MANUAL TÉCNICO

dojo de karate "Cobra"

Curso	Carné	Nombre
IPC1	201907179	Wilson Manuel Santos Ajcot

I. TABLA DE CONTENIDO

II. INTRODUCCIÓN

El "Código Cobra: Sistema de Gestión de Inventarios y Ventas" es una herramienta diseñada para facilitar la administración de inventarios, el proceso de ventas y la generación de reportes en cualquier negocio. Con funciones para agregar productos, realizar ventas, cargar datos de forma masiva desde archivos CSV y visualizar reportes de productos vendidos e historial de ventas, este sistema simplifica la gestión diaria de recursos.

Inspirado en el espíritu de Cobra Kai y con un enfoque práctico, el sistema es intuitivo, rápido y adaptable, brindando información clave para la toma de decisiones en el ámbito de ventas y control de inventarios.

III. OBJETIVOS

El objetivo principal de este manual es proporcionar a los programadores y estudiantes una comprensión detallada del desarrollo de la aplicación, enfocándose en la implementación de un sistema para la gestión de usuarios, inventarios, ventas y reportes.

El sistema está diseñado para que los usuarios puedan interactuar con la aplicación de forma sencilla, y los programadores puedan aprender a aplicar los principios de la programación en Java para crear aplicaciones funcionales.

Además, el manual ofrece una guía para el uso adecuado de validaciones y manejo de archivos, fundamentales para el desarrollo de este tipo de sistemas.

IV. DIRIGIDO

Este manual está orientado a todas aquellas personas interesadas en entender el funcionamiento y desarrollo de esta aplicación, así como a programadores que deseen utilizar esta práctica como referencia para aplicar métodos y funciones en sus propios proyectos.

El manual proporciona una guía clara sobre cómo se implementaron las diversas funcionalidades del sistema, permitiendo a los usuarios comprender y adaptar las soluciones propuestas.

V. ESPECIFICACIÓN TÉCNICA

- **REQUISITOS DEL HARDWARE**
  - Computadora de escritorio o portátil.
  - Memoria RAM: Mínimo 4 GB.
  - Almacenamiento: Disco duro de 250 GB o superior.
  - Procesador Intel Core i3 o superior
  - Resolución Gráfica: Resolución mínima de 1024 x 768 píxeles

- **REQUISITOS DEL SOFTWARE**
  - Sistema Operativo: Windows 7 o superior.
  - Java Runtime Environment (JRE): Versión 8.2 o superior.
  - Entorno de desarrollo recomendado: IntelliJ IDEA, Eclipse o cualquier IDE compatible con Java.
  - Librería javax.SWING.
  - Librería IO
  - Librería java.util

VI. LÓGICA DEL PROGRAMA

Funcionalidad	Descripción	Métodos/Funciones Contenidas.
Autenticación	Contiene la lógica que gestiona el login de los usuarios y la validación de sus credenciales.	Login
Menú Principal	Muestra el menú principal donde el usuario puede elegir diferentes opciones (agregar producto, ventas, etc.).	menu
Agregar Producto	Contiene la lógica que gestiona el ingreso de producto desde consola y valida que los datos sean correctos.	agregarProducto

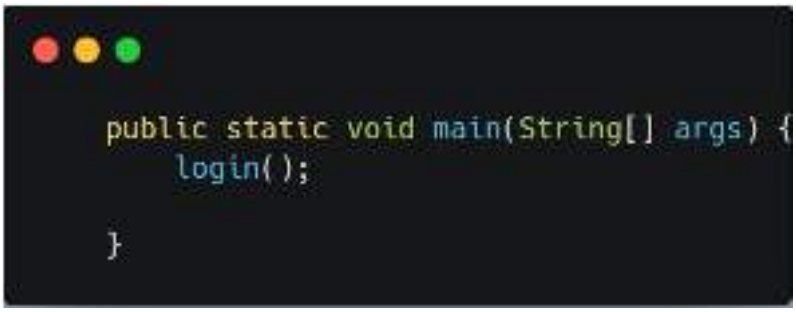
Carga Masiva	Contiene la lógica que gestiona la carga masiva de productos desde archivos y valida que los datos sean correctos.	cargaMasiva
Agregar Producto al Inventario	Contiene la lógica que Valida y agrega productos al inventario de manera controlada.	agregarProductoInventario
Ventas	Contiene la lógica que gestiona el proceso de ventas, validando la cantidad de productos y calculando el total de la venta.	realizarVenta
Realizar Venta	Registra los detalles de la venta realizada en el sistema.	registrarVenta
Inventario	Contiene la lógica relacionada con visualización de los productos disponibles en el inventario.	verInventario
Reportes	Genera reportes como el de productos más vendidos y el historial de ventas.	ReporteTop, ReporteHistorico
Generar Factura	Contiene la lógica que genera una factura con los detalles de los productos vendidos.	generarFactura
Validaciones	Contiene las funciones que validan la existencia de productos, la validez del precio y otros.	ExistenProducto, PrecioValido,

- **Clases principales**

La clase principal contiene métodos estáticos que implementan toda la lógica del sistema

- **Método Main**

  Este método es el encargado de llamar el método lógin.



```
public static void main(String[] args) {
    login();

}
```

- **Método Login**

  El método login solicita las credenciales del usuario en un ciclo. Si las credenciales son correctas, muestra un mensaje de bienvenida y cambia el estado de logged a true. Si son incorrectas, pide los datos nuevamente. Al iniciar sesión correctamente, se llama al método menu.

```
public static void login() {
        System.out.println("\n*******************************************");
        System.out.println("*                LOGIN                    *");
        System.out.println("*******************************************");

        do {

            System.out.println("\nIngrese sus credenciales:");
            System.out.print("Usuario: ");
            inUser = loginSc.nextLine();
            System.out.print("Contraseña: ");
            inPass = loginSc.nextLine();

            if (Objects.equals(inUser, user) && Objects.equals(inPass, password)) {
                System.out.println("\n*******************************************");
                System.out.println("*        ¡BIENVENIDO ADMINISTRADOR!       *");
                System.out.println("*******************************************");
                logged = true;
            } else {
                System.out.println("\n-------------------------------------------");
                System.out.println("*       Usuario o contraseña incorrectos.   *");
                System.out.println("*        Por favor, inténtelo de nuevo.     *");
                System.out.println("-------------------------------------------");
            }
        } while (!logged);

        menu(); // Llama al menú después de iniciar sesión correctamente
    }
```

- **Método menú**

El método menu despliega un menú interactivo para el usuario, donde se valida la opción seleccionada, asegurando que esté dentro del rango permitido. Si la opción es válida, se ejecuta la acción correspondiente utilizando un switch. En caso de que la entrada no sea

válida, se maneja la excepción y se solicita una nueva opción. El ciclo continúa hasta que el usuario selecciona la opción de salida. Al seleccionar "Salir", se cierra el sistema usando un método de terminación controlada.

- **Método agregarProducto**

El método agregarProducto valida que el inventario no esté lleno, que el producto no exista y que el precio sea válido. Si todo es correcto, agrega el producto al inventario. Si hay errores, muestra mensajes de advertencia o captura excepciones.

- **Método cargaMasiva**

El método cargaMasiva permite cargar productos desde un archivo CSV. Abre un cuadro de diálogo para seleccionar el archivo, lee las líneas y procesa cada producto, validando que el precio sea mayor a 0 y que el producto no exista en el inventario. Si los datos son válidos, los agrega al inventario. Si ocurre un error al leer el archivo o procesar los datos, muestra mensajes de error correspondientes.

- **Menú agregarProductoInventario**

El método agregarProductoInventario agrega un producto al inventario. Si el inventario está lleno, muestra un mensaje y no realiza ninguna acción. Si hay espacio, guarda el nombre, precio y cantidad inicial del producto en la matriz ProductsMatrix y aumenta el contador Product_Counter. Finalmente, muestra un mensaje confirmando la adición del producto.

- **Método existeProducto**

  El método existeProducto verifica si un producto ya está en el inventario. Recorre la matriz ProductsMatrix buscando un producto cuyo nombre coincida con el proporcionado (ignorando mayúsculas y minúsculas). Si encuentra una coincidencia, devuelve true; si no, retorna false.

- **Método verInventario**

  El método verInventario muestra todos los productos en el inventario. Recorre la matriz ProductsMatrix e imprime el nombre, precio y cantidad vendida de cada producto, junto con un número de índice para cada uno.

- **Método Realizar venta**

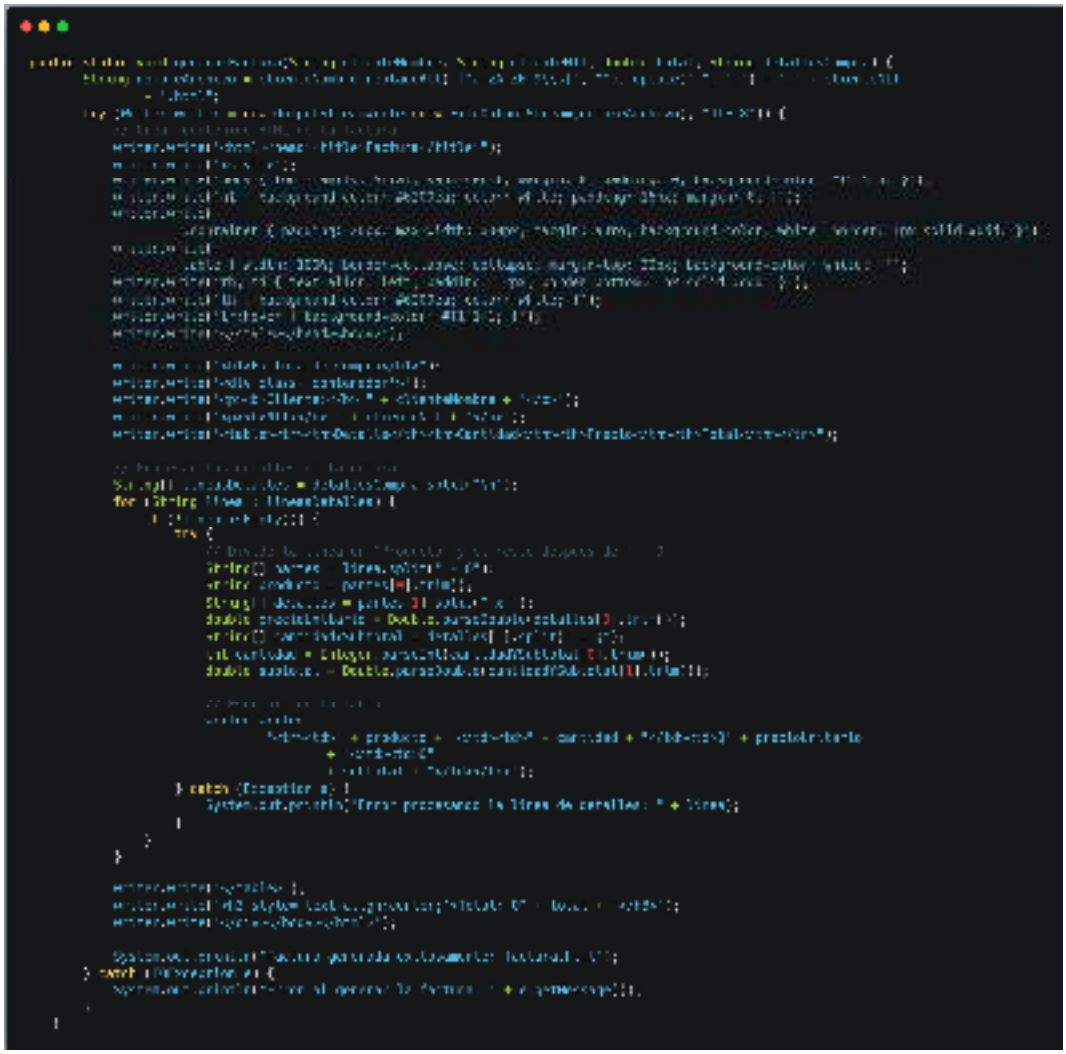
  El método realizarVenta gestiona el proceso de ventas, verificando primero si hay productos disponibles en el inventario. Luego, solicita los datos del cliente y permite seleccionar productos para agregar a la compra, validando la selección y la cantidad. Calcula el total de la venta, actualiza el inventario y registra los detalles de cada producto comprado. Una vez que el usuario finaliza la compra, se registra la venta y se genera la factura correspondiente con los detalles de los productos y el total.

- **Método registrarVenta**

  El método registrarVenta guarda los detalles de una venta en los arreglos correspondientes (clientes, NIT, totales y detalles de ventas), siempre que no se haya alcanzado el límite de ventas (MAX_VENTAS). Si el número de ventas es menor que el límite, registra la venta y aumenta el contador; de lo contrario, muestra un mensaje indicando que no se pueden registrar más ventas.

- **Método generarFactura**

  El método generarFactura crea una factura en formato HTML para una venta. Utiliza el nombre del cliente y el NIT para generar un archivo único. El contenido HTML incluye el nombre y NIT del cliente, los detalles de los productos comprados (producto, cantidad, precio y total), y el total de la compra. Si se encuentra algún error al procesar los detalles de la compra, lo maneja y muestra un mensaje de error. El archivo se guarda con un nombre basado en el cliente y NIT.



- **Método reporteTop**

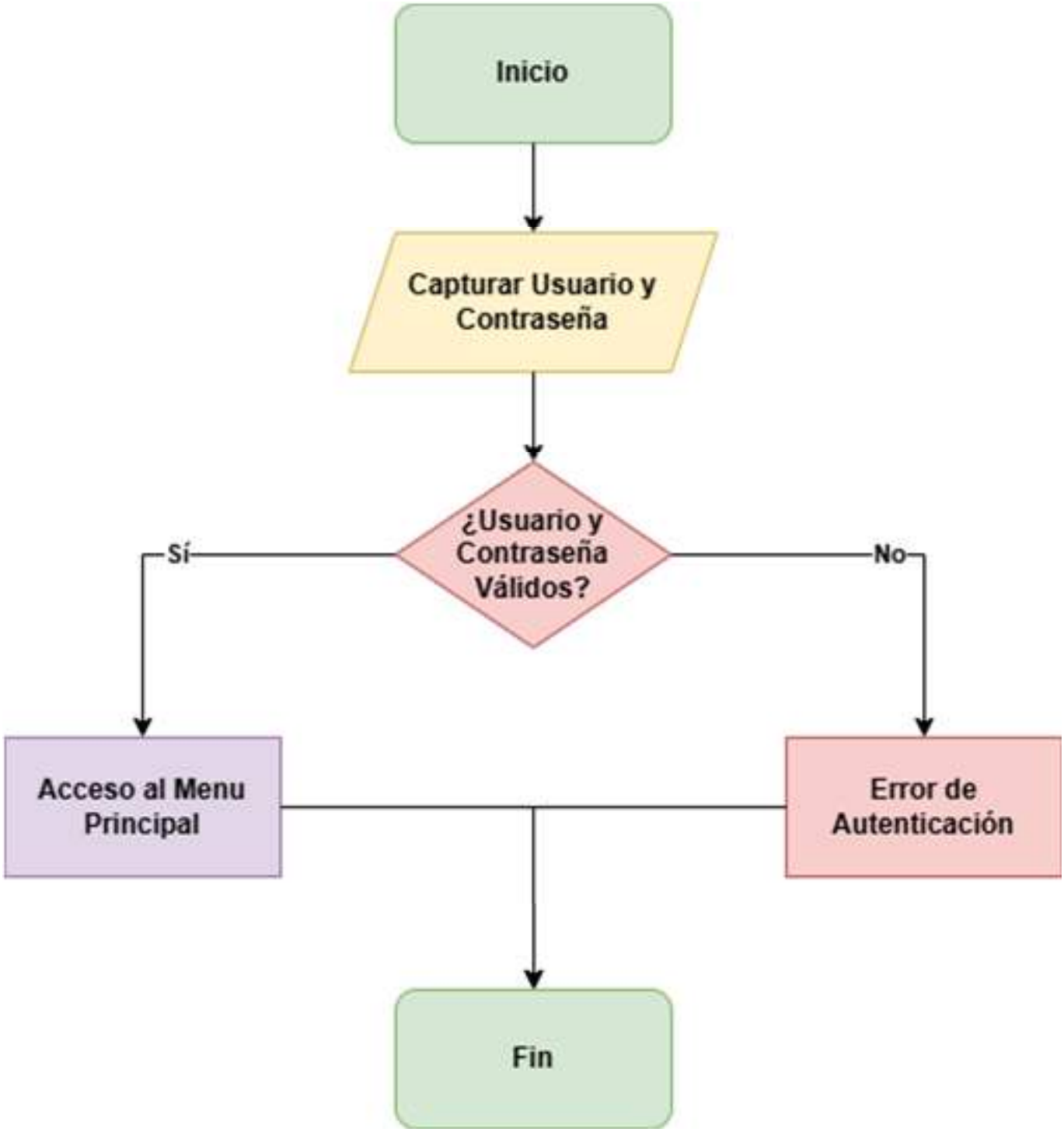
  El método reporteTop() genera un reporte HTML con los 5 productos más vendidos del inventario. Primero verifica si hay productos disponibles y luego recopila sus nombres, precios y cantidades en tres arreglos. A continuación, ordena los productos de acuerdo con la cantidad vendida en orden descendente utilizando un algoritmo de ordenación simple. Después, crea un archivo HTML que muestra los 5 productos más vendidos, su precio y cantidad, mostrando menos si hay menos de 5 productos. Si ocurre un error al generar el archivo, se muestra un mensaje de error, y se asegura de manejar excepciones durante todo el proceso.

- **Método reporteHistorico**

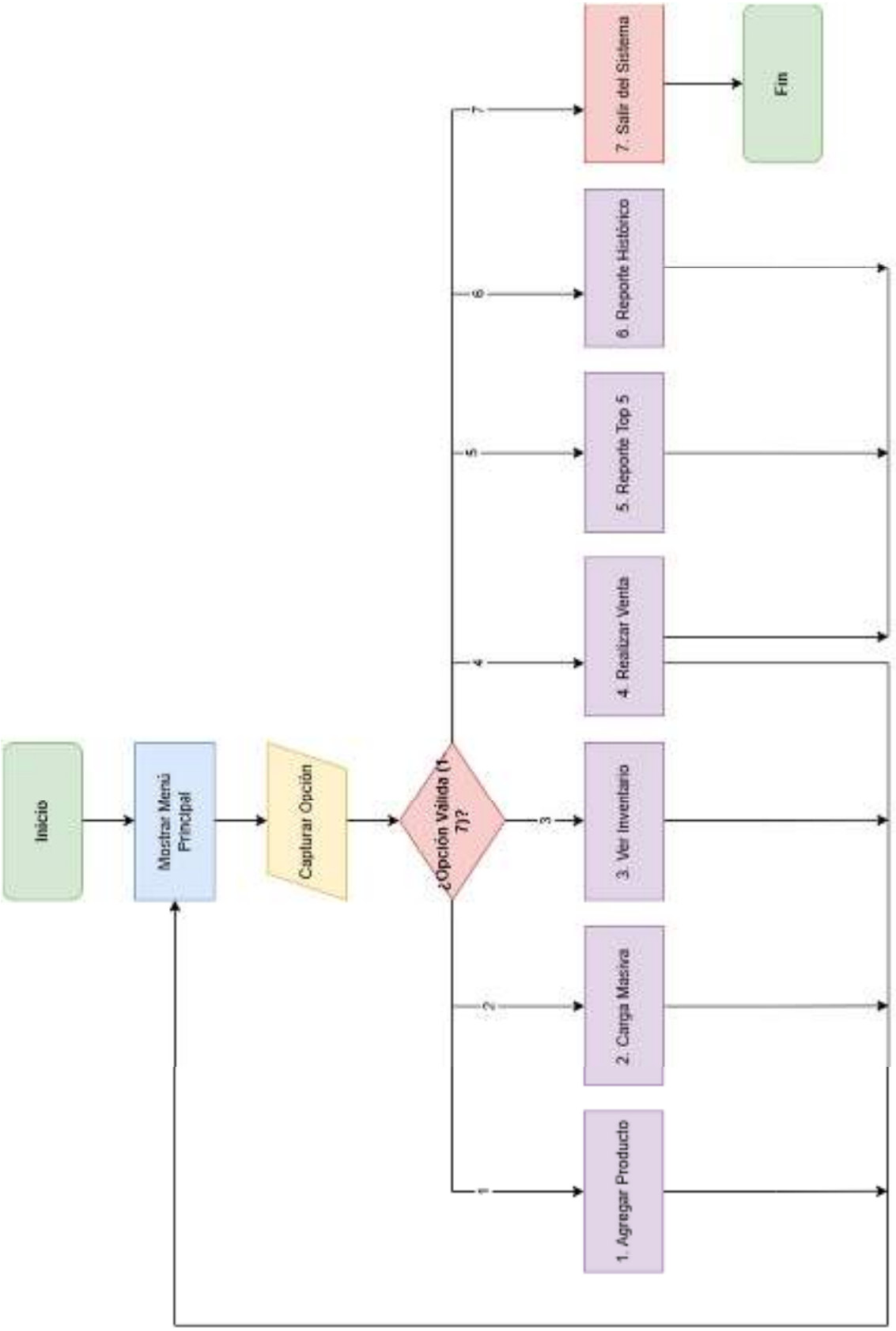
  El método reporteHistorico() genera un archivo HTML que muestra un reporte con el historial de todas las ventas registradas. Primero verifica si hay ventas realizadas; si no, imprime un mensaje indicando que no hay ventas registradas. Luego, crea un archivo HTML donde escribe los detalles de cada venta, incluyendo el nombre del cliente, el NIT, el total de la venta y el detalle de los productos vendidos (con saltos de línea formateados como
). Si ocurre algún error durante la creación del archivo, se maneja mediante una excepción. Finalmente, muestra un mensaje indicando si el reporte fue generado exitosamente o si hubo un error.

VII. DIAGRAMAS DE FLUJO

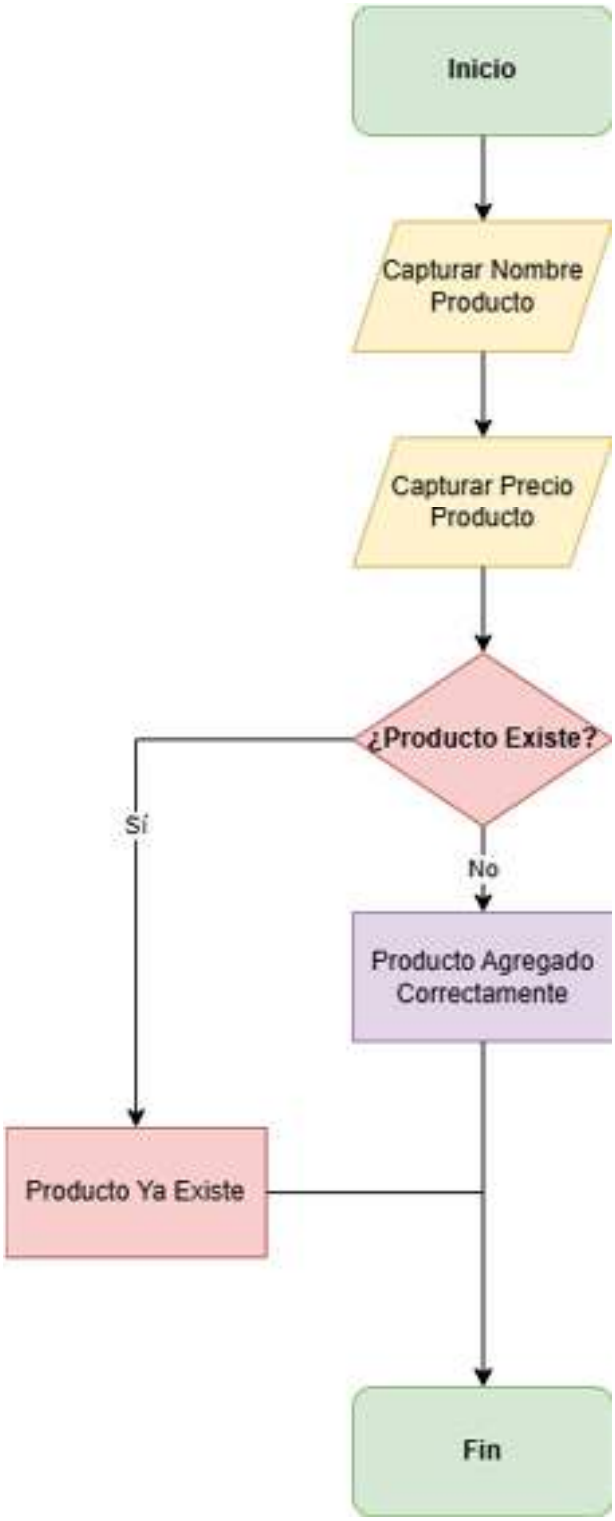
- **Login**



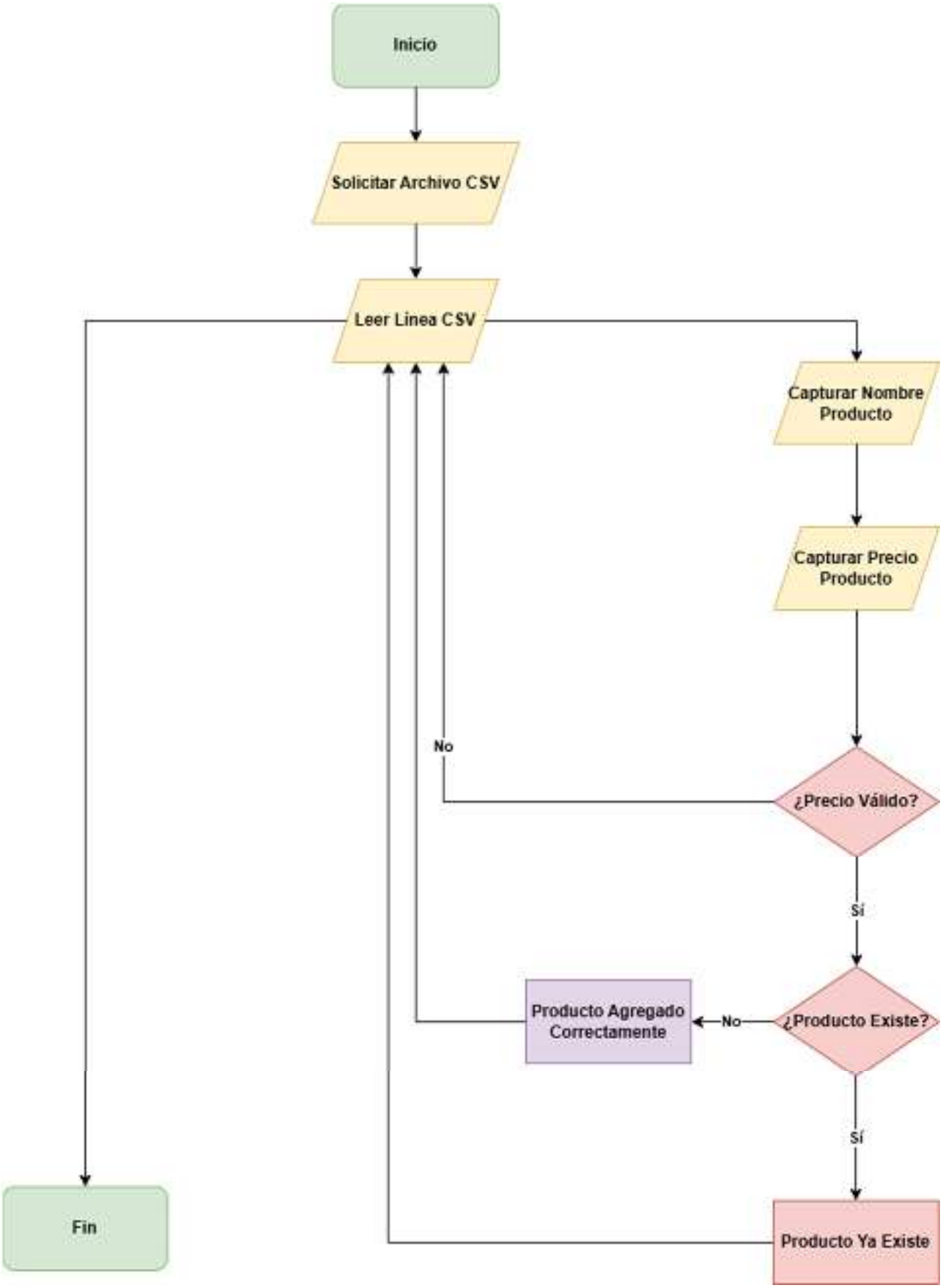
- **Menú principal**



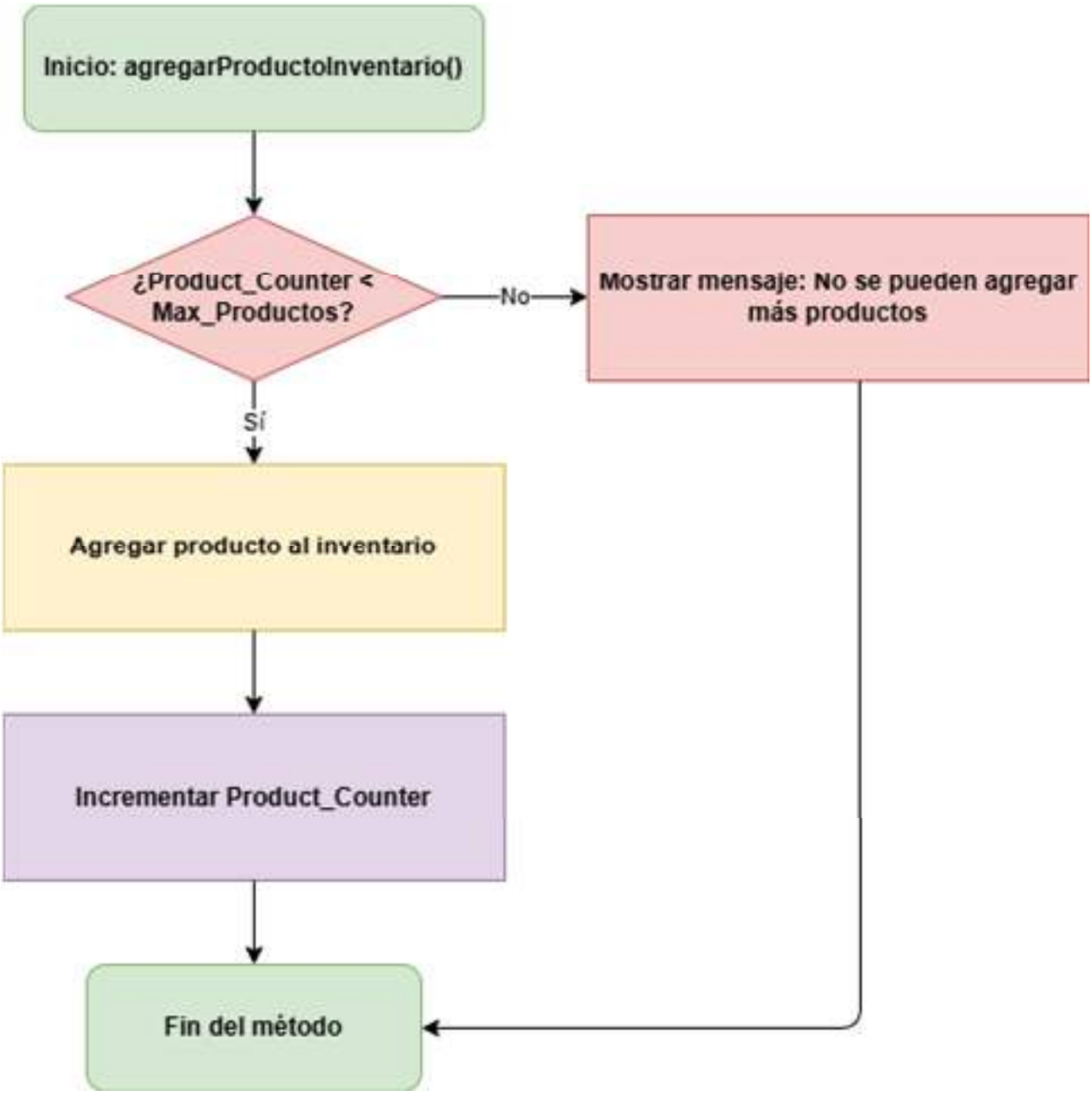
- **Agregar producto**



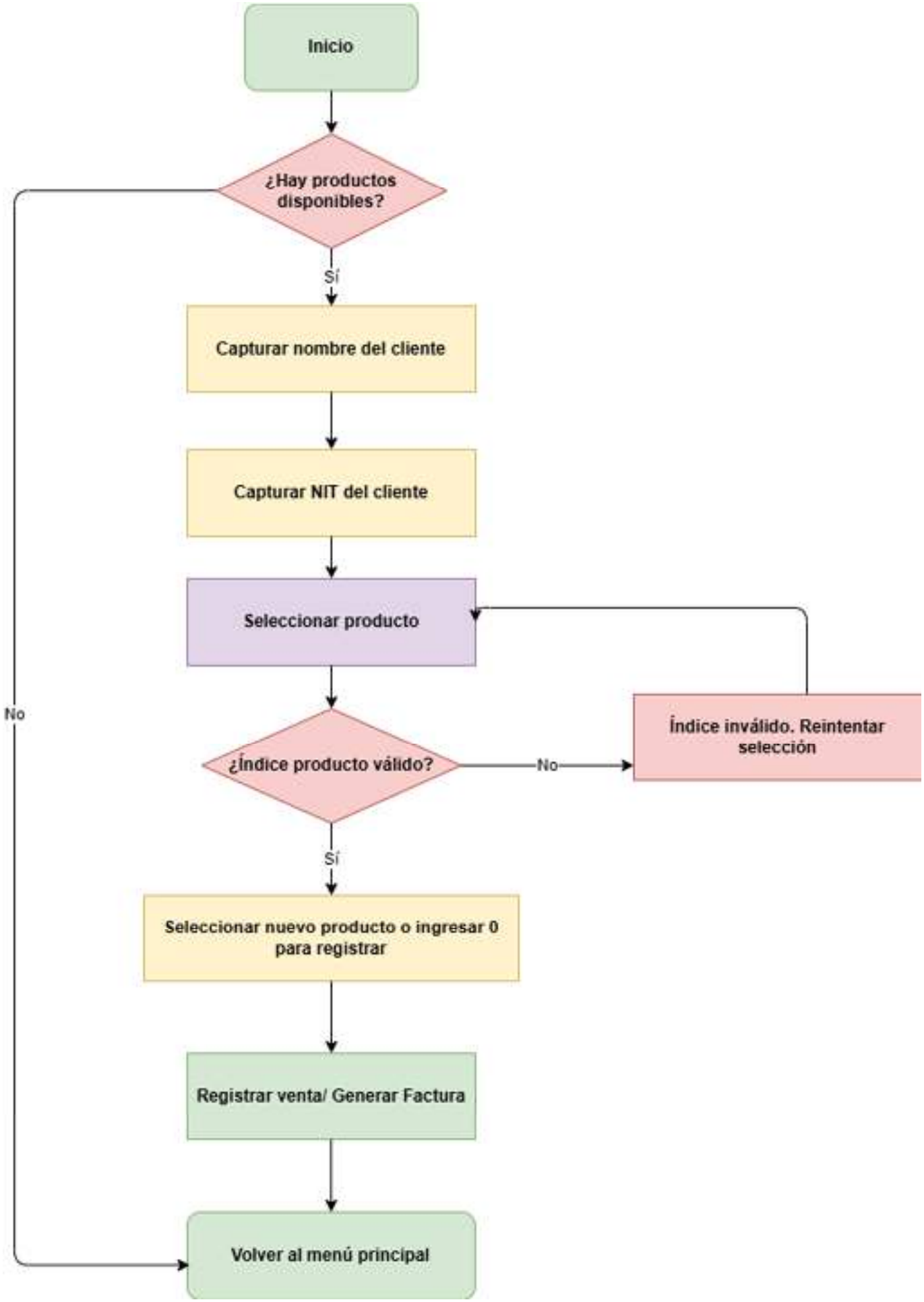
- **Agregar producto (masiva)**



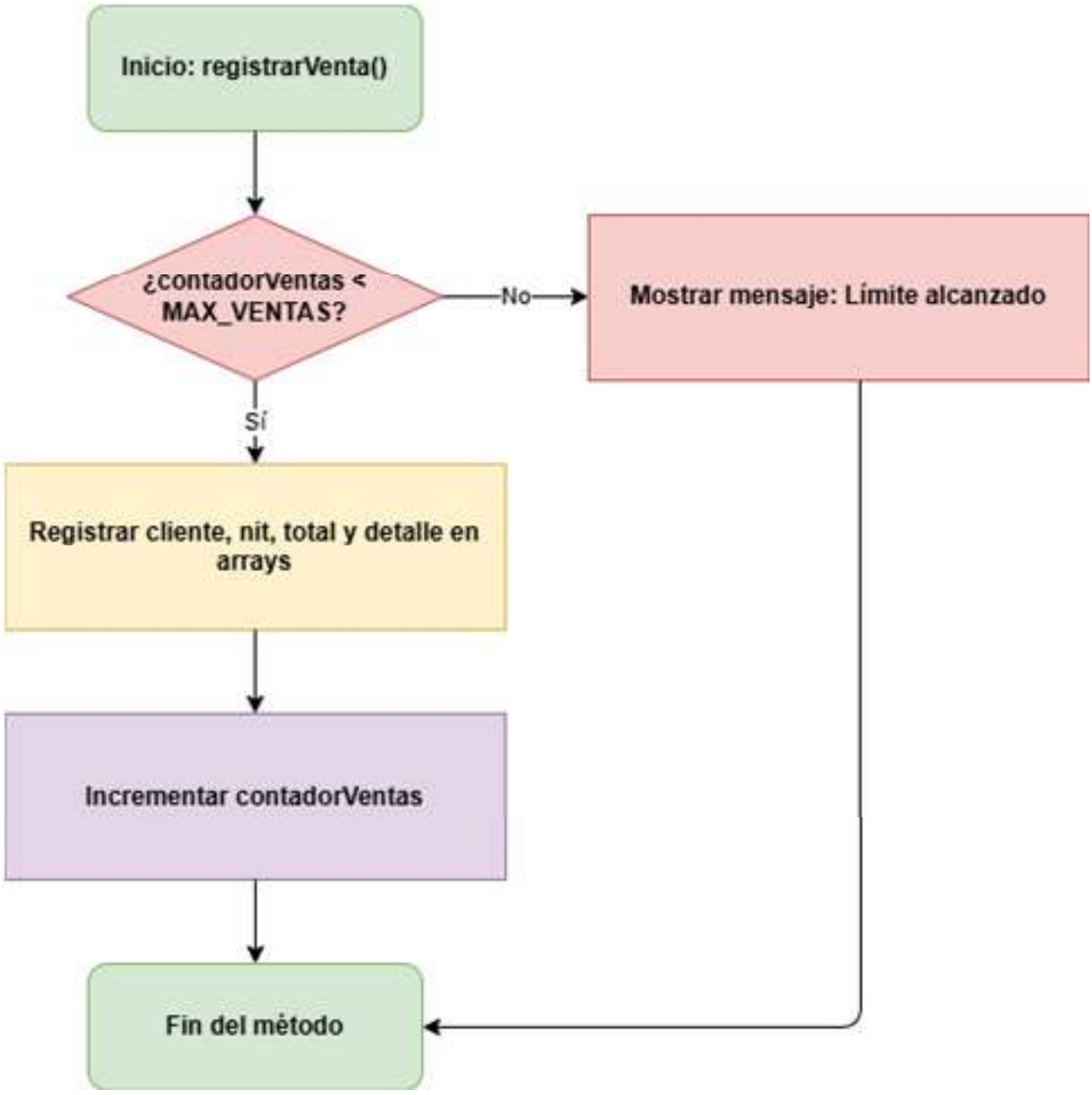
- **Agregar Producto a Inventario**



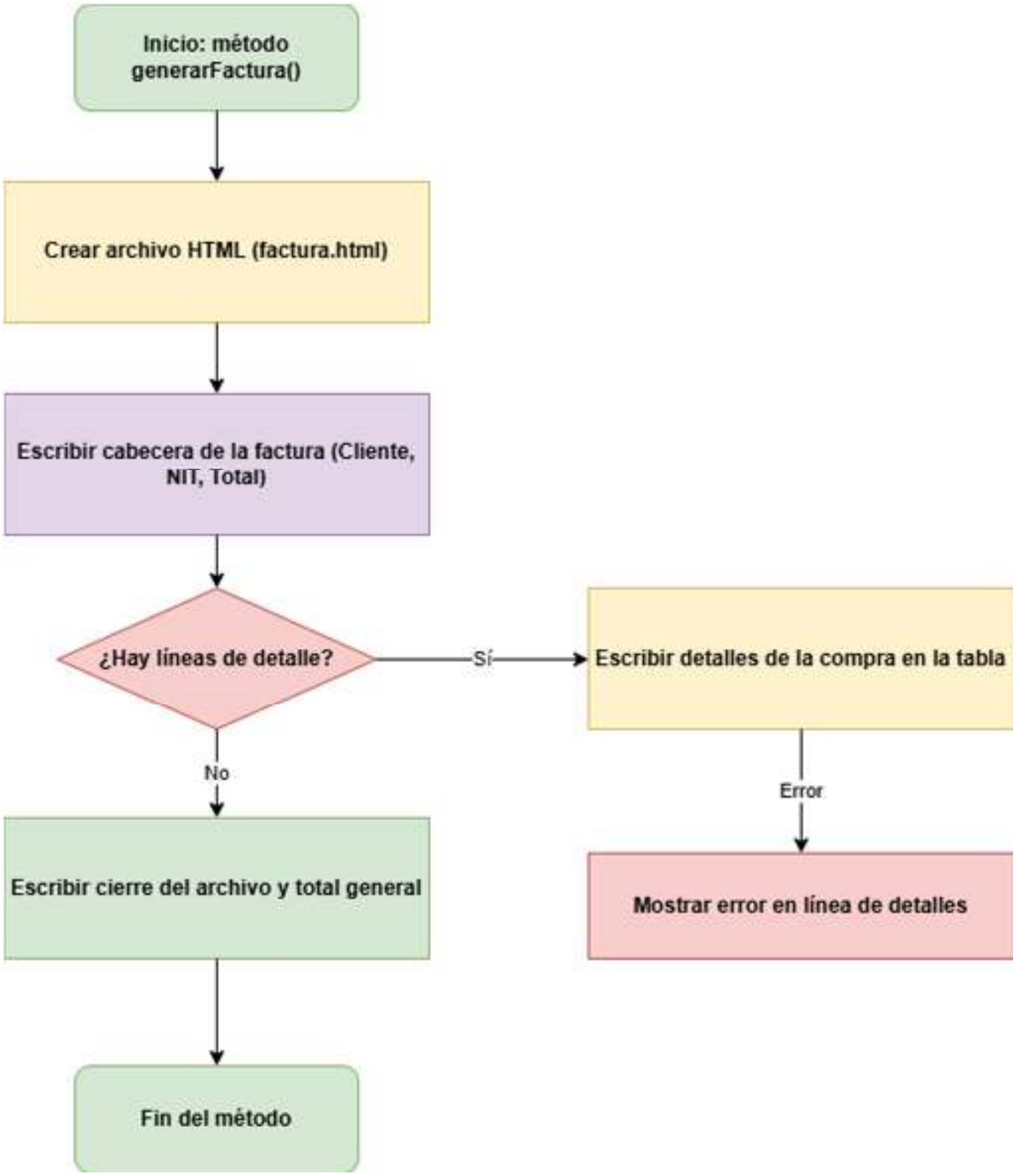
- **Realizar venta**



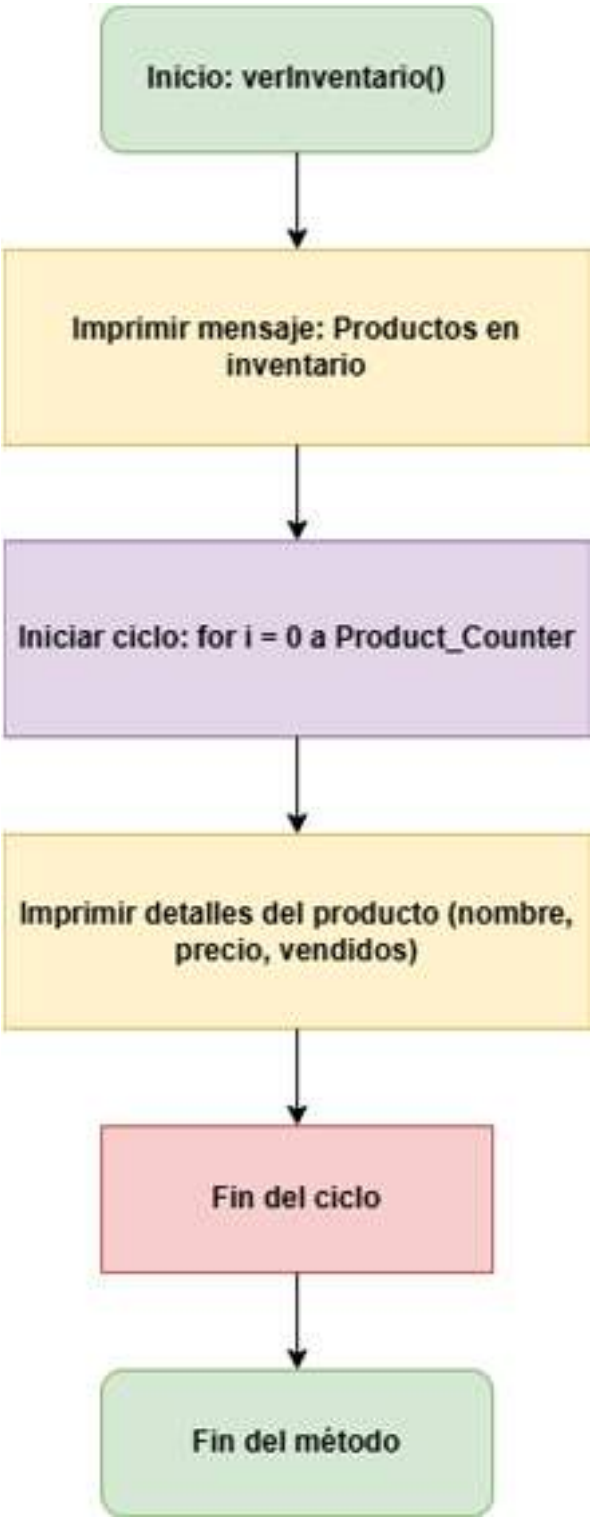
- **Registrar Venta**



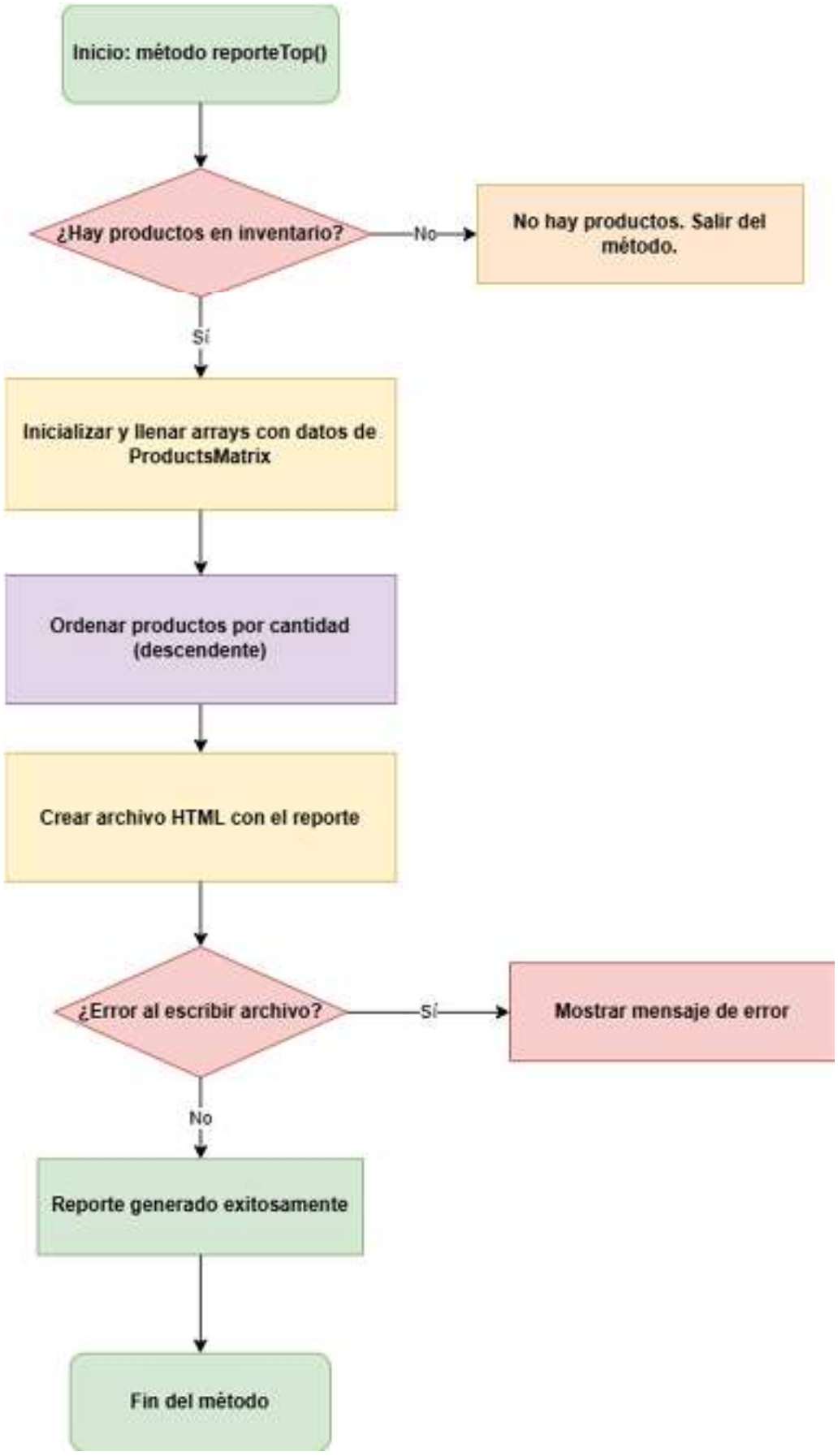
- **Generar Factura**



- **Ver Inventario**



- **Reporte Top 5**



- **Reporte Histórico**