**Universidad de las Américas**

**Taller 6**

**Daniela Muñoz**

**Isaac Cordero**

**Matías Albuja**

**Flavio Ibujes**

**Joseph Lema**

**Elian Hidalgo**

**FICA**

**Programación III**

**NRC:**

**16-11-2025**

**Introducción**

Este proyecto implementa un sistema simple para una tienda en línea que maneja productos y ventas mensuales.
Además, muestra el uso de **métodos de búsqueda** y del concepto de **interfaces y clases en Java**.

Las funcionalidades incluyen:

- Registro de ventas
- Actualización del precio de productos
- Búsqueda por ID y por nombre
- Mostrar ventas de los últimos 3 meses
- Uso de búsqueda binaria para editar productos

---

**1. Clase Producto**

Representa un producto de la tienda con los atributos:

- id
- nombre
- precio

**Métodos:**

**getId(), getNombre(), getPrecio()**

Retornan los valores actuales del producto. Permiten acceder a los atributos encapsulados.

**setId(), setNombre(), setPrecio()**

Permiten modificar los atributos del producto.
Se usan especialmente cuando se desea **actualizar el precio** del producto desde la interfaz gráfica.

**toString()**

Retorna un texto con todos los datos del producto.
Es útil para mostrar resultados en la interfaz y en búsquedas.

---

**2. Clase Tienda**

Administra los productos y el registro de ventas.

**Método iniciar()**

```
public void iniciar() {
   listado.add(new Producto(1, "Levadura", 1.3f));
   listado.add(new Producto(2, "Harina", 0.50f));
   listado.add(new Producto(3, "Mantequilla", 1.29f));
}
```

Carga los 3 productos iniciales de la tienda.
Se ejecuta al iniciar la ventana.

---

**Método agregarVenta(Venta venta)**

Agrega un objeto Venta a la lista de ventas.

Este método se invoca desde el botón **"Agregar"** cuando el usuario registra una venta.

---

**Método obtenerProductos()**

Retorna la lista completa de productos.
Es usado por la interfaz para mostrar productos y para realizar búsquedas.

---

**Método obtenerVentas()**

Retorna la lista completa de ventas registradas.
Es usado para llenar el JList con las ventas de los últimos 3 meses.

---

**Método buscarEditar(int id, float precio)**

**Este método implementa un algoritmo de búsqueda binaria**.

**¿Qué hace?**

Busca un producto por su ID, y si lo encuentra, actualiza su precio.

**Tipo de búsqueda: Búsqueda Binaria**

**¿Por qué binaria?**

Porque los productos están ordenados por ID:
1, 2, 3

La búsqueda binaria permite encontrar el producto más rápido dividiendo el rango en mitades.

**¿Cómo funciona el código?**

1. Se define un rango inicial (i = 0, s = tamaño-1)
2. Se calcula el punto medio
3. Se compara el ID buscado con el ID en el centro
4. Según la comparación, se descarta la mitad izquierda o derecha
5. Si encuentra la coincidencia → actualiza el precio

---

**3. Clase Venta**

Registra una venta realizada.

**Constructor Venta(int mes, int cantidad, Producto producto)**

Calcula automáticamente el total:

this.total = this.cantidad * producto.getPrecio();

---

**Métodos get y set**

Permiten acceder y modificar los datos de una venta.
Los más importantes:

**getTotal()**

Retorna el valor total calculado de la venta.

---

**toString()**

Convierte toda la información de la venta en texto.
Esto se usa para mostrar ventas en el JList.

---

**4. Clase Ventana (Interfaz gráfica)**

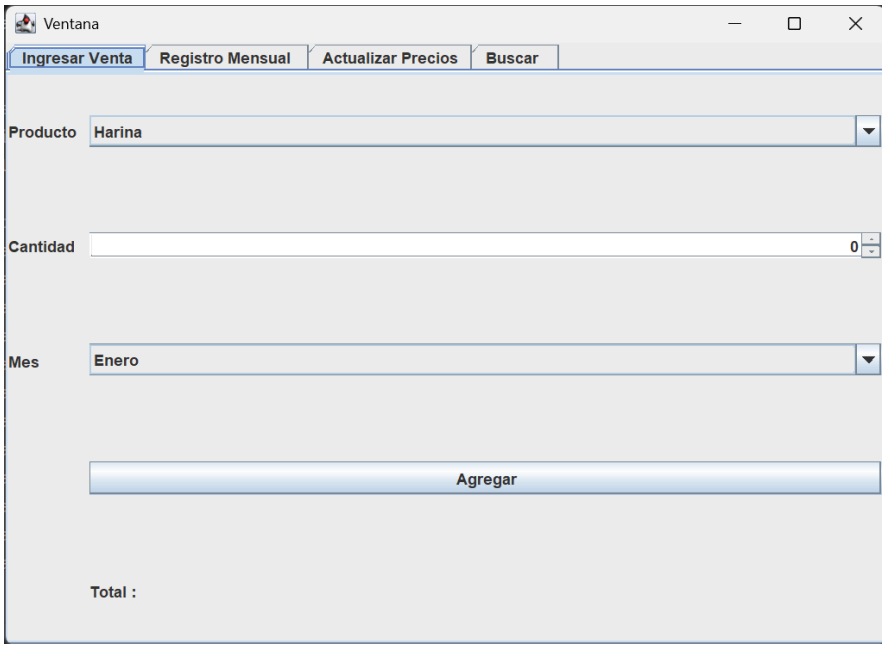
Controla todos los botones y eventos del usuario.

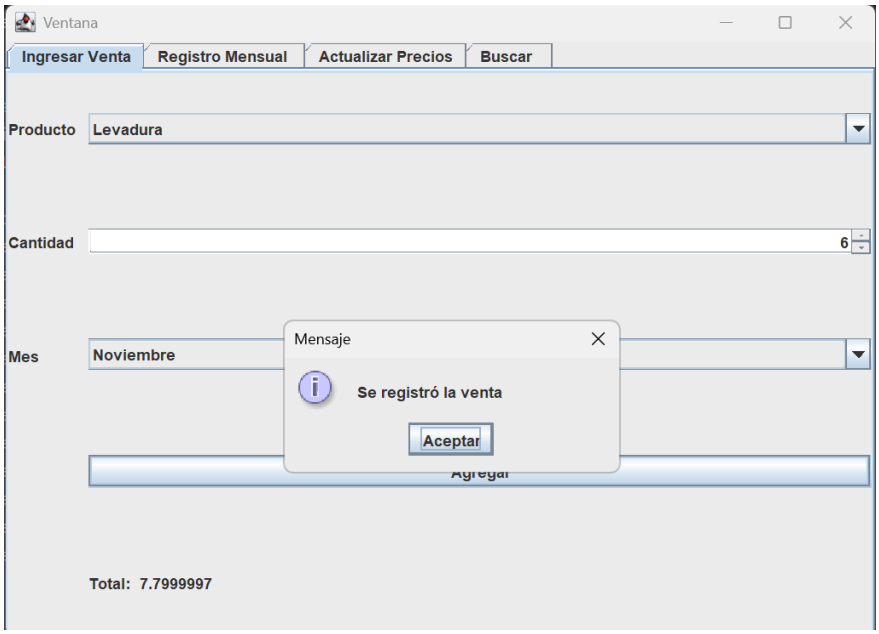
---

**Botón Agregar Venta**

1. Toma los datos ingresados

2. Busca el producto por nombre
3. Crea la venta
4. Actualiza el total en pantalla
5. Guarda la venta

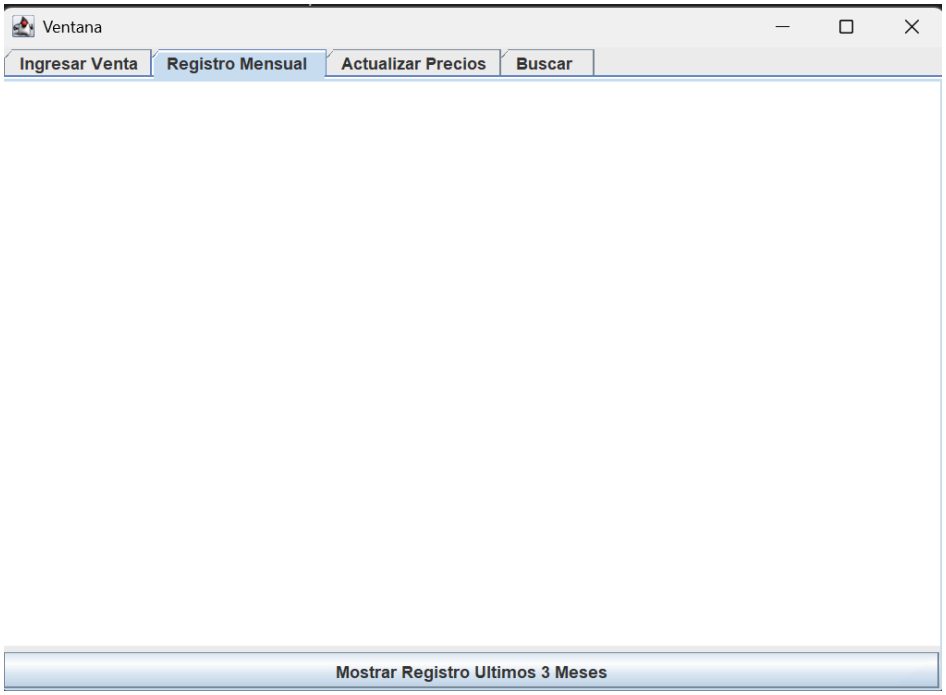


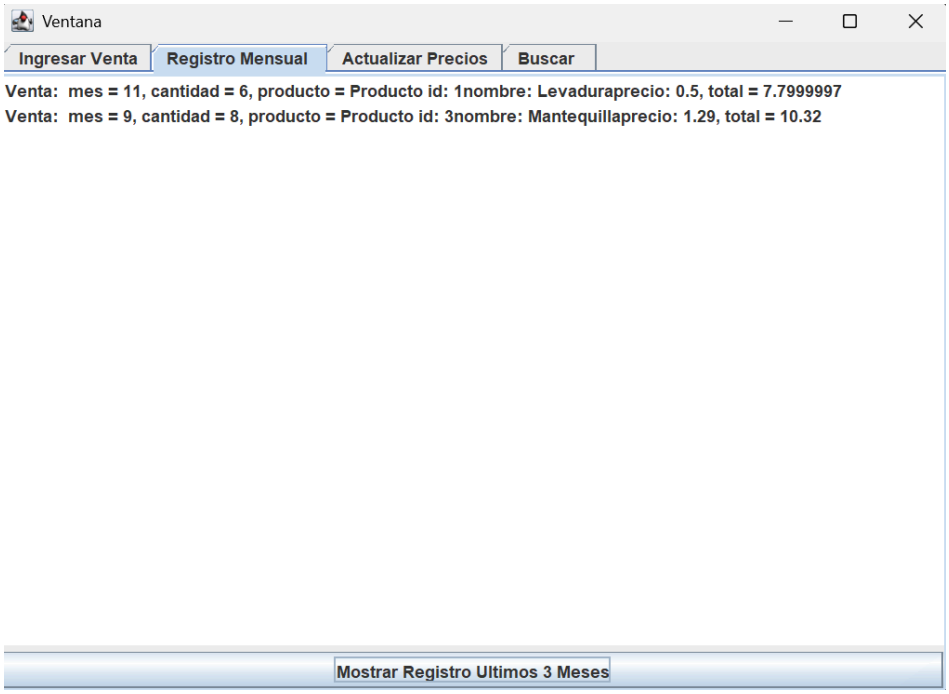



---

**Botón Mostrar Registro Últimos 3 Meses**

Filtra las ventas cuyo mes está entre 9 y 11 (septiembre–noviembre).
Permite cumplir con el requerimiento de la actividad.






---

**Botón Editar Precio**

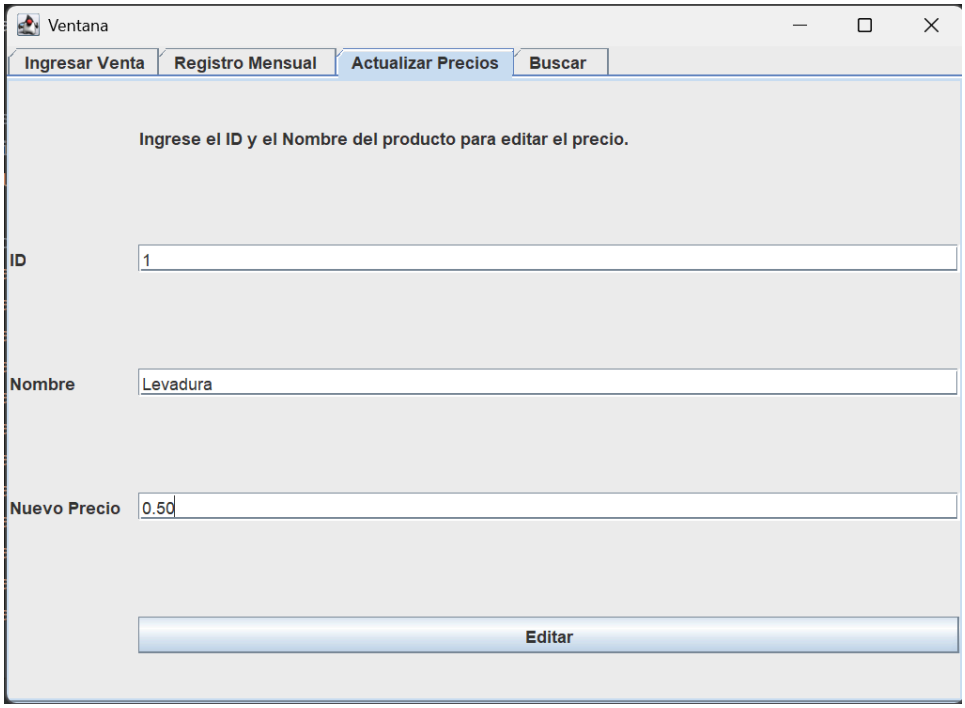
Implementa una **búsqueda secuencial (lineal)**:

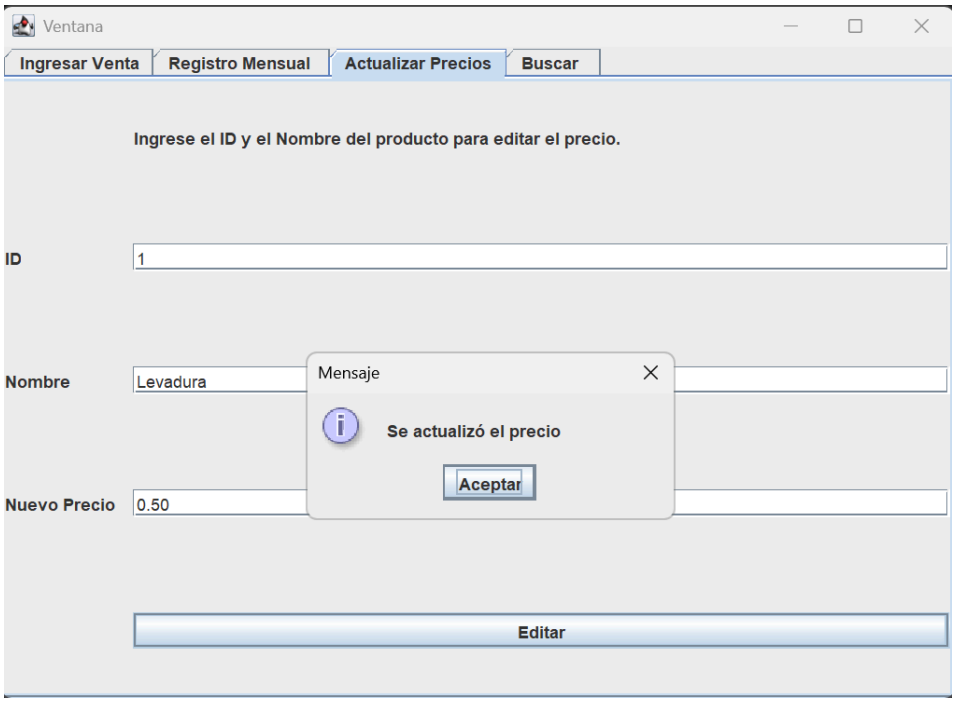
```
for (Producto p : productos) {
   if(id == p.getId() && p.getNombre().equals(nombre)){
      p.setPrecio(nuevoPrecio);
   }
}
```

**Tipo de búsqueda: Lineal / Secuencial**

**¿Por qué es secuencial?**

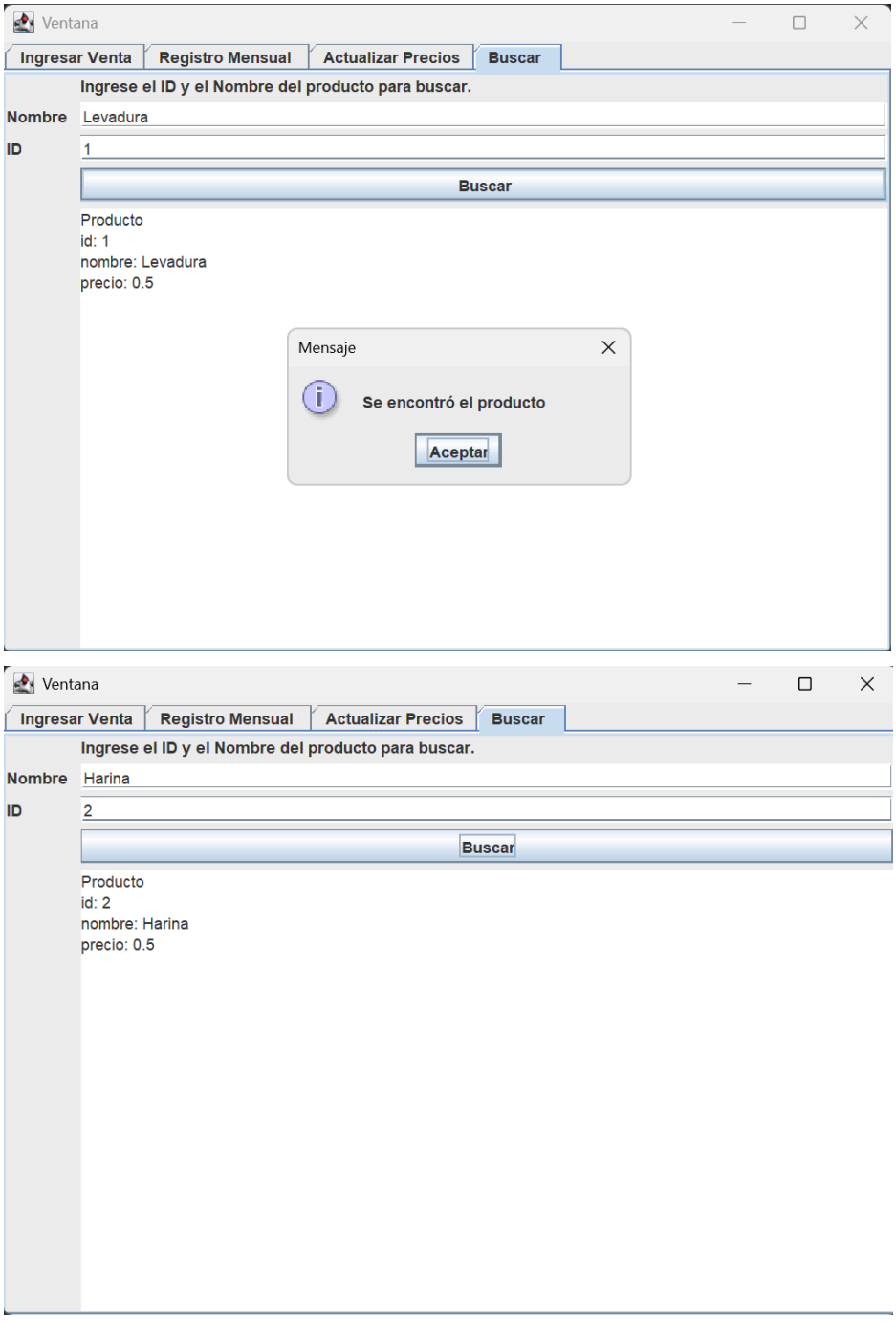
Revisa cada producto uno por uno hasta que encuentres coincidencias.


Ventana
Ingresar Venta
Registro Mensual
Actualizar Precios
Buscar
Ingrese el ID y el Nombre del producto para editar el precio.
ID
1
Nombre
Levadura
Nuevo Precio
0.50
Editar


Ventana
Ingresar Venta
Registro Mensual
Actualizar Precios
Buscar
Ingrese el ID y el Nombre del producto para editar el precio.
ID
1
Nombre
Levadura
Nuevo Precio
0.50
Editar
Mensaje
Se actualizó el precio
Aceptar

### Botón Buscar Producto

Usa también **búsqueda secuencial** por ID y nombre.



---

**MÉTODOS DE BÚSQUEDA IMPLEMENTADOS**

| Funcionalidad | Tipo de búsqueda | Clase | Método |
|---|---|---|---|
| Editar precio por ID | Búsqueda binaria | Tienda | buscarEditar |
| Buscar producto (botón Buscar) | Búsqueda lineal | Ventana | btnBuscar |
| Editar precio (botón Editar) | Búsqueda lineal | Ventana | btnEditar |

---

**Conclusión**

La elaboración de este proyecto permitió aplicar de manera práctica distintos métodos de búsqueda, específicamente la búsqueda lineal y la búsqueda binaria, integrándolos dentro de un sistema funcional. Asimismo, se reforzó el uso adecuado de conceptos fundamentales de programación orientada a objetos, como clases, objetos y estructuras de datos dinámicas. Finalmente, se logró gestionar correctamente operaciones esenciales del sistema, como el registro de ventas, la administración de productos, el filtrado por fechas y la actualización de información, todo esto implementado a través de una interfaz gráfica construida con Swing en Java.

Link GitHub

Isaac Cordero: https://github.com/Is44c2005/Taller-6.git