea el carrito y registra los observadores.
   * Simula operaciones como agregar y eliminar libros, desencadenando las actualizaciones automáticas.

***Diagrama UML***

***Explicación del UML***

1. Relaciones Principales:

CarroObserver es la interfaz que define el método actualizar.

MostrarCarroObserver y MostrarTotalObserver implementan CarroObserver como observadores concretos.

CarroCompras mantiene una lista de observadores (CarroObserver) que son notificados mediante el método notificarObservadores().

1. Flujo General:

Cuando cambia el estado del carrito (por ejemplo, se agrega o elimina un libro), el carrito invoca notificarObservadores().

Este método recorre la lista de observadores y llama a su método actualizar, pasando la lista actualizada de items.

1. Ventajas Visuales:

El diagrama muestra cómo los observadores (MostrarCarroObserver, MostrarTotalObserver) están desacoplados del CarroCompras.

Facilita la extensión del sistema agregando nuevos observadores sin modificar la lógica del carrito.