Universidad de San Carlos de Guatemala
Escuela de Ciencias y Sistemas
Facultad de Ingeniería
Introducción a la programación y Computación 1
Segundo Semestre 2025
Catedrático: William Escobar

MANUAL TÉCNICO

Gabriel Eduardo Urbina Sunún
Carnet: 202300384
Cui: 2794761080101

INTRODUCCIÓN

El sistema tiene como finalidad administrar el inventario de una tienda de forma sencilla, permitiendo agregar, buscar, eliminar productos, registrar ventas y generar reportes en PDF.

El proyecto fue diseñado siguiendo una estructura modular en clases (Producto, Venta, BitacoraEntrada, SistemaInventario), de manera que cada componente tiene responsabilidades claras y fáciles de mantener.

Además, la aplicación utiliza la librería externa iText para la generación de reportes en formato PDF, los cuales incluyen la información del inventario y de las ventas realizadas.

Este manual tiene como objetivo proporcionar una guía técnica para comprender la estructura del código, los requerimientos de ejecución y una descripción de los principales métodos implementados.

REQUERIMIENTOS DEL SISTEMA

- Windows 11.
- Netbeans IDE 24.
- Intel core i5 de onceava generación.
- Tarjeta gráfica integrada.
- No tiene necesidad de una conexión a internet.
- Java
- Librería externa: iText 5.x para la generación de reportes en PDF.

EXPLICACIÓN

```java
static int menu() {
    System.out.println("\n----- BIENVENIDO A LA TIENDA DE ROPA -----");
    System.out.println("1. Agregar Producto");
    System.out.println("2. Buscar Producto");
    System.out.println("3. Eliminar Producto");
    System.out.println("4. Registrar Venta");
    System.out.println("5. Generar Reportes");
    System.out.println("6. Ver Datos del Estudiante");
    System.out.println("7. Bitácora");
    System.out.println("8. Salir");
    System.out.print("Seleccione la opción: ");
    int op = leerEntero();
    return op;
}
```

Se muestra el menú principal que enumera las opciones del programa.

```java
static void agregarProducto() {
    System.out.println("\n--- Agregar Producto ---");
    if (numProductos >= MAX_PRODUCTOS) {
        System.out.println("Inventario lleno.");
        registrarBitacora("AGREGAR PRODUCTO", false);
        return;
    }
    System.out.print("Código único: ");
    String codigo = sc.nextLine().trim();
    if (buscarIndiceProductoPorCodigo(codigo) != -1) {
        System.out.println("Error: Código ya existente.");
        registrarBitacora("AGREGAR PRODUCTO", false);
        return;
    }
    System.out.print("Nombre: ");
    String nombre = sc.nextLine().trim();
    System.out.print("Categoria: ");
    String categoria = sc.nextLine().trim();
    System.out.print("Precio (Q): ");
    double precio = leerDoublePositivo();
    System.out.print("Cantidad en stock: ");
    int stock = leerEnteroPositivo();

    inventario[numProductos++] = new Producto(codigo, nombre, categoria, precio, stock);
    guardarInventario();
    System.out.println("Producto agregado correctamente.");
    registrarBitacora("AGREGAR PRODUCTO", true);
}
```

Se validan las opciones para nuestros productos.

EXPLICACIÓN

```java
static void buscarProducto() {
    System.out.println("\n--- Buscar Producto ---");
    System.out.println("Buscar por: 1. Código  2. Nombre  3. Categoría");
    System.out.print("Seleccione: ");
    int tipo = leerEntero();
    boolean encontrado = false;
    switch (tipo) {
        case 1:
            System.out.print("Código: ");
            String codigo = sc.nextLine().trim();
            int idx = buscarIndiceProductoPorCodigo(codigo);
            if (idx != -1) {
                System.out.println("Encontrado: " + inventario[idx]);
                encontrado = true;
            }
            break;
        case 2:
            System.out.print("Nombre del producto: ");
            String nombre = sc.nextLine().trim().toLowerCase();
            for (int i = 0; i < numProductos; i++) {
                if (inventario[i].nombre.toLowerCase().contains(nombre)) {
                    System.out.println(inventario[i]);
                    encontrado = true;
                }
            }
            break;
        case 3:
            System.out.print("Categoría: ");
            String categoria = sc.nextLine().trim().toLowerCase();
            for (int i = 0; i < numProductos; i++) {
                if (inventario[i].categoria.toLowerCase().equals(categoria)) {
                    System.out.println(inventario[i]);
                    encontrado = true;
                }
            }
            break;
        default:
            System.out.println("Opción de búsqueda inválida.");
    }
    if (!encontrado) System.out.println("No se encontraron coincidencias.");
    registrarBitacora("BUSCAR", encontrado);
}
```

El método buscarProducto nos permite encontrar los productos ingresados posteriormente.

EXPLICACIÓN

```java
static void eliminarProducto() {
    System.out.println("\n--- Eliminar Producto ---");
    System.out.print("Código: ");
    String codigo = sc.nextLine().trim();
    int idx = buscarIndiceProductoPorCodigo(codigo);
    if (idx == -1) {
        System.out.println("No existe ese código.");
        registrarBitacora("ELIMINAR", false);
        return;
    }
    System.out.println("¿Confirmar eliminar? (s/n): ");
    String conf = sc.nextLine().trim().toLowerCase();
    if (!conf.equals("s")) {
        System.out.println("Operación cancelada.");
        registrarBitacora("ELIMINAR", false);
        return;
    }
    for (int i = idx; i < numProductos - 1; i++) {
        inventario[i] = inventario[i + 1];
    }
    inventario[--numProductos] = null;
    guardarInventario();
    System.out.println("Producto eliminado.");
    registrarBitacora("ELIMINAR", true);
}
```

El método eliminarProducto nos permite eliminar los productos ingresados posteriormente.

El método registrarVenta nos permite verifica que existan los productos y permite que se produzca una venta de dicho producto, almacenando datos.

```java
static void registrarVenta() {
    System.out.println("\n--- Registrar Venta ---");
    if (numVentas >= MAX_VENTAS) {
        System.out.println("Registro de ventas lleno.");
        registrarBitacora("VENTA", false);
        return;
    }
    System.out.print("Código del producto: ");
    String codigo = sc.nextLine().trim();
    int idx = buscarIndiceProductoPorCodigo(codigo);
    if (idx == -1) {
        System.out.println("Producto no existe.");
        registrarBitacora("VENTA", false);
        return;
    }
    System.out.print("Cantidad vendida: ");
    int cant = leerEnteroPositivo();
    if (cant > inventario[idx].stock) {
        System.out.println("Stock insuficiente. Disponible: " + inventario[idx].stock);
        registrarBitacora("VENTA", false);
        return;
    }
    inventario[idx].stock -= cant;
    double total = inventario[idx].precio * cant;
    String fechaHora = fechaHora();
    Venta v = new Venta(codigo, cant, fechaHora, total);
    ventas[numVentas++] = v;
    guardarInventario();
    appendVentaArchivo(v);
    System.out.printf("Venta registrada. Total: Q%.2f\n", total);
    registrarBitacora("VENTA", true);
}
```

EXPLICACIÓN

```java
static boolean generarReporteStockTXT(String ruta) {
    try (BufferedWriter bw = new BufferedWriter(new FileWriter(ruta))) {
        bw.write("CODIGO | NOMBRE | CATEGORIA | PRECIO | STOCK");
        bw.newLine();
        for (int i = 0; i < numProductos; i++) {
            Producto p = inventario[i];
            bw.write(String.format("%s | %s | %s | Q%.2f | %d", p.codigo, p.nombre, p.categoria, p.precio, p.stock));
            bw.newLine();
        }
    return true;
    } catch (IOException e) {
        System.out.println("Error al generar reporte de stock: " + e.getMessage());
        return false;
    }
}

static boolean generarReporteVentasTXT(String ruta) {
    try (BufferedWriter bw = new BufferedWriter(new FileWriter(ruta))) {
        bw.write("CODIGO | CANTIDAD | FECHA_HORA | TOTAL");
        bw.newLine();
        double suma = 0.0;
        for (int i = 0; i < numVentas; i++) {
            Venta v = ventas[i];
            bw.write(String.format("%s | %d | %s | Q%.2f", v.codigoProducto, v.cantidad, v.fechaHora, v.total));
            bw.newLine();
            suma += v.total;
        }
        bw.write(String.format("TOTAL ACUMULADO: Q%.2f", suma));
        bw.newLine();
        return true;
    } catch (IOException e) {
        System.out.println("Error al generar reporte de ventas: " + e.getMessage());
        return false;
    }
}
```

Los métodos nos permiten generar reportes ya sean de ventas o reportes de productos, utilizando ciclos recorre los vectores y los imprime.

```java
static void verDatosEstudiante() {
    System.out.println("\n--- Datos del Estudiante ---");
    System.out.println("Nombre : " + ESTUDIANTE_NOMBRE);
    System.out.println("Carnet : " + ESTUDIANTE_CARNET);
    registrarBitacora("DATOS", true);
}

static void verBitacora() {
    System.out.println("\n--- Bitácora (temporal) ---");
    for (int i = 0; i < numBitacora; i++) {
        System.out.println(bitacora[i]);
    }
    guardarBitacoraArchivo();
    registrarBitacora("BITACORA", true);
}

static void salir() {
    guardarInventario();
    guardarBitacoraArchivo();
    registrarBitacora("SALIR", true);
    System.out.println("Saliendo... ¡Hasta luego!");
}
```

En el primer método nos muestra los datos del estudiante.

El segundo método nos muestra los cambios realizados ya sea modificar productos, ingresar productos o generar reportes.

CONCLUSIÓN

El Sistema de Inventario en Java desarrollado cumple con los objetivos planteados en el proyecto, ofreciendo una solución práctica y sencilla para la gestión de productos en una tienda.

A través de la implementación de funciones básicas como agregar, buscar, eliminar productos, registrar ventas y generar reportes en PDF, se logró un programa que simula de manera efectiva un sistema de control de inventario real.

El uso de vectores como estructura de almacenamiento garantiza el cumplimiento de los requerimientos académicos, mientras que la incorporación de la librería iText permitió ampliar las capacidades del sistema mediante la generación de reportes en formato PDF, ofreciendo al usuario una salida más profesional y fácil de compartir.

En conclusión, el proyecto no solo permitió poner en práctica los conceptos de programación en Java, manejo de archivos y modularidad, sino que también fortaleció las habilidades de diseño y organización del código, preparando la base para el desarrollo de aplicaciones más complejas en el futuro.