	Carátula para entrega de prácticas
Facultad de Ingeniería	Laboratorios de docencia

Laboratorio de Computación Salas A y B

Profesor(a): René Adrián Dávila Pérez

Asignatura: POO

Grupo: 01

No de Práctica(s): Practica 5 y 6

Integrante(s): 322125337

322080869

322059179

323629814

322113536

No. de lista o brigada: 6

Semestre: 2025

Fecha de entrega: 03/10/2025

Observaciones:

CALIFICACIÓN: _____

Índice

1. Introducción

- **Planteamiento del problema**: La práctica consiste en modificar la aplicación dada por el profesor aplicando encapsulamiento y empaquetamiento y que la aplicación siga funcionando de la misma manera
- **Motivación**: Al realizar un código a escala pequeña no te preocupas normalmente por detalles como empauetamiento y encapsulamiento, pero al momento de realizar algo más grande no puedes omitir estos detalles. Trabajarás con volumenes más grandes de archivos y es necesario que estos esten organizados (empsquetamiento) y tambien es necesario contemplar la integridad y seguridad del código por modificaciones accidentales o por uso indebido (encapsulamiento)
- **Objetivos**: El objetivo principal es desarrollar habilidad para poder proteger y controlar el código aplicando encapsulamiento (principalmente mediante el uso se getters y setters sin modificar las propiedades directaente) y aprender a organizar el codigo mediante empaquetamiento mediante un orden lodigo (el cual en este caso es mx.unam.fi.poo.g1.p56.equipo6)

2. Marco Teórico

Clase: Una clase en Java es un plano o un modelo que define la estructura y el comportamiento de los objetos que se pueden crear a partir de ella, tiene atributos que son las características que describen el estado de un objeto, como el color o la marca de un coche. Igualmente contiene métodos, que son las acciones o comportamientos que el objeto puede realizar, y suelen manipulan los atributos para cambiar el estado del objeto.[3]

Encapsulamiento: Es un mecanismo para reunir datos y métodos dentro de una estructura ocultando la implementación del objeto, es decir, impidiendo el acceso a los datos por cualquier medio que no sean los servicios propuestos. La encapsulación permite, por tanto, garantizar la integridad de los datos contenidos en el objeto.[2]

Paquetes: Son el mecanismo de Java para agrupar clases e interfaces relacionadas en un espacio de nombres jerárquico, el cual corresponde directamente a una estructura de carpetas. Su principal objetivo es organizar el código y prevenir conflictos entre clases con el mismo nombre. El uso de paquetes, por tanto, es fundamental para mantener la modularidad, facilitar la reutilización del código y controlar el acceso a sus elementos.[1]

3. Desarrollo

Primero comenzamos por plantearnos una solución teórica para poder abordar los requisitos principales: encapsulamiento, empaquetado y la conexión entre la vista y el modelo. Dado que la clase Vista no podía ser modificada, la solución teórica fue implementar la lógica de control en la clase MainApp, aunque las que modificamos principalmente fueron las clases Articulo y Carrito en las cuales realizaremos la encapsulación de los datos y tendríamos que modificar las variables y el cómo se solicitan en la clase Carrito.

Una vez que terminamos de plantear la solución teórica, llevamos a cabo las siguientes implementaciones en el código fuente:

Implementación del Encapsulamiento.
En la clase Articulo.java, los atributos se declararon como privados y se implementaron sus respectivos métodos de acceso públicos, lo principal fue que añadimos todos los metodos set y get que necesita la clase que fueron: getNombre, setNombre, getPrecio, setPrecio, ademas de modificamos el constructor de la clase para que funcionara con los atributos encapsulados.Despues en la clase Carrito se colocó como privada y final el ArrayList que fungirá como el carrito y donde se guardarian los artículos por eso sería final para que este no pueda ser reemplazado ya que el punto es que solo haya un carrito y cambiar llamadas a atributos que antes eran públicos utilizando getNombre y getArticulos con lo que el programa podría estar encapsulado y seguir funcionando normalmente.

Implementación del Empaquetado.
Preparamos las carpetas en las que íbamos a meter a nuestros archivos que serian las 7 que se nos comento en clase que colocaremos: mx/unam/fi/poo/g1/p56/equipo6 una vez que ya habíamos colocado los archivos dentro de las carpetas asignamos en cada uno de los códigos el paquete en el que se encontraba: package mx.unam.fi.poo.g1.p56.equipo6; ya con esas indicaciones las clases de la aplicación quedaron empaquetadas.

Finalmente ejecutamos la aplicación para verificar que todos los cambios realizados al momento de empaquetar y encapsular los datos no afecten al código y se pueda seguir accediendo a los datos por lo que después de esto logramos el objetivo que se nos solicitó para la aplicación.

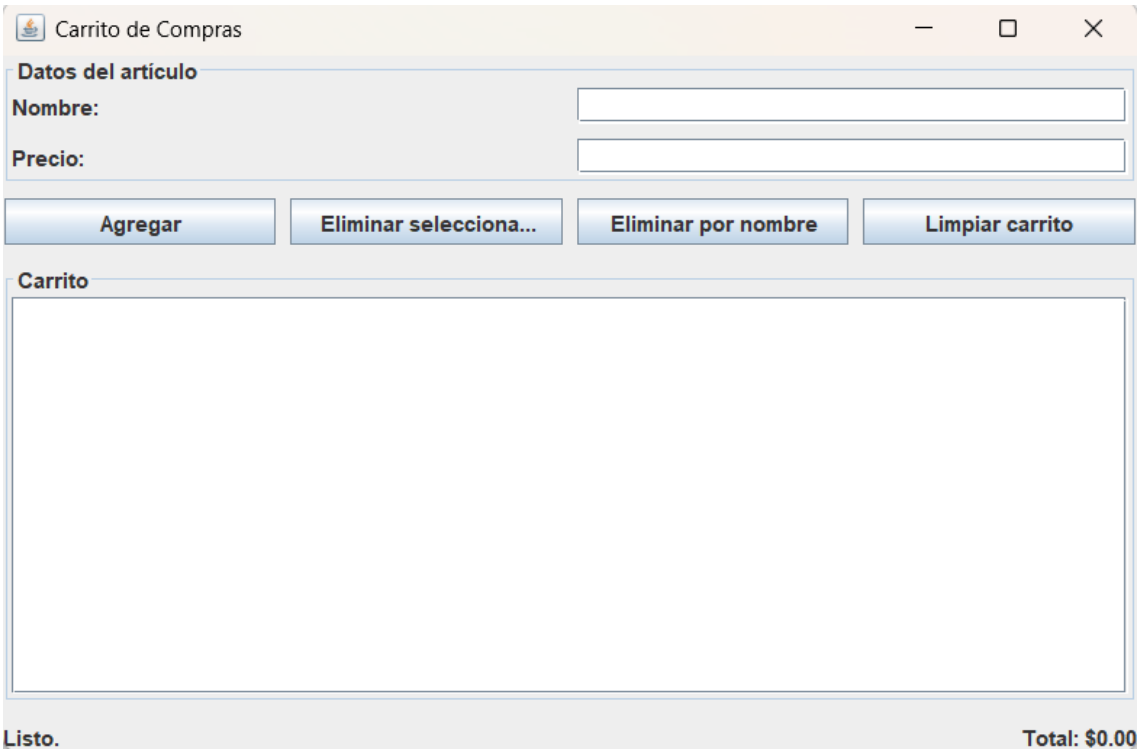
4. Resultados

Se comprueba la correcta ejecución de cada una de las funciones de la aplicación.

Compilación y ejecución desde terminal:

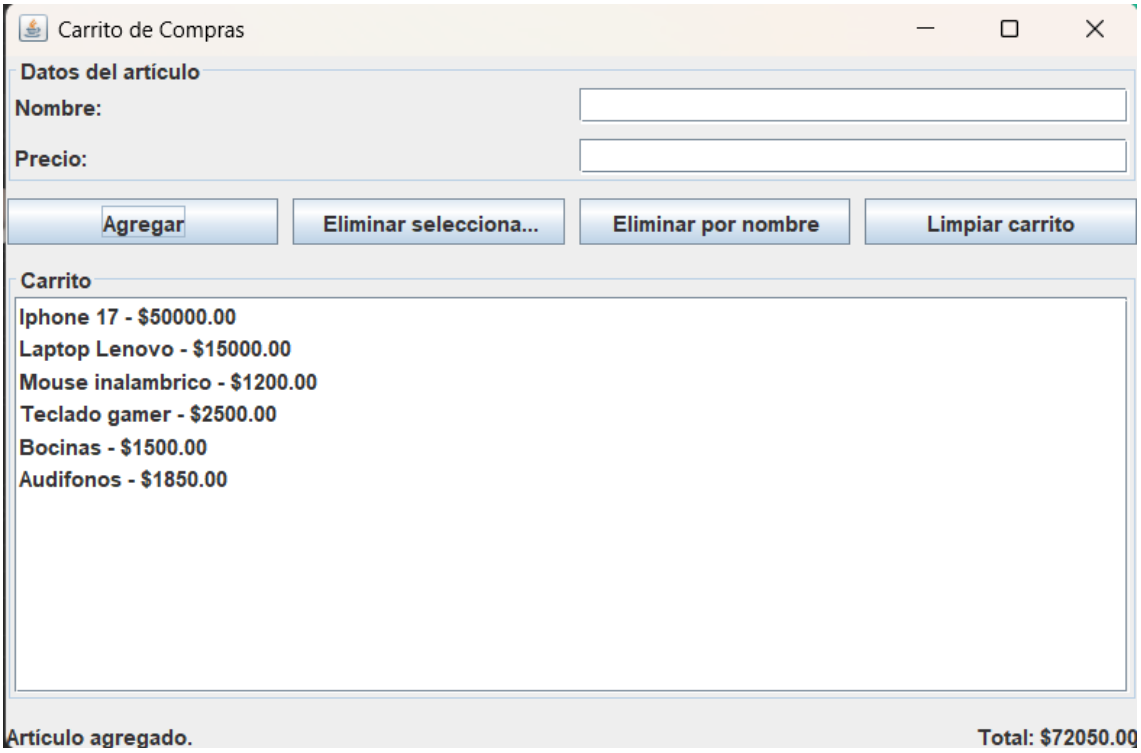
```
PS C:\Users\irvin\Documents\Nueva carpeta\RepoPersonal> javac -d bin mx\unam\fi\poo\g1\p56\equipo6\*.java
PS C:\Users\irvin\Documents\Nueva carpeta\RepoPersonal> java -cp bin mx.unam.fi.poo.g1.p56.equipo6.MainApp
```

Estado inicial de la aplicación:



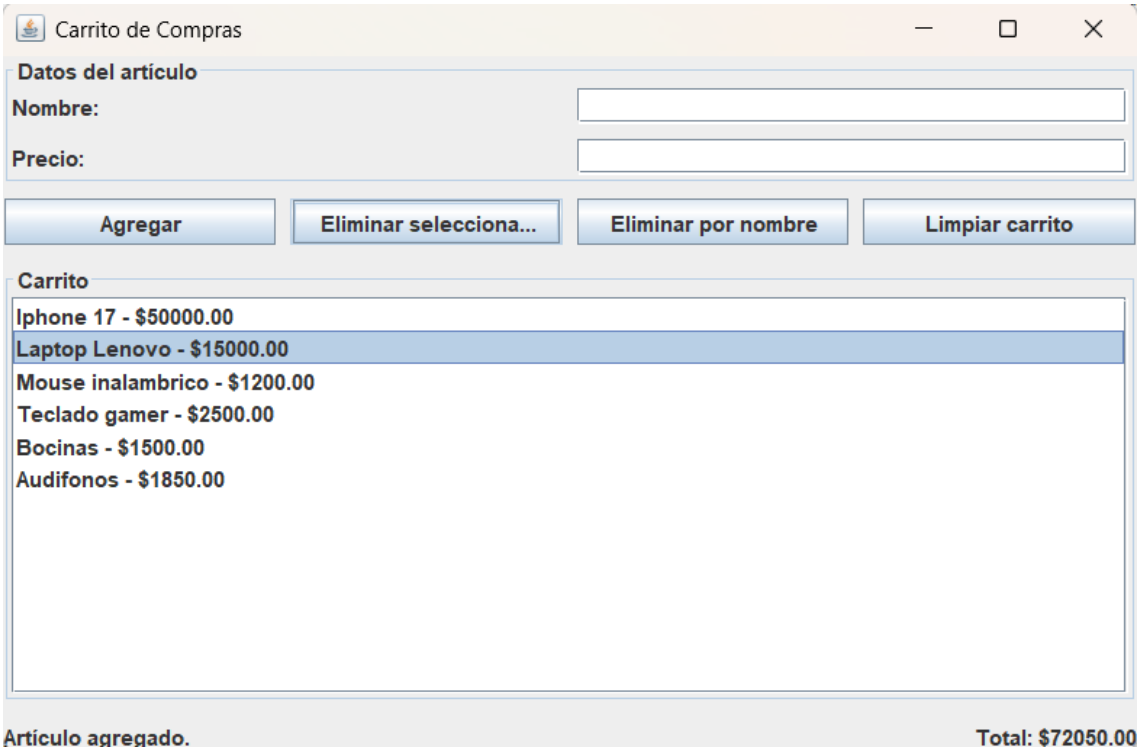
- En esta parte se muestra el inicio de la aplicacion despues de la ejecución desde terminal. Por lo tanto, todos los campos de la aplicación aparecen vacios.

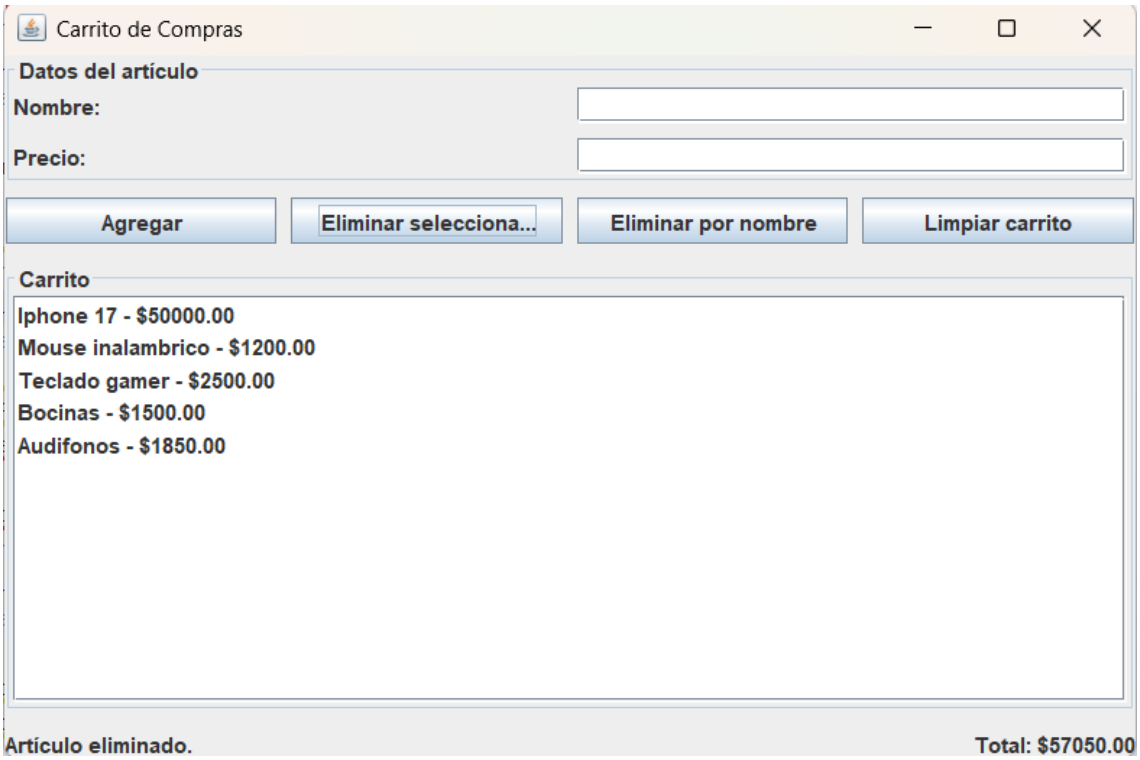
Añadiendo artículos al carrito:



- En esta parte se demuestra la función de la aplicacion para agregar objetos al carrito. Primero se agrega el nombre y el precio del articulo y posteriormente se selecciona el botón de agregar.

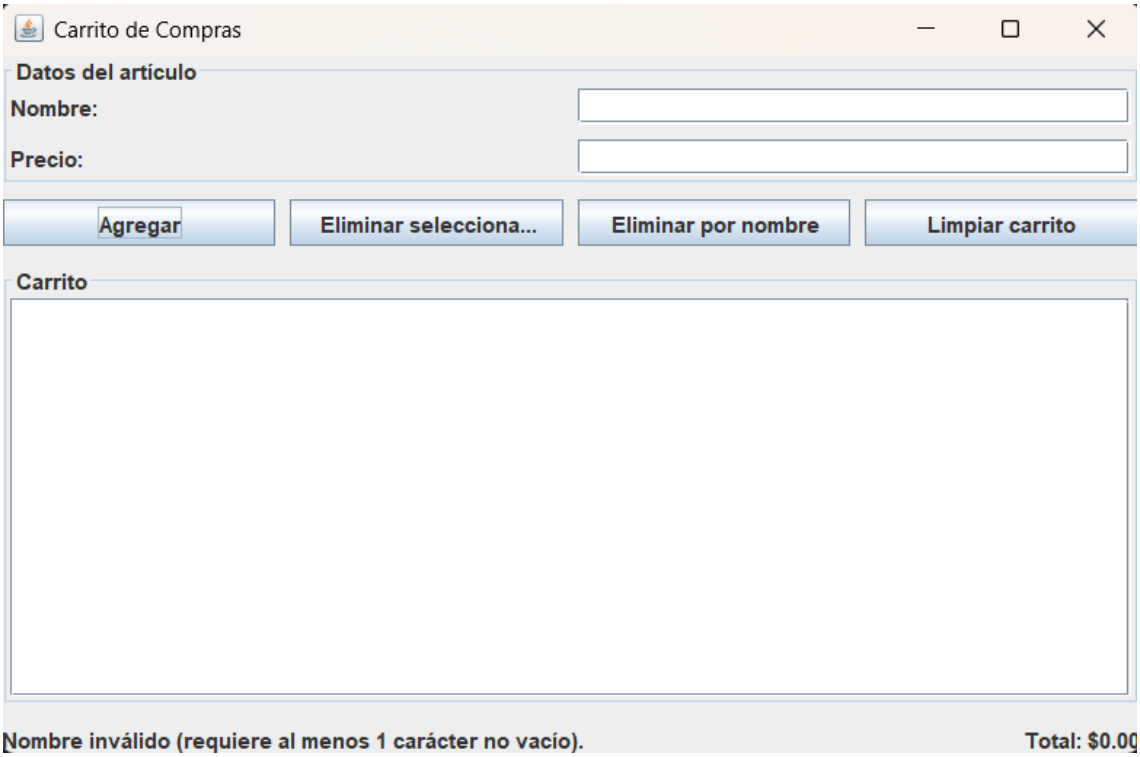
Eliminando un artículo:





- En esta parte se demuestra la función de la aplicación para eliminar artículos, lo primero que se tiene que hacer es seleccionar el artículo a eliminar y después hacer click en el botón de eliminar seleccionados.

Entrada inválida:



- En esta imagen podemos observar que se intentó agregar un artículo sin nombre o con un precio no numerico, por ende, la aplicación mostró un mensaje de error en la parte inferior.

5. Conclusiones

La finalización de este proyecto demuestra cómo la aplicación de conceptos teóricos como el encapsulamiento y los paquetes son necesarios para la construcción de software mas robusto y que sea mas sencillo realizar mantenimiento. La implementación del encapsulamiento en la clase Articulo, al declarar sus atributos como privados y gestionar el acceso a través de métodos públicos, fue lo que fue clave para garantizar la integridad de los datos en el codigo, protegiéndolo de modificaciones indebidas que podrían causar errores en la aplicación.

El uso de paquetes no solo cumplió con un requisito de nombramiento, sino que estableció un espacio de nombres jerárquico que previene colisiones y organiza el proyecto de una manera escalable. El correcto funcionamiento del carrito de compras es la evidencia de que la aplicación respeta los principios de la Programación Orientada a Objetos vistos en la clase que permite construir sistemas donde los componentes interactúan de forma predecible y segura, reforzando la base para el desarrollo de aplicaciones más complejas.

6. Referencias bibliográficas

[1] Alarcón, J. M. (2021, 27 de septiembre). Paquetes en Java: qué son, para qué se utilizan y cómo se usan. campusMVP. https://www.campusmvp.es/recursos/post/paquetes-en-java-que-son-para-que-se-utilizan-y-como-se-usan.aspx

[2] David. (2025, 21 de agosto). Encapsulación: definición e importancia. DataScientest. https://datascientest.com/es/encapsulacion-definicion-e-importancia

[3] Mercader, J. J. (2020, 16 de junio). Introducción a POO en Java: objetos y clases. OpenWebinars. https://openwebinars.net/blog/introduccion-a-poo-en-java-objetos-y-clases/