UNITEC

PROGRAMACION I

SECCIÓN 899

DOCUMENTACIÓN PROYECTO FINAL

Ing. Darío Cardona Aguilar

Alumna: Isabella Esther Dubón Alvarenga

Número de Cuenta: 12541427

Fecha de entrega: 14 de diciembre del 2025

Mini Tienda: Sweet Flame

Clases

-Usuario (guarda el progreso de cada usuario)

-Producto (arreglos para manejar stock)

-Pedido (para generar aleatoriamente los pedidos de los clientes)

-Factura (guarda cada pedido entregado correctamente)

-GestorPedido (contiene los métodos de generar y validar pedido)

Funciones:

Jugar

Empieza el simulador

Nuevo Usuario

Agrega un usuario

Como Jugar

Pequeña descripción del objetivo y funciones del juego.

Salir

Cierra las ventanas

Al seleccionar “Jugar”:

Se pide seleccionar al usuario.

Utiliza un ArrayList<Usuario> para generar las opciones

Validaciones:

-El ArrayList no puede estar vacío antes de continuar (JOptionPane para mostrar mensaje de advertencia)

Al seleccionar “Nuevo Usuario”:

Pide ingresar un nombre de usuario.

Usa el mismo ArrayList<Usuario> y el botón de Guardar para actualizar la lista y la ventana Elegir Usuario.

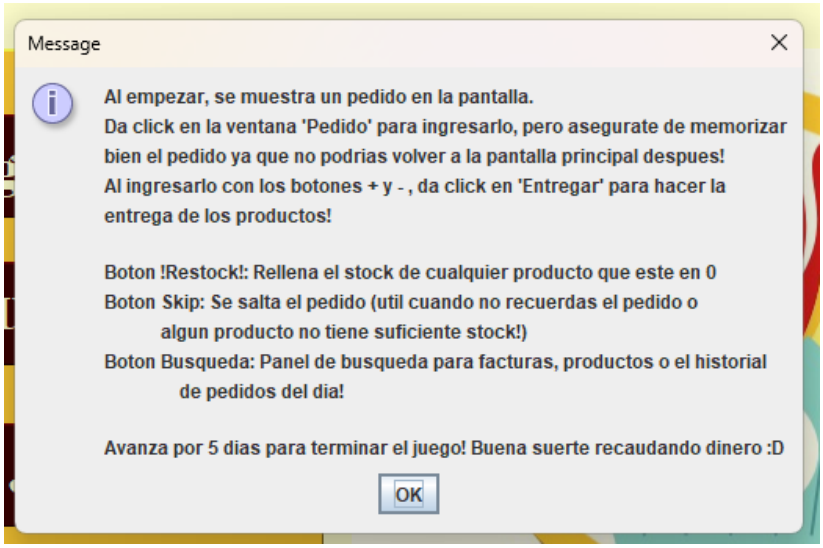
Validaciones:

-No puede quedar vacío

-No puede ser mayor a 15 caracteres

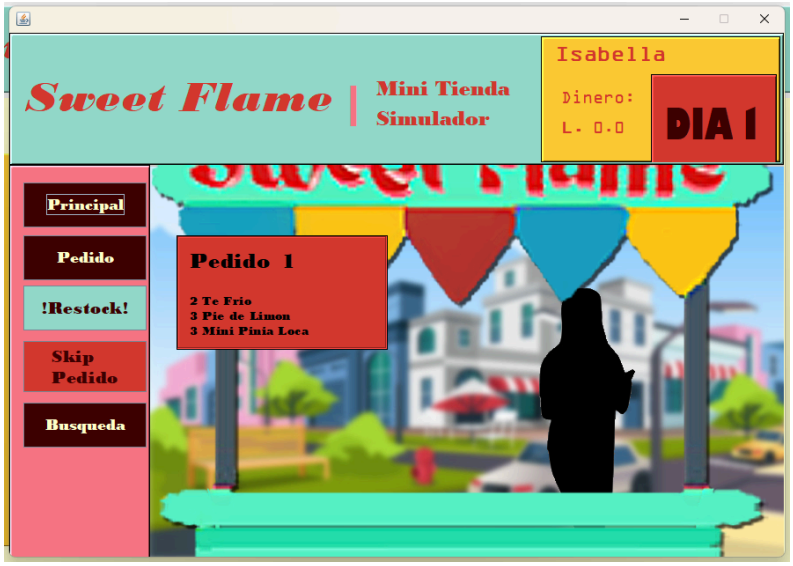
-No se puede ingresar el mismo usuario dos veces.

Al seleccionar “Cómo Jugar”:



Muestra un JOptionPane con el objetivo del juego y las funciones del menú.

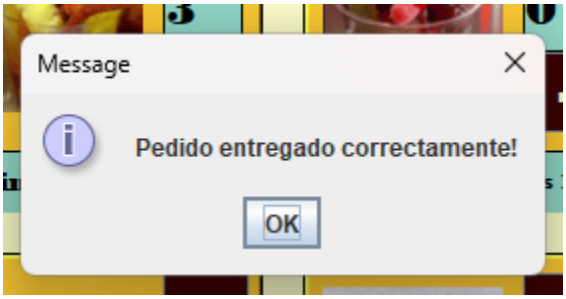
Al seleccionar “Siguiente” en la ventana “Elegir Usuario”:



Al elegir un usuario, se muestra la pantalla principal del juego. El recuadro amarillo muestra el nombre de usuario seleccionado, el dinero recaudado y el día en el que esta (atributos de la clase Usuario). Se genera un png de un cliente y un pedido aleatoriamente (Random) y se muestra en pantalla.

```java
public class GestorPedido {
    public Pedido generarPedido (Usuario usuario, Producto[] tienda, int numPedido ){
        ArrayList<Producto> productos = new ArrayList<>();
        ArrayList<Integer> cantidad = new ArrayList<>();
        Random random = new Random();
        int cantPedidos = random.nextInt(1, 5); //cantidad random de productos 1-4
        while (productos.size() < cantPedidos){
        int index = random.nextInt(tienda.length);
        Producto producto1 = tienda[index];
            if (!productos.contains(producto1)){
            productos.add(producto1);
            //elige una cantidad del producto de forma random 1-3
            int cant = random.nextInt(1, 4);
            cantidad.add(cant);
            }
        }
        Pedido pedido = new Pedido(productos, cantidad, numPedido);
        return pedido;
    }
```

Pantalla “Pedido”:



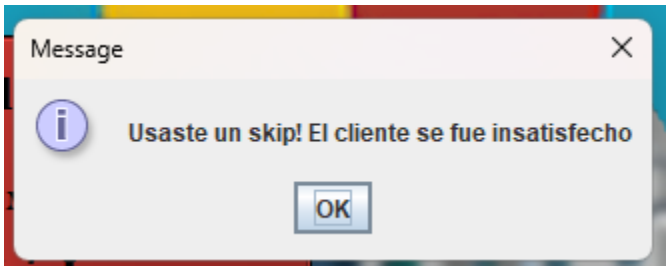
Se debe memorizar el pedido ya que al cambiar a la pestaña Pedido, la pestaña Principal se bloquea. Al ingresar el pedido y dar click en “Entregar”, se hace la comparación del pedido ingresado con el pedido del cliente. Si está correcto, muestra un mensaje JOptionPane y agrega el pedido a las facturas (clase Factura) y al historial de pedidos (matriz boolean) del usuario. Si es incorrecto, permite intentar ingresar los productos de nuevo, lo almacena en el historial como fallido y no crea factura.

Botones:

!Restock!: Busca un producto que tenga el stock en 0 y lo rellena, restando el dinero invertido del perfil del usuario. Muestra el resultado de la inversión en la consola.

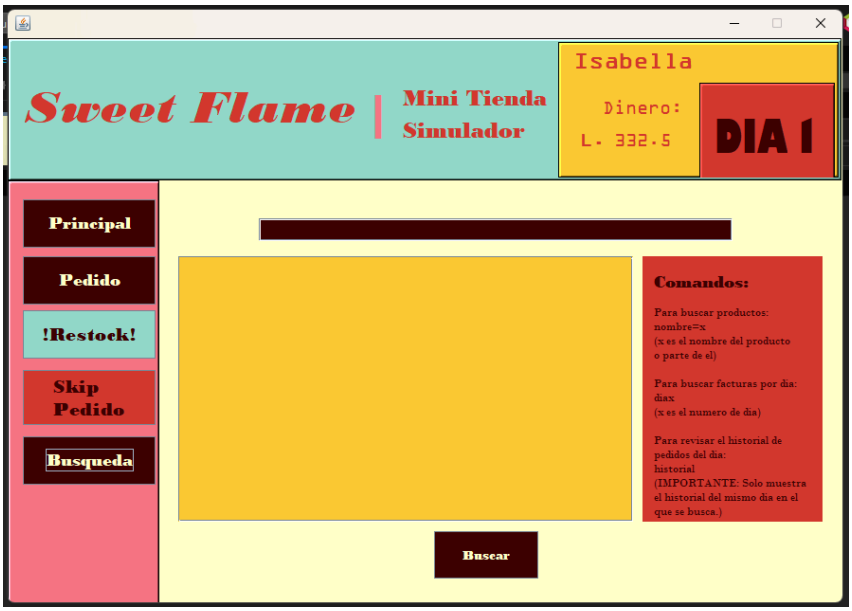
```
private void restockBtnActionPerformed(java.awt.event.ActionEvent evt) {
    //busca los productos vacios y les resetea el stock
    boolean restocked = false;
    double inversionTotal = 0;
    for (int i = 0; i < tienda.length; i++){
        if (tienda[i].getStock() == 0){
            tienda[i].setStock(6);

            double precio = tienda[i].getPrecio();
            double inversionProd = 8 * precio;
            inversionTotal += inversionProd;
            restocked = true;
            System.out.println();
            System.out.println(tienda[i].getProducto()+" restocked: "+tienda[i].getStock());
            System.out.println("-"+inversionTotal);
        }
    }
    if (!restocked){
        JOptionPane.showMessageDialog(this, "No hay stock vacio.");

    }
    seleccion.setDinero(seleccion.getDinero()-inversionTotal);
    actualizarDinero();
}
```

Skip Pedido: Genera el siguiente pedido y muestra un JOptionPane con un mensaje:



No se guarda el pedido ni en el historial.

Busqueda:

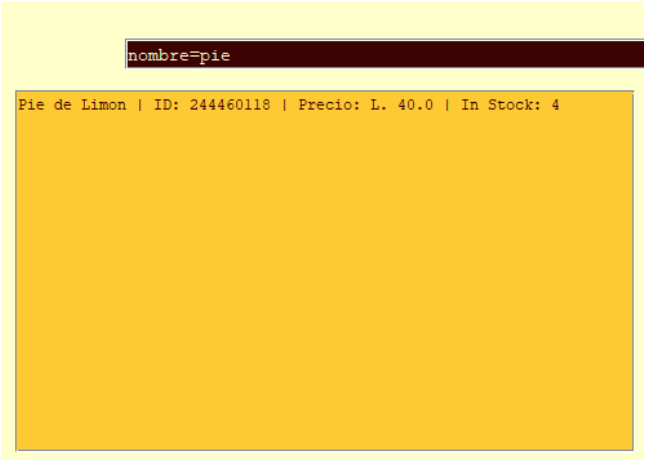


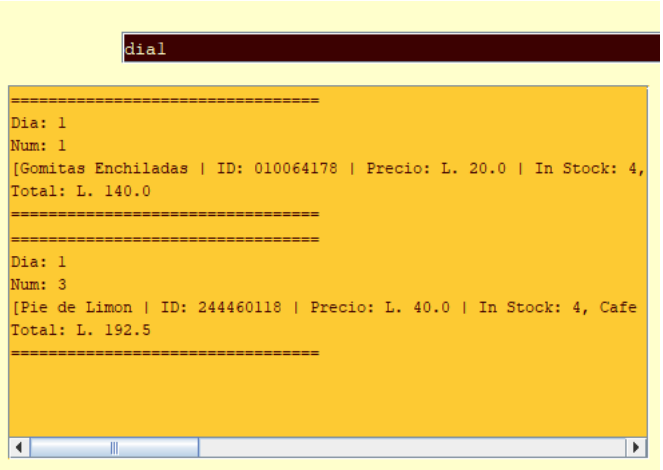
Abre la pestaña de búsqueda con los comandos disponibles.

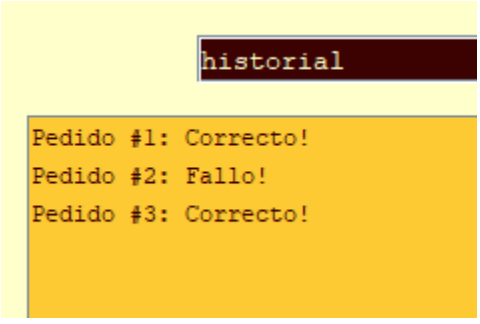
"nombre=x" busca un producto por su nombre completo o parte de el. (metodo recursivo)

"Diax" muestra todas las facturas de un dia especifico.

"Historial" muestra el historial de pedidos correctos o fallidos del dia (se reinicia en cada dia).








```
private void buscarActionPerformed(java.awt.event.ActionEvent evt) {
    facturasUsuario = this.seleccion.getFacturas();
    String busqueda = buscarTiendaTxt.getText();
    if (busqueda.contains("dia")){ //si contiene "dia" (ej. "dia 1")
        modeloTienda.clear(); //borra la lista primero
        String copia = busqueda.replace("dia", "").trim(); //le quita "dia"
        try{
            int dia = Integer.parseInt(copia);
            boolean hayEnDiaX = false;
            for (int i = 0; i < facturasUsuario.size(); i++){
                if (facturasUsuario.get(i).getDia() == dia){ //imprime las facturas del numero de dia ingresado
                    modeloTienda.addElement(facturasUsuario.get(i).toString());
                    hayEnDiaX = true;
                }
            }

            if (!hayEnDiaX){
                modeloTienda.addElement("No hay facturas.");
            }
            mostrar.setModel(modeloTienda);
            mostrar.revalidate();
            mostrar.repaint();
        }catch (NumberFormatException numero){ //si el formato falla
                JOptionPane.showMessageDialog(this, "Formato invalido. Usa: 'dia1', 'dia2', etc.");
            }
```

```
}else if (busqueda.contains("nombre=")){
    modeloTienda.clear(); //borra la lista primero
    String copia = busqueda.replace("nombre=", "").trim(); //le quita "nombre="
    Producto encontrado = buscarProducto(tienda, copia, 0);
    if (encontrado != null){
        modeloTienda.addElement(encontrado.toString());
    }else{
        modeloTienda.addElement("Producto no encontrado.");
    }
    mostrar.setModel(modeloTienda);
    mostrar.revalidate();
    mostrar.repaint();
}else if(busqueda.equals("historial")){
    modeloTienda.clear();
    boolean[][] historial = this.seleccion.getHistorialPedidos();
    int total = this.seleccion.getIndicePedido();

    for (int i = 0; i < total; i++) {
        String simbolo;
        if (historial[i][0]) {
            simbolo = "Correcto!";  //correcto
        } else {
            simbolo = "Fallo!";  //incorrecto
        }
    modeloTienda.addElement("Pedido #" + (i+1) + ": " + simbolo);
    mostrar.setModel(modeloTienda);
    mostrar.revalidate();
    mostrar.repaint();
    }
```

```
    }else{
        JOptionPane.showMessageDialog(this, "Formato invalido. Ej: 'nombre=Pie de Limon' || Usa: 'dia1', 'dia2', etc. || Usa: 'historial'");
    }
}

private Producto buscarProducto(Producto[] tienda, String nombre, int index){
    if (index >= tienda.length){
        return null;
    }
    if (tienda[index].getProducto().toLowerCase().trim().contains(nombre.toLowerCase().trim())){
        return tienda[index];
    }
    return buscarProducto(tienda,nombre,(index+1));
}
```

Al finalizar los 5 días, se muestra esta pantalla y reinicia el perfil del usuario. (formato html para salto de linea)

Fuentes

Code, B. (2020, Septiembre 14). *Java Swing GUI Full Course* ☕. Youtube. https://www.youtube.com/watch?v=Kmgo00avvEw

Code, T. (2021, Julio 28). ☕ *MODELO DE CAPAS: IGU, LOGICA y PERSISTENCIA* 🤓 | *JAVA para PRINCIPIANTES |CURSO COMPLETO 2022* 🚀|*#18*. Youtube. https://www.youtube.com/watch?v=C6J0TOlCieM

Code, T. (2021, Agosto 12). ☕ *INTERFAZ GRÁFICA DE USUARIO* 🎨 *- PARTE 1* 🤓 | *JAVA para PRINCIPIANTES | CURSO COMPLETO 2022* 🚀 | *#19*. Youtube. https://www.youtube.com/watch?v=Fc4uFeMXBS8

Codex, G. (2018, Febrero 24). *Generar o dar saltos de linea en un JLABEL en JAVA*. Youtube. https://www.youtube.com/watch?v=b4N9C8S3PxE

Demand, T. O. (2023, Julio 13). *Easily Add Multiple Panels in One JFrame | Java Swing GUI Tutorial*. Youtube. https://www.youtube.com/watch?v=GPX7hrUvQgw

Ernesto, L. G. d. (2018, Junio 5). *Curso Java Intermedio #36 | Recursividad en Java*. Youtube. https://www.youtube.com/watch?v=jEfmotrL7jQ

Obtener el ancho y alto, escalar y convertir a otro formato imágenes con Java. (2022, Noviembre 25). Blog Bitix. https://picodotdev.github.io/blog-bitix/2022/11/obtener-el-ancho-y-alto-escalar-y-convertir-a-otro-formato-imagenes-con-java/

Tutorials, T. T. (2011, Junio 28). *JOptionPane - Java Basics*. Youtube. https://www.youtube.com/watch?v=E3JQrxxQKIo

w3schools. (n.d.). *Java Exceptions - Try...Catch*. w3schools. https://www.w3schools.com/java/java_try_catch.asp

Zabs, A. (2021, Agosto 30). *JPanel dentro de otro JPanel Java | JFrame*. Youtube. https://www.youtube.com/watch?v=iFlHGwmYdjI