Universidad de San Carlo de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas
Introducción a la programación y computación 1
Ing. Moisés Velásquez
Auxiliar. Pablo Oliva

PROYECTO No.1: INFORME DE DESARROLLO

Guillermo Enrique Marroquin Morán

202103527

Guatemala 12 de sep. de 25

INDICE

Informe de desarrollo del proyecto

¿Cómo desarrolle la práctica?

Para la realización de este proyecto se llevó a cabo con el lenguaje de java, con el IDE de NetBeans, esta misma tuvo un registro en GitHub por medio de dos commit´s semanales en su debido repositorio.

Para iniciar con lo que realmente importa, leí el enunciado repetidamente para su claridad, luego anoto los puntos clave, con todas las validaciones que se están pidiendo, como el valor de stock sea un valor entero, que sea positivo y que no sea una letra, etc.

Empecé a realizar el menú principal, con mucha base de la práctica dado que, en sí, el menú como tal es casi el mismo, claro con sus diferencias, como que ahora son menos opciones, y ahora hay que haber una validación para cuando se añadan valores negativos o de por si valores no válidos, usando el ciclo do while para el menú, y el “switch case” para cada opción y opción no valida,

```
do {
    System.out .println("Bienvenido al menu principal");
    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. Bitacora de acciones");
    System.out .println("8. Salir");
    System.out .println("Elije una opción:");
```

De igual manera que en la práctica, en este sistema de inventario no se podía usar arraylist, entonces esta vez hice una matriz variable de 100*5, para registrar hasta 100 productos, cosa que dudo que alguien tenga la paciencia de eso, pero oye ya es algo.

```
public class Proyecto1 {

    public static void main (String[] args) {
        Scanner sc = new Scanner(System.in );
        int opcion = 0;
        //Declaracion de la matriz del oinventario y continuar
        final int MAXPROD = 100;
        final int ATRI = 5;
        String[][] Inventario = new String[MAXPROD][ATRI];
        int CantInventario = 0;
        String continuar;
        String vendedor = "Guillermo Marroquin";
```

Para las validaciones del programa, hice una función con “*try catch”,* para que en el menú verifique si hay algún número positivo, o si en dado caso hay algo que NO sea un número, con el “!sc.hasNextInt” verificamos que el numero sea un entero valido, y el “InputMismatchException” hace que si recibe un valor que no corresponde, imprima en pantalla el Error 002(SIP no tuve mucha creatividad para eso), en la mayoría del proyecto lo dividí, para que en cada clase hayan funciones/métodos que “tengan sentido”, en toda verificación salga bien o mal está el método/función nombrado como “VerAccion”, SIP ya lo se ser creativo no es lo mío.

```
try {
    opcion = Validar.VerificarNum (sc);
    sc.nextLine();
    // Consumir el salto de linea restante
} catch (InputMismatchException e) {
    System.out .println(e.getMessage());
    System.out .println("Error 001: Debes ingresar un valor numérico. Inténtalo de nuevo
    sc.nextLine();
    VerAccion (vendedor, "Ingreso de opcion al menu", "fallida");
    break;
}
```

```
public static int VerificarNum (Scanner sc) throws InputMismatchException {
    if (!sc.hasNextInt()) {
        throw new InputMismatchException("Error 002: Por favor ingresa un numero del 1 al 8");
    }
    return sc.nextInt();
}
//Un nuevo metodo para ver si la cantidad de estock y lo otro sea positivos, o en general numeros vaya
public static int verNumPos (Scanner sc) {
    int num;
    while (true) {
        try {
            num = sc.nextInt();
            if (num < 0) {
                System.out .println("Error 003: por favor ingrese un valor positivo");
                System.out .println("Por favor elija un numero postivo");
                System.out .println();
            } else {
                sc.nextLine();
                return num;
            }
        } catch (InputMismatchException e) {
            System.out .println("Error 004: por favor ingrese un valor numerico que sea valido");
            sc.nextLine();
        }
    }
}
```

Siguiendo con esto, desarrollé las opciones facilitas, mi nombre y salir, la diferencia es que acá use el “/n” para ahorrar en impresiones, también podría hacerlo en el menú, pero me perdería si lo hiciera,

```
            case 6:
                System.out .println("Ver Datos del Estudiante"); // esto de los saltos de linea me sirve para ahorrar espacio
                System.out .println("Este programa fue realizado por Guillermo Enrique Marroquin Morán. \nCarnet: 202103527. \nEste programa si tiene Derechos de autor");
                System.out .println();
                VerAccion (vendedor, "Ver datos del estudiante", "Correcta");// como chuccha podria NO ver mis datos....
                break;
```

```
                case 8:
                    System.out .println("¡Feliz tarde, adiós!");
                    VerAccion (vendedor, "Salir del programa", "Correcta");
                    break;
                default:
                    System.out .println("Opción no válida, ingresa otra opción");
                    System.out .println();
                    VerAccion (vendedor, "Ingreso de opcion del menu ", "Fallida");
            }
        } while (opcion != 8);
        sc.close();
    }
}
```

Siguiendo con el orden de desarrollo del proyecto, empecé con la opción 1, la de agregar productos, al igual que en la práctica hay que agregar que si en dado caso hay algo repetido nos indique, y que debamos ingresar un código diferente, en esta parte ya tenía un problema de un bucle infinito de “ingrese el código del producto”, ya en la explicación de problemas hare hincapié en eso. Regresando al desarrollo, al ingresar un producto este mismo se guarda en la matriz llamada “Inventario”, como dato para agregar la categoría del producto hice una clase llamada productos donde están todos los métodos/funciones que tengan relación a estos mismos a excepción de la validación de la categoría de los productos, aquí si no sé porque lo puse acá, pero allí se queda.

```
switch (opcion) {
  case 1:
    do {
      if (CantInventario >= MAXPROD) {
        System.out.println("Haz llegado al limite de productos ingresados");
        break;
      }
      sc.nextLine();
      System.out.println("Bienvenido al apartado de agregar productos");
      System.out.println();

      String codigo;
      boolean CodRep;
      do {
        CodRep = false;  //Validamos que el codigo sea unico
        System.out.println("Ingrese el codigo del producto:");
        codigo = sc.nextLine();
        for (int i = 0; i < CantInventario; i++) {
          if (Inventario[i][0] != null && Inventario[i][0].equalsIgnoreCase(codigo)) {
            CodRep = true;
            System.out.println("Ese codigo ya fue ingresado, favor de ingresar otro");
            break;
          }
        }
      } while (CodRep);

      System.out.print("Nombre del Producto:");
      String nombre = sc.nextLine();

      //yay hice un llamado a un metodo wuuuuu
      String categoria = Validar.validarCategoria(sc);

      System.out.println("Precio del producto(Q):");
      int precio = Validar.verNumPos(sc);

      System.out.println("Cantidad en Stock:");
      int stock = Validar.verNumPos(sc);
```

```
      Inventario[CantInventario][0] = codigo;
      Inventario[CantInventario][1] = nombre;
      Inventario[CantInventario][2] = categoria;
      Inventario[CantInventario][3] = String.valueOf(precio);
      Inventario[CantInventario][4] = String.valueOf(stock);
      CantInventario++;

      System.out.print("¿Desea ingresar otro producto? (s/n):  ");
      continuar = sc.nextLine().toLowerCase();
      System.out.println();
    } while (continuar.equals("s"));
    VerAccion(vendedor, "Agregar producto", "Correcta");
    break;
```

```
public static String validarCategoria(Scanner sc) {
  int optCatego;
  do {
    System.out.println("Selecione la categoria a la que pertenezca el producto:");
    System.out.println("1. Camisa");
    System.out.println("2. Pantalón");
    System.out.println("3. Accesorio");
    System.out.println("Ingrese la opcion:");
    try {
      optCatego = sc.nextInt();
      sc.nextLine();
      switch (optCatego) {
        case 1:
          return "Camisa";
        case 2:
          return "Pantalón";
        case 3:
          return "Accesorio";
        default:
          System.out.println("Opcion no valida, por favor, elige un numero del a al 3");
          break;
      }
    } catch (InputMismatchException e) {
      System.out.println("Error 005: Debes de ingresar un numero, intentelo de nuevo");
      sc.nextLine();
    }
  } while (true);
 }
}
```

Siguiendo con este proceso llamado desarrollo, la opción que sigue es la de buscar productos, para eso, al igual que en la práctica hice dos métodos, uno para buscar en la matriz de inventario por: su nombre, su categoría y su código, y el otro para mostrar los productos encontrados, para estos me base mucho en la práctica, solo con la diferencia, que debía buscarse por opciones, y al igual que en la práctica, que recorra con un ciclo for la matriz de inventario por las filas correspondientes, 0 para el código, 1 para el nombre y 2 para la categoría, en este ultima, hay que escribir el nombre de la categoría(en pantalón hay que poner la tilde)

```
public static void buscarProducto (String[][] Inventario, int CantInventario, Scanner sc, String vendedor) {
    int optBuscar;
    String valBuscar;
    boolean encontrar = false;
    System.out.println("Bienvenido a la seccion de busqueda de Productos ");
    System.out.println();//sip, un salto de linea pa que se vea mejor
    System.out.println("¿Como desea buscar el producto'");
    System.out.println("1. Por Codigo");
    System.out.println("2. Por Nombre");
    System.out.println("3. Por Categoria");
    System.out.println("Ingrese su opcion:");
    try {
        optBuscar = sc.nextInt();
        sc.nextLine();
    } catch (InputMismatchException e) {
        System.out.println("Error 006: Debe de ingresar un numero valido, intente de nuevo");
        sc.nextLine();
        VerAccion(vendedor, "Busqueda de productos, opcion invalida ", "fallida");
        return;
    }
    switch (optBuscar) {
        case 1:
            System.out.println("Ingrese el codigo a buscar:");
            valBuscar = sc.nextLine();
            for (int i = 0; i < CantInventario; i++) {
                if (Inventario[i][0] != null && Inventario[i][0].equalsIgnoreCase(valBuscar)) {
                    mostrarProd(Inventario, i);
                    encontrar = true;
                    break;
                }
            }
            break;
```

```
        case 2:
            System.out.println("Ingrese el nombre a buscar:");
            valBuscar = sc.nextLine();
            for (int i = 0; i < CantInventario; i++) {
                //esto va a ignorar las mayusculas y minusculas, solo toma las palabras
                if (Inventario[i][1] != null && Inventario[i][1].toLowerCase().contains(valBuscar.toLowerCase())) {
                    mostrarProd(Inventario, i);
                    encontrar = true;
                }
            }
            break;
        case 3:
            System.out.println("Ingrese la categoria a buscar:");
            valBuscar = sc.nextLine();
            for (int i = 0; i < CantInventario; i++) {
                //Igual aca, solo que.. bueno hace una equivalencia entre los productos ingresado y los que encuentre, solo que ignora las mayusculas
                if (Inventario[i][2] != null && Inventario[i][2].toLowerCase().equals(valBuscar.toLowerCase())) {
                    mostrarProd(Inventario, i);
                    encontrar = true;
                }
            }
            break;
        default:
            System.out.println("Opcion no valida, intente otra vez");
            VerAccion(vendedor, "busqueeda de producto, opcion invalida ", "fallida");
            return;
    }
    if (!encontrar) {
        System.out.println("No se encontraron productos que coincidan con la busqueda");
        VerAccion(vendedor, "Busqueda de producto, no se encontró el producto ", "fallida");
    } else {
        VerAccion(vendedor, "Busqueda de producto", "Correcta");
    }
}
```

```
//Este metodo ayuda a visualizar los porductos, igual me base de la practica... sip mi creatividad llego a 0
public static void mostrarProd (String[][] Inventario, int i) {
    System.out.println("Se ha encontrado el producto");
    System.out.println("Codigo: " + Inventario[i][0]);
    System.out.println("Nombre: " + Inventario[i][1]);
    System.out.println("Categoria: " + Inventario[i][2]);
    System.out.println("Precio: " + Inventario[i][3]);
    System.out.println("Stock: " + Inventario[i][4]);
}
```

Regresando al ruedo, vamos con la tercera opción de igual manera usamos nuestra clase productos y cree un método para eliminar productos que lo nombre como: “EliminarProd” en este pedimos el código del producto y con el ciclo for en la matriz de inventario buscamos el código, ignorando mayúsculas de minúsculas, si se encuentra, imprime en pantalla el producto a eliminar, usando la función de mostrarProd, ya al poner “s” o “n” el programa borrara o no el producto de la matriz inventario, luego de eso limpia la última fila para evitar duplicaciones por posición, cosa que pasaba en la práctica, y mucho.

```java
//Sip un Metodo que sea para buscar productos exclusivamante para eliminarlos, me base mucho en la de la practica yeeey
public static int EliminarProd (String[][] Inventario, int CantInventario, Scanner sc, String vendedor) {
    System.out .println("Ingrese el Codigo del producto que se quiera eliminar:  ");
    String codEli = sc.nextLine();
    int Eliminar = -1;
    //Usamos un bucle for para buscar en la matriz inventario para encontrar el producto
    for (int i = 0; i < CantInventario; i++) {
        if (Inventario[i][0] != null && Inventario[i][0].equalsIgnoreCase(codEli)) {
            Eliminar = i;
            break;
        }
    }
//Si el producto fue encontrado
    if (Eliminar != -1) {
        System.out .println("Se encontro el siguiente producto: ");
        //Hacemos un llamado para mostrar el producto a eliminar
        mostrarProd (Inventario, Eliminar);
        System.out .println();
        System.out .println("¿Esta seguro de la eliminacion del producto (s/n): ");
        String Confirmar = sc.nextLine().toLowerCase();

        if (Confirmar.equals("s")) {
            //Eliminamos la posicion donde se encuentre el producto
            for (int i = Eliminar; i < CantInventario - 1; i++) {
                Inventario[i] = Inventario[i + 1];
            }
//Este bloquecito de codigo bonito es para "limpiar" la ultima fila para que no haya duplicaciones
            Inventario[CantInventario - 1] = new String[5];
            System.out .println("El producto fue eliminado ");
            VerAccion (vendedor, "Eliminacion del producto con el codigo:  " + Eliminar, "Correcta");
            CantInventario--;//baja el "contenido" del inventario una unidad UWU
        } else {
            System.out .println("El producto no fue eliminado ");
            VerAccion (vendedor, "Eliminacion del producto con el codigo:  " + Eliminar, "fallida");
        }
    } else {
        System.out .println("El producto: " + codEli + "no fue encontrado");
```

Regresando al menú principal seguimos con la opción de registrar ventas, en este al igual que en buscar producto, con un ciclo for se busca en la matriz de inventario, por el código ignorando mayúsculas y minúsculas, si se encuentra, procede hacer una validación de si en dado caso la cantidad vendida es mayor, al stock actual no permita hacer la venta, y se reingrese la cantidad, siguiendo con el código, se hace la venta, con los datos del articulo vendido junto a la fecha y hora en la que se hizo, en un archivo de texto, y si en dado caso este archivo no se encuentra no se puede registrar nada.

```java
//Una funcion para registrar una venta, usando como base la parte de la parctica de buscar por nombre
public static void RegisVenta (String[][] Inventario, int CantInventario, Scanner sc, String vendedor) {
    int Prod = -1;
    System.out .println("Ingrese el codigo del producto a vender: ");
    String CodVenta = sc.nextLine();
    for (int i = 0; i < CantInventario; i++) { // Buscamos en el inventario por el codigo con un ciclo for
        if (Inventario[i][0] != null && Inventario[i][0].equalsIgnoreCase(CodVenta)) {
            Prod = i;
            break;
        }
    }
    if (Prod != -1) {// Esto es porque si  en dado caso el indice a buscar es -1, por la posicion -1 de la matriz esto no va afuncionar
        mostrarProd (Inventario, Prod);

        System.out .println("Ingrese la cantidad del produco que se va a vender");
        int cantidad = Validar.verNumPos (sc);
        int Stock2 = Integer.parseInt (Inventario[Prod][4]);

        //Validadmos que la venta sea exitosa
        if (cantidad > Stock2) {
            System.out .println("Error 006: La cantidad a vender es superior al stock");
            VerAccion (vendedor, "Venta, Stock INSUFICIENTE ", "fallida");
            return;
        }
        // Actualizamos el nuevo stock con la cantidad vendida
        int Stock3 = Stock2 - cantidad;
        Inventario[Prod][4] = String.valueOf (Stock3);

        //calculamos el total de la venta :3
        double precio = Double.parseDouble (Inventario[Prod][3]);
        double Total = precio * cantidad;

        //esto es para la fecha y hora de la venta! yuju :3
        LocalDateTime actual = LocalDateTime.now ();
        DateTimeFormatter formato = DateTimeFormatter.ofPattern ("DD-MM-YYYY HH-mm:ss");
        String FechaHora = actual.format(formato);
```

```java
        // Registramos la venta en un archibo de texto
        try {
            FileWriter fw = new FileWriter("Venta.txt", true);
            PrintWriter pw = new PrintWriter(fw);
            pw.println("--------------------");
            pw.println("Ventas Efectuadas: ");
            pw.println("Hora y Fecha de la transacción: " + FechaHora);
            pw.println("Producto vendido: " + Inventario[Prod][1]);
            pw.println("Cantidad vendida del producto es : " + cantidad);
            pw.println("el total a pagar es de: " + Total + " Qeutzales");
            pw.println("----------------------");
            pw.close();
        } catch (IOException e) {
            System.out .println("Error 007: ocurrio un error al registrar la venta en el archivo");
            e.printStackTrace();
        }
        System.out .println("La venta fue efectuada");
        System.out .println("Fueron vendidas: " + cantidad + " unidades de :" + Inventario[Prod][1]);
        System.out .println("La cantidad actual del producto es: " + Stock3);
        System.out .println("La cantidad a pagar es :" + Total + "Quetzales");
        VerAccion (vendedor, "venta de:  " + cantidad + "unidades " + Inventario[Prod][1], "corecta");
    } else {
        System.out .println("Error 008: el producto con el codigo : " + CodVenta + " No fue encontrado");
        VerAccion (vendedor, "Venta del producto, producto no encontrado ", "fallida");
    }
}
```

Siguiendo con las ultimas opciones tenemos la quinta opción, la cual es la de generar reportes de ventas y stock, para arrancar con esto vamos con el mayor lio que tuve… ¿qué librerías usar?, inicialmente iba a usar pdfbox, pero no me termino de convencer por qué no sabía usarlas, luego con itext5 pero este me dio muchos dolores de cabeza, y por ultimo me decante por el itext7 con el repositorio del auxiliar(Gracias) y el video de un yutuber de programación español, pude hacerme una idea de cómo usar las librerías, primero tuve que descargar una carpeta (se llama itext7-core-7) con archivos .jar, luego las agregue todas al NetBeans, use unas 5 librerías de itext, para párrafos, tablas para crear y escribir en archivos pdf.

```java
//Okeeeeey.... hice una clase que no estaba en el proyecto y tuve que hacer chanchuyo
//a veces me preocupa lo despistadoo que soy
package proyecto1;

/**
 *
 * @author kiquemarroquin
 */
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.InputMismatchException;
import java.util.Scanner;

//Liberias de itext7
import com.itextpdf.kernel.pdf.PdfDocument;
import com.itextpdf.kernel.pdf.PdfWriter;
import com.itextpdf.layout.Document;
import com.itextpdf.layout.element.Paragraph;
import com.itextpdf.layout.element.Table;
//Las librerias que leen, que encuentran y que hacen excepciones si no encuentran arhicvos de audio
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import static proyecto1.ValidarAccion.VerAccion;
```

Empecé haciendo una pequeña opción de si alguien quisiera generar ya sea un reporte de stock, o uno de ventas, con su debido llamado en cada caso, sea “ReporteStock” para reporte de stock y “ReporteVenta” para el reporte de ventas, para el reporte de stock hice llamado de las librerías de formato de hora y fecha, y de hora local (el de la compu) y usando el PdfDocument para crear el pdf con el nombre de hora y fecha.pdf. y con el parámetro “esc” escribir en el, con la variable “tabla”, agregamos celdas para: nombre, código, categoría, precio y stock, para cada producto registrado, con su debida información.

```java
            e.printStackTrace();
        } catch (Exception e) {
            System.out.println("Error 011: ocurrio un problema al general el reporte");
            e.printStackTrace();
            VerAccion(vendedor, "Generar reporte de stock ", "fallida");

        }
    }
```

Para el reporte de ventas siguiendo la misma lógica que el pdfwriter se encarga de crearlo, el pdfdocumet representando el documento en la memoria del programa, el

document agregar el titulo con la hora y fecha del reporte, como acá no se agregan tablas no se usa, más que nada es texto en esta parte, haciendo búsqueda del archivo de texto, ojo ese se debe llama EXACTAMENTE IGUAL (Me trajo muchos errores eso), o no encontrara el archivo y bueno… no generara nada.

```
//Una funcion para el reporte de ventas, asi mejor, y no se ve todo desordenado
public static void ReporteVenta (String vendedor) {
   LocalDateTime HFactual = LocalDateTime.now ();
   //ya lo dije arriba :P
   DateTimeFormatter HFormato = DateTimeFormatter.ofPattern ("dd_MM_yyyy_HH_mm_ss");
   //nombramos al otro archivo
   String nomArchiv = HFactual.format(HFormato) + "_Ventas.pdf";

   try {
      // se ordeno muy feo... asi que se vaya
      // aca declaro las variables de escribir en el archivo, de documento, de lectura y escrituta en archivo de texto
      PdfWriter Escribir = new PdfWriter(nomArchiv);
      PdfDocument pdf = new PdfDocument(Escribir);
      Document documento = new Document(pdf);
      Scanner Fs = new Scanner(new FileReader("Venta.txt"));
      {
      }
      documento.add(new Paragraph("Historial de Ventas - " + HFactual.format(DateTimeFormatter.ofPattern ("dd7mm/yyyy HH:mm:ss"))));
      while (Fs.hasNextLine()) {
         documento.add(new Paragraph(Fs.nextLine()));
      }
      documento.close();
      System.out .println("Reporte de venta generdado en : " + nomArchiv);
      VerAccion (vendedor, "Generar reporte de ventas ", "Correcta");
   } catch (FileNotFoundException e) {
      System.out .println("Error 012: el archivo de ventas.txt no se encontró, no hay ventas por reportar");
      VerAccion (vendedor, "Generar reporte de venta, no se encontro el archivo ", "fallida");
   } catch (IOException e) {
      System.out .println("Error 013: ocurrio un error de lectura o escritura");
      VerAccion (vendedor, "Generar reporte de ventas ", "fallida");
      e.printStackTrace();
   }
  }
}
```

Siguiendo con la séptima opción y la que más quebraderos de cabeza y fatiga mental me trajo, la bitácora de acciones, en esta cree una clase solo para ella, de igual manera hice llamado de las librerías de hora local y formato de hora, ahora, al igual que las ventas se registran en un archivo de texto, imprimir en el documento la fecha y hora de la acción , mi nombre, que acción se hizo, si salió bien o mal, y cree otro método para mostrar en pantalla a partir del archivo de texto de la bitácora(por la librería de file scanner) QUE TAMBIEN SE DEBE LLAMA IGUAL(Dios cuanto sufrimiento), y como último, como en el enunciado decía que debía ser “temporal” y el coco no daba más de si, entonces hice una función para limpiar el historial de acciones y las ventas, donde en vez de sobrescribir el archivo lo que hará es “borrar” (en realidad lo que hace es añadir lo último que se hizo pero como no se hizo nada lo borra) el contenido del archivo txt de ventas y bitácora.

```java
/
public class ValidarAccion {
    // una funcion para registrar cada accion e imprimirlas en un archivo de texto

    public static void VerAccion (String vendedor, String accion, String estado) {
        try (FileWriter Fw = new FileWriter("Bitacora.txt", true); PrintWriter pw = new PrintWriter(Fw)) {

            LocalDateTime Ahora = LocalDateTime.now ();
            DateTimeFormatter Formato = DateTimeFormatter.ofPattern ("dd_mm_yyyy HH:mm:ss");
            String fechaHora = Ahora.format(Formato);

            pw.println("-----------------------------------------------");
            pw.println("--------BITACORA DE ACCIONES--------");
            pw.println("Fecha y Hora: " + fechaHora);
            pw.println("Usuario: " + vendedor);
            pw.println("Accion :" + accion);
            pw.println("Estado :" + estado);
            pw.println("-----------------------------------------------");

        } catch (IOException e){
            System.out .println("Error 013: Ocurrio un error al registrar la accion en la bitaora");
        }
    }
    //funcion para mostar la Bitacora
    public static void MostarAcciones (){
        System.out .println("-------Historial de acciones----------");
        try(Scanner Fs = new Scanner(new File("Bitacora.txt"))){
            while(Fs.hasNextLine()) {
                System.out .println(Fs.nextLine());
            }
        } catch(FileNotFoundException e){
                System.out .println("Error 014: No se ha encontrado el archivo de la bitacora");
        }
    }
    //Pense que como TECNICAMENTE los archivos son temporales....
//me saque del coco una forma para limpiar los archivos de texto, me regañaran? lo dudo
    public static void LimpHistorial (){
        try {
            //  limpiar la bitacora
            new FileWriter("Bitacora.txt", false).close();
            System.out .println("La bitacora fue limpiada :D");

            //Limpiar las ventas
            new FileWriter("Venta.txt", false).close();
            System.out .println("Historial de ventas fue limpiado :D");
        } catch(IOException e){
            System.out .println("Erro 015: Ocurrio un error al intentar limpiar los archivos");
        }
    }
}
```

¿Qué problemas hubo durante el desarrollo de la práctica?

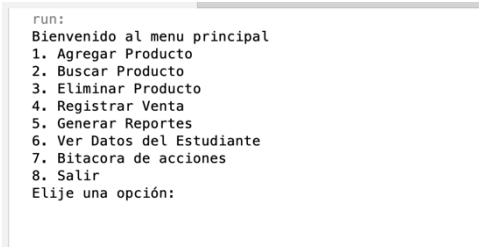
Durante este proyecto hubo múltiples problemas en su mayoría, por descuidos míos otros fueron por inexperiencia al usar ciertas librerías, a continuación, mencionaré ciertos problemas que fueron los que más dificultades me trajeron:

- Uno de los problemas que acarreaba desde la práctica número uno fue que al ingresar un código o nombre y éste esté repetido se creaba un bucle infinito de, “ingrese el código”.
- A la hora implementar el Try Catch en el menú este me provocaba que el programa no corriese.
- Durante el registro de las ventas en un archivo de texto al imprimir esto no me dejaba porque me salía el siguiente mensaje: “not directory find”.
- A la hora de registrar un producto y querer venderlo el sistema asaltaba el proceso de ingresar el código y mandaba directamente que no existía ningún producto e incluso si ya había ingresado alguno anteriormente.
- A la hora de hacer la bitácora de acciones no encontraba el archivo de texto correspondiente para generarlo.
- Para el reporte de ventas el programa no detectaba el archivo de texto para generar el archivo en PDF.
- A la hora de empezar a ver las opciones para generar un PDF inicialmente iba a usar PDF Box, pero no encontraron los archivos correspondientes .jar para llevarlo a cabo, luego me canté por iText5, pero hasta donde yo tengo entendido sólo funciona en versiones anteriores de NetBeans, y luego por decirme para usar iText7, no encontraba los archivos correspondientes para cargarlos al IDE.
- Al crear la clase de reportes no me permitía importarla a la clase principal.
- A leer otra vez el enunciado del proyecto me percaté que la bitácora es de archivos temporales mientras el programa esté abierto lo que significa que al cerrar programa esos archivos se deben eliminar permanentemente, no lograba conseguir este resultado.

¿Cómo pude solucionar estos problemas?

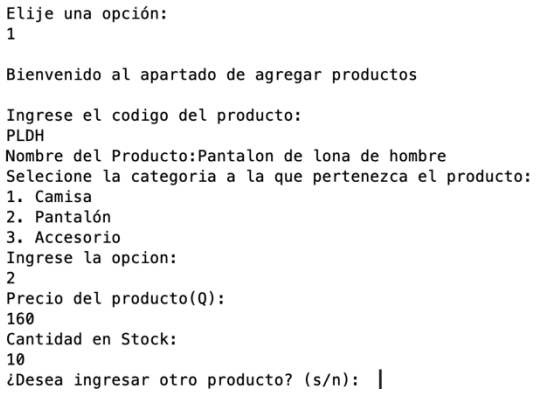
- A la hora de querer solucionar el problema de este bucle infinito, lo que se me ocurrió fue declarar la variable booleana antes del ciclo “Do While”, esto para evitar que el código repetido sea falso en la declaración para así poder solucionar este problema.
- A la hora de implementar “Try-catch” en el menú principal cree una clase pura parte con el nombre de verificar número donde verifica si el valor está entre 1 y 8, y para el Stock y valor del precio del producto cree un método llamado “ver número positivo” donde igual manera va a mandar una condición si se cumple que el número es menor a cero que retorne al ingreso de valor, y si se ingresa algo que no sea un número de igual manera mandará un mensaje que indique que vuelva a ingresar un valor válido.
- Para el problema del “not directory find”, lo que debía hacer es nombrar el archivo exactamente como quiero que lo busque el programa, si el archivo se llama ventas exactamente tiene que nombrarse así para que el programa encuentre el archivo y sobrescriba en el.
- Para este problema que se saltaba el proceso de ingresar el código del producto lo que tuve que hacer fue agregar un salto de línea que consumiera en el menú un Enter y de esta manera poder ingresar el código para la venta del producto.
- De igual manera que en el problema de qué no encontraba el archivo de texto para registrar las ventas el archivo de la bitácora, aunque esté en blanco debe nombrarse exactamente como el programa quiere que lo busque si se llama:” bitácora.txt” se debe nombrar así o el programa no lo va a encontrar y por consiguiente no buscara nada.
- Para este problema lo que tuve que hacer es dirigirme a un canal de YouTube de un youtuber español que tenía exactamente la carpeta que yo necesitaba con todos los archivos de Java para importarlos al IDE, y con el repositorio del auxiliar me pude guiar para llenar las tablas que contendrá el archivo PDF.
- Para este problema de qué no podía importar la clase reportes a la clase principal lo que tuve que revisar fue donde cree la clase y la clase se creó en un paquete aparte del proyecto lo que tuve que hacer fue crear una clase dentro del proyecto como siempre y copiar y pegar lo que ya tenía en la nueva clase e impórtala en la clase principal.
- Lo que hice en este caso fue crear un método para limpiar los archivos de texto de registro de ventas y de bitácora lo que hace este método es sobre escribir el archivo, pero como no se ha hecho nada anteriormente escribirá la “nada” en el archivo limpiándolo.

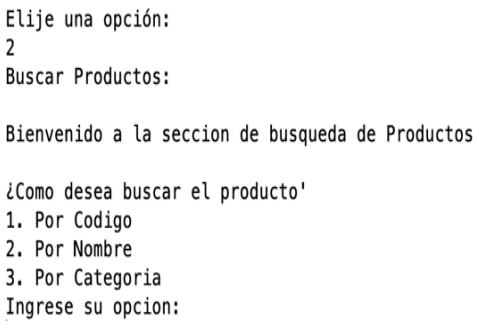
MENU INTERACTIVO



Es éste es el menú principal para ingresar cada opción se agrega un número del uno al ocho y se debe dar doble Enter. Al terminar cada opción nos dirigirá automáticamente al menú principal.

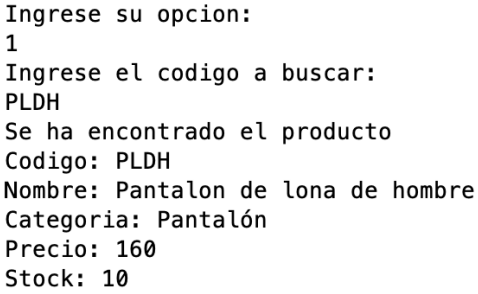
Al ingresar la opción uno nos aparecerá el menú para agregar un producto primero pidiéndonos, el código, el nombre, la categoría, el precio y la cantidad de Stock disponible, el programa automáticamente nos preguntarás si queremos agregar otro producto.





Al elegir la opción dos nos saldrá este menú donde nos permitirá buscar el producto por código, por nombre y por su categoría, en las opciones uno y dos no importa si hay mayúsculas y minúsculas en opción tres al buscar por nombre pantalón hay que incluir la tilde en la categoría a buscar

A elegir cualquier de las tres opciones no saldrá la opción para buscar el producto al encontrar el producto nos mostrará todos los datos del producto.



```
Elije una opción:
3
Eliminar Productos

Ingrese el Codigo del producto que se quiera eliminar:
PLDH
Se encontro el siguiente producto:
Se ha encontrado el producto
Codigo: PLDH
Nombre: Pantalon de lona de hombre
Categoria: Pantalón
Precio: 160
Stock: 10

¿Esta seguro de la eliminacion del producto (s/n):
```

Al elegir la tercera opción no saldrá el menú para eliminar productos, de igual manera nos pedirá el código del producto para eliminarlo y no saltará una última opción por si queremos eliminarlo o no de ser el último caso escribimos la letra N caso contrario escogeremos la letra S para eliminar el producto.

A elegir la opción S eliminaremos el producto, el programa automáticamente nos preguntará si queremos eliminar otro producto si elegimos si se eliminará otro producto.

```
¿Esta seguro de la eliminacion del producto (s/n):
s
El producto fue eliminado
¿Desea eliminar otro producto? (s/n):  s
No hay productos para eliminar

¿Desea eliminar otro producto? (s/n):  n
```

```
Elije una opción:
4
Registrar Ventas
Ingrese el codigo del producto a vender:
PLDH
Se ha encontrado el producto
Codigo: PLDH
Nombre: Pantalon de lona de hombre
Categoria: Pantalón
Precio: 150
Stock: 10
Ingrese la cantidad del produco que se va a vender
```

Al elegir la cuarta opción nos saltará el menú para registrar ventas al ingresar el código del producto y éste está disponible en el Stock nos mostrará los datos del producto a vender y nos mostrará cuánta cantidad del producto vamos a vender.

Al ingresar la cantidad que queremos vender nos mostrará la cantidad de producto actual del Stock y cuánto nos salió la compra, estos datos se guardan en un archivo de texto con el nombre de venta

```
Ingrese la cantidad del produco que se va a vender
5
La venta fue efectuada
Fueron vendidas: 5 unidades de :Pantalon de lona de hombre
La cantidad actual del producto es: 5
La cantidad a pagar es :750.0Quetzales
```

Venta.txt

```
----------------------
Ventas Efectuadas:
Hora y Fecha de la transacción: 255-09-2025 12-:37:47
Producto vendido: Pantalon de lona de hombre
Cantidad vendida del producto es : 5
El total a pagar es de: 750.0 Qeutzales
----------------------
```

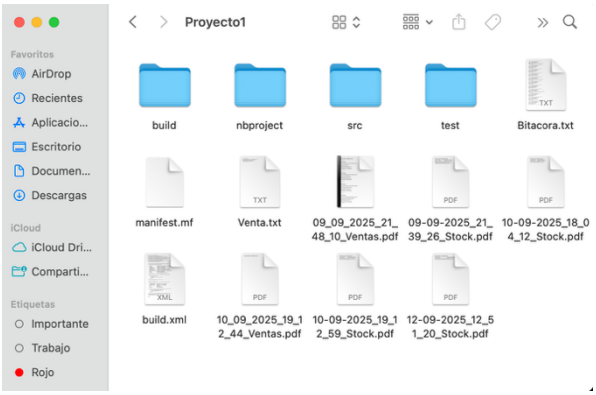
```
Elije una opción:
5
Generar Reportes

¿Qué reporte se desea generar?
1. Reporte de Stock
2. Reporte de ventas
Ingresa tu opcion:
```

A elegir la quinta opción nos saldrá un submenú con las opciones para generar reportes ya sea de Stock o de ventas.

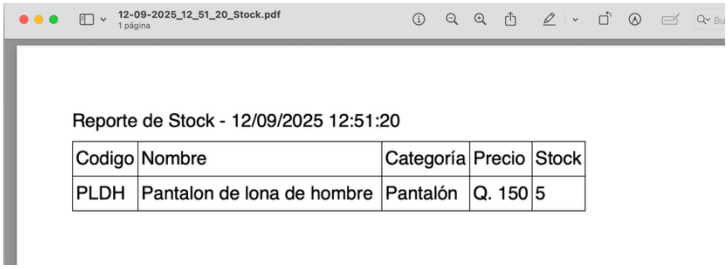
Al ingresar ya sea la opción uno o dos el archivo se guardará con la fecha actual del dispositivo.

```
Ingresa tu opcion:
1
El reporte de stock fue generado en :12-09-2025_12_51_20_Stock.pdf
```



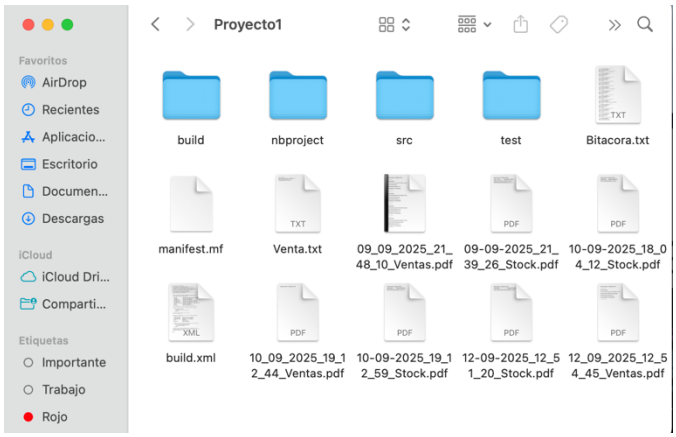
El archivo se guardará en la carpeta del proyecto, ya sea el registro de ventas como el reporte de stock.

Al abrir el archivo PDF se mostrará la información del producto con su disponibilidad.




```
Ingresa tu opcion:
2
Reporte de venta generdado en : 12_09_2025_12_54_45_Ventas.pdf
```

De igual manera al ingresar la opción dos se hará un archivo en PDF de las ventas realizadas, con el nombre de la fecha actual del dispositivo.



El archivo de ventas en PDF se guardará en la carpeta donde se está realizando este proyecto.

El archivo de las ventas registrará todas las ventas realizadas, basándose en el archivo de texto con el nombre venta.

Historial de Ventas - 12754/2025 12:54:45

Ventas Efectuadas:

Hora y Fecha de la transacción: 255-09-2025 12-:37:47

Producto vendido: Pantalon de lona de hombre

Cantidad vendida del producto es : 5

el total a pagar es de: 750.0 Qeutzales

```
Elije una opción:
6
Ver Datos del Estudiante
Este programa fue realizado por Guillermo Enrique Marroquin Morán.
Carnet: 202103527.
Este programa si tiene Derechos de autor
```

Al elegir la sexta opción se mostrará los datos del estudiante, osea yop.

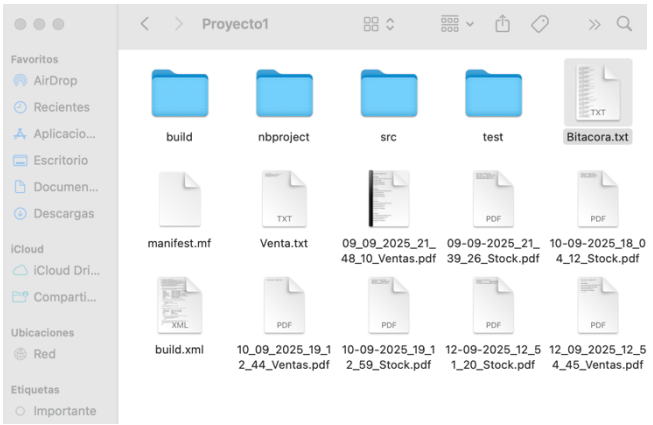
Al elegir la séptima opción nos saldrá un menú donde nos indique si queremos generar un archivo con bitácora de acciones realizadas o si queremos limpiar este mismo.

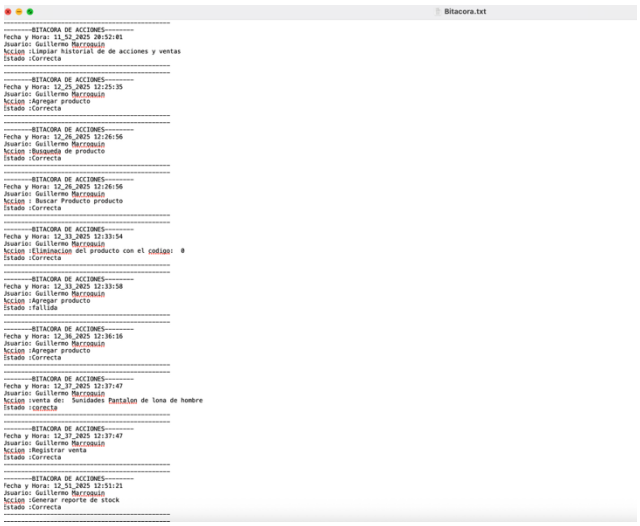
```
Elije una opción:
7
Bievenido a las opciones de la bitaciora de acciones
1. Ver bitacora de acciones :
2. Limpiar bitacora e historial de ventas :
Elije una opcion:
```

```
Bievenido a las opciones de la bitaciora de acciones
1. Ver bitacora de acciones :
2. Limpiar bitacora e historial de ventas :
Elije una opcion:
1
-------Historial de acciones----------
------------------------------------------------
--------BITACORA DE ACCIONES--------
Fecha y Hora: 11_52_2025 20:52:01
Usuario: Guillermo Marroquin
Accion :Limpiar historial de de acciones y ventas
Estado :Correcta
------------------------------------------------
------------------------------------------------
--------BITACORA DE ACCIONES--------
Fecha y Hora: 12_25_2025 12:25:35
Usuario: Guillermo Marroquin
Accion :Agregar producto
Estado :Correcta
------------------------------------------------
------------------------------------------------
```

Tanto en la consola del programa con un archivo en texto nos indicará todas las acciones realizadas en este proyecto junto a la hora y fecha del dispositivo.

El archivo de la bitácora se guardará con el nombre de bitácora en la carpeta del proyecto.





Al abrir el archivo de texto nos mostrará también todas las acciones realizadas.

A elegir de nuevo la opción siete y elegimos esta es la opción dos nos preguntará si queremos eliminar todo el historial de acciones realizadas, si queremos hacerlo escribimos una “s” de lo contrario escribimos una “n”.

```
Elije una opción:
7
Bievenido a las opciones de la bitaciora de acciones
1. Ver bitacora de acciones :
2. Limpiar bitacora e historial de ventas :
Elije una opcion:
2
¿Esta seguro de eliminar el historial? (s/n)
```

```
¿Esta seguro de eliminar el historial? (s/n)s
La bitacora fue limpiada :D
Historial de ventas fue limpiado :D
```

Al elegir la “s” se limpiará el historial de ventas y el historial de la bitácora, de los archivos de texto.

Al elegir la octava y última opción saldremos del programa y nos imprimirá un mensaje de despedida.

```
Elije una opción:
8
¡Feliz tarde, adiós!
BUILD SUCCESSFUL (total time: 55 minutes 33 seconds)
```

EGRAFIA

Caules, C. Á. (2022, February 2). *Java Try Catch y su manejo*. Arquitectura Java. Retrieved September 12, 2025, from https://www.arquitecturajava.com/java-try-catch-y-su-manejo/

Divertitto, A. (2024, February 2). *¿Cómo obtener un carácter de nueva línea en Java?* CodeGym. Retrieved September 12, 2025, from https://codegym.cc/es/groups/posts/es.577.como-obtener-un-caracter-de-nueva-linea-en-java-

González, J. D. M. (2020, April 26). *Final y constantes*. Retrieved September 12, 2025, from https://www.programarya.com/Cursos/Java/Sistema-de-Tipos/Final-y-Constantes

Miadelets, O. (2025, April 15). *Matrices en Java*. CodeGym. Retrieved September 12, 2025, from https://codegym.cc/es/groups/posts/es.142.matrices-en-java

Galarraga, E. (2018, March 31). *Como guardar pdf*. Stack Overflow En Español. Retrieved September 12, 2025, from https://es.stackoverflow.com/questions/151942/como-guardar-pdf

Equipo editorial de IONOS. (2025, January 7). *Java primitives: tipos de datos elementales*. IONOS Digital Guide. Retrieved September 12, 2025, from https://www.ionos.com/es-us/digitalguide/paginas-web/desarrollo-web/primitivos-de-java/

David. (2020, April 21). *Busqueda de una Cadena en una matriz en Java*. Stack Overflow En Español. Retrieved September 12, 2025, from https://es.stackoverflow.com/questions/348115/busqueda-de-una-cadena-en-una-matriz-en-java

Miadelets, O. (2025a, February 13). *String equalsIgnoreCase() method in Java*. CodeGym. Retrieved September 12, 2025, from https://codegym.cc/groups/posts/string-equalsignorecase-method-in-java

makigas. (2022, November 23). *Cómo usar un FileReader y un BufferedReader – Curso de Java IO* [Video]. YouTube. Retrieved September 12, 2025, from https://www.youtube.com/watch?v=yy3DA7v6cIU

Santiago. (2016, October 4). *¿Cómo eliminar un objeto de un arreglo de objetos en Java?* Stack Overflow En Español. Retrieved September 12, 2025, from https://es.stackoverflow.com/questions/26024/c%C3%B3mo-eliminar-un-objeto-de-un-arreglo-de-objetos-en-java

Selawsky, J. (2024, February 8). *BufferedReader y BufferedWriter*. CodeGym. Retrieved September 12, 2025, from https://codegym.cc/es/groups/posts/es.150.bufferedreader-y-bufferedwriter

Portianko, V. (2025, January 9). *Java PrintWriter Class*. CodeGym. Retrieved September 12, 2025, from https://codegym.cc/groups/posts/java-printwriter-class

W3 Schools (Ed.). (n.d.). *W3Schools.com*. W3 Schools. Retrieved September 12, 2025, from https://www-w3schools-com.translate.goog/java/java_files_create.asp?_x_tr_sl=en&_x_tr_tl=es&_x_tr_hl=es&_x_tr_pto=tc

DiscoDurodeRoer. (2021, August 10). *Ejercicios Java - PDF #1 - Crear un PDF con IText 5* [Video]. YouTube. Retrieved September 12, 2025, from https://www.youtube.com/watch?v=PO4mwNzpwJA

Punto y Coma. (2020, May 1). *Crear Archivos PDF en Java , usando PDFBox* [Video]. YouTube. Retrieved September 12, 2025, from https://www.youtube.com/watch?v=XAFA3IQXTK0

DiscoDurodeRoer. (2023, September 10). *GitHub - DiscoDurodeRoer/ejercicios-java-youtube: Ejercicios de Java del canal de discoduroderoer*. GitHub. Retrieved September 12, 2025, from https://github.com/DiscoDurodeRoer/ejercicios-java-youtube.git

Brax. (2015, April 26). *Java - How to Clear a text file without deleting it?* Stack Overflow. Retrieved September 12, 2025, from https://stackoverflow-com.translate.goog/questions/29878237/java-how-to-clear-a-text-file-without-deleting-it?_x_tr_sl=en&_x_tr_tl=es&_x_tr_hl=es&_x_tr_pto=sge

Novo, J. Y. (2016, February 24). *Java iText PDF - Creando un pdf en Java con iText - Código Xules*. Código Xules. Retrieved September 12, 2025, from https://codigoxules.org/java-itext-pdf-creando-pdf-java-itext/

Pablooliva. (2025, September 6). *GitHub - 27Pablooliva27/IPC1C2S2025*. GitHub. Retrieved September 12, 2025, from https://github.com/27Pablooliva27/IPC1C2S2025.git

Brax. (2015b, April 26). *Java - How to Clear a text file without deleting it?* Stack Overflow. Retrieved September 12, 2025, from https://stackoverflow-com.translate.goog/questions/29878237/java-how-to-clear-a-text-file-without-deleting-it?_x_tr_sl=en&_x_tr_tl=es&_x_tr_hl=es&_x_tr_pto=sge

Data Camp. (n.d.). *Java File Handling*. Retrieved September 12, 2025, from https://www.datacamp.com/es/doc/java/category/file-handling