📂 Preentrega de proyecto

⚙️ Revisión de progreso

Obligatorio | Entregable

Revisión de progreso

Estamos acercándonos a un **punto clave en el desarrollo del proyecto**, donde tendrán la oportunidad de mostrar lo que han aprendido hasta ahora. Es un excelente momento para **compartir su progreso y recibir comentarios** que les ayudarán a pulir detalles y afinar sus ideas antes de la entrega final.

Aprovechen esta instancia para obtener una visión más clara de su trabajo y hacer los ajustes necesarios para llegar a un resultado óptimo.

Entregable

Como parte de tu avance en el curso y para prepararte para el desarrollo del **Trabajo Final Integrador (TFI)**, se te solicita que diseñes un sistema básico que permita gestionar información inicial sobre los productos de la empresa. Este proyecto servirá como la base sobre la cual construirás funcionalidades más complejas en las próximas semanas. Así, **podrán recibir comentarios que les permitirán fortalecer su proyecto y enfocarse en los detalles clave para la etapa final**.

⚙️ Revisión de progreso

Obligatorio | Entregable

Revisión de progreso

Esta instancia evaluativa es de carácter obligatorio y es un punto clave dentro de la cursada ya que nos permitirá evaluar tu progreso en el recorrido y asegurar que estás en el camino correcto en la construcción del **Proyecto Final Integrador**.

Este espacio de entrega está conformado por:

Cuestionario de Autoevaluacion

Preentrega de Proyecto

⚙️ Preentrega de proyecto

Obligatorio | Entregable

A partir de la **Clase N° 8** tendrás **7 días** de corrido para realizar la autoevaluación y la entrega en el campus virtual

---

Cuestionario de autoevaluación

- Te permitirá reflexionar sobre tu propio aprendizaje, progreso y cumplimiento de las consignas o rúbricas previamente establecidas y en caso de ser necesario realizar las modificaciones o ajustes correspondientes antes de realizar la preentrega.
- Se encontrará disponible en la Ruta N°2 de Campus Virtual.

Preentrega de proyecto:

- Se evaluará la aplicación de los conocimientos adquiridos en la realización de un proyecto.
- La realización progresiva de los "**Ejercicios prácticos**" los guiará paso a paso hacia la realización de la "**Preentrega**" y el "**Proyecto Integrador Final**"

⚙️ Ejercicio práctico

Obligatorio | Entregable

Vas a desarrollar un programa en Java que cumpla con las siguientes características:

Objetivo general

- Diseñar una aplicación en Java que permita **registrar, mostrar y gestionar productos**, así como **crear pedidos** que involucren varios productos. El sistema deberá hacer uso de **variables, operadores, colecciones (listas), POO (clases, objetos, encapsulamiento, herencia, polimorfismo), excepciones y organización de código en paquetes/módulos**.

⚙️ Ejercicio práctico

Obligatorio | Entregable

Requerimientos

Ingreso de productos

- El sistema debe permitir **agregar productos** con la siguiente información mínima:
  - **Nombre** del producto (cadena de texto, por ejemplo "Café Premium").
  - **Precio** (double, puede tener decimales).
  - **Cantidad en Stock** (int).
- Estos productos deben almacenarse en una **colección** (por ejemplo, `ArrayList<Producto>`).

Visualización de productos

- Debe haber una **funcionalidad** que liste en pantalla todos los productos registrados, mostrando su **ID (o posición)**, **Nombre**, **Precio** y **Cantidad en Stock**.
- El ID puede ser autogenerado (por ejemplo, un contador estático) o la posición en la lista.

⚙️ Ejercicio práctico

Obligatorio | Entregable

Búsqueda y actualización de productos

- El sistema permitirá **buscar un producto** por su nombre o ID.
- Si se encuentra el producto, se mostrará su información completa.
- Opcionalmente, se podrá **actualizar** alguno de sus datos (precio o stock), validando que los valores sean coherentes (por ejemplo, que el stock no sea negativo).

Eliminación de productos

- El sistema debe permitir **eliminar** un producto de la lista, identificándolo por su ID o posición en la colección.
- Antes de eliminar, el sistema podría pedir confirmación (opcionales).

⚙️ Ejercicio práctico

Obligatorio | Entregable

Creación de pedidos

- Además de manejar productos, se sugiere agregar la **clase Pedido** (o `Orden`) que contenga:
  - Una **lista de productos** asociados.
  - Cantidad deseada de cada producto (por ejemplo, usando un objeto intermedio `LineaPedido` o similar).

- El sistema debe permitir **crear** un pedido nuevo:
  a. Solicitar al usuario qué productos desea y en qué cantidad (validar que haya suficiente stock).
  b. Calcular el **costo total** (sumando `precio * cantidad` de cada producto).
  c. **Disminuir el stock** de cada producto si el pedido se confirma.
- Debe haber una **funcionalidad** para **mostrar** los pedidos realizados y su costo total, así como la lista de productos asociados.

⚙️ Ejercicio práctico

Obligatorio | Entregable

Menú principal interactivo

El programa presentará un **menú** con opciones, por ejemplo:

- **Agregar producto**
- **Listar productos**
- **Buscar/Actualizar producto**
- **Eliminar producto**
- **Crear un pedido**
- **Listar pedidos** (opcionales)
- **Salir**

El menú se repetirá hasta que se elija la opción de **Salir**.

Ejemplo del menú de opciones:

```
==========================================
==== SISTEMA DE GESTIÓN - TECHLAB
==========================================

1) Agregar producto
2) Listar productos
3) Buscar/Actualizar producto
4) Eliminar producto
5) Crear un pedido
6) Listar pedidos
7) Salir

Elija una opción:
```

⚙️ Ejercicio práctico

Requerimientos técnicos

- **Tipos de datos y variables**

Emplear **variables** de tipo `int` (para cantidades e IDs), `double` (para precios), `String` (para nombres/descripciones), y `boolean` si fuera necesario.

Asegurate de **usar operadores aritméticos, lógicos y relacionales** en las funciones de cálculo y validación.

- **Colecciones (Arrays, Listas)**

Para manejar los productos, se sugiere un `ArrayList<Producto>`.

Para manejar los productos dentro de un pedido, podría usarse otra lista, por ejemplo `ArrayList<LineaPedido>`.

O bien, un `Map<Integer, Integer>` si querés asociar ID de producto con cantidad solicitada (detalles a tu elección).

⚙️ Ejercicio práctico

Requerimientos técnicos

- **POO y Colaboración de Clases**

Clase Producto: con atributos `id`, `nombre`, `precio`, `stock`, getters y setters.

Clase Pedido (u Orden): con atributos `id`, `lista de productos/lineas`, `metodos` para calcular total, etc.

Clase Principal (Main): orquesta el menú, invoca métodos de servicios (por ejemplo, un `ProductoService` que encapsule la lógica de agregar/buscar/eliminar).

- **Herencia/Polimorfismo (opcional para extender)**

Si deseás, podés crear subclases de `Producto` (por ejemplo, `Bebida`, `Comida`) con atributos específicos (fecha de vencimiento, volumen en litros, etc.).

Mostrar cómo el polimorfismo ayuda a tratar distintos productos de forma genérica.

⚙️ Ejercicio práctico

Requerimientos técnicos

- **Excepciones**

Manejar errores con **try/catch**. Por ejemplo, al convertir datos ingresados por la usuaria o usuario, podrías atrapar `NumberFormatException` si ingresa texto en lugar de un número.

Podrías crear una **excepción personalizada** como `StockInsuficienteException` y lanzarla cuando se intenta crear un pedido con cantidad mayor al stock disponible.

- **Paquetes/módulos (organización de código)**

Dividir las clases en **paquetes** lógicos:

- `com.techlab.productos` (para `Producto`, `Bebida`, etc.)
- `com.techlab.pedidos` (para `Pedido`, `LineaPedido`)
- `com.techlab.excepciones` (para excepciones personalizadas)

⚙️ Ejemplo de flujo de uso (escenario)

- Aparece un menú:
  a. Agregar Producto
  b. Listar Productos
  c. Buscar/Actualizar Producto
  d. Eliminar Producto
  e. Crear Pedido
  f. Listar Pedidos
  g. Salir
- Selecciona “1” para Agregar Producto. El programa pregunta el `nombre`, `precio`, `stock`. Se crea un objeto `Producto` y se agrega a la lista.
- Selecciona “2” para Listar Productos, y el sistema muestra todos los productos con su `id`, `nombre`, `precio`, `stock`.
- Selecciona “5” para Crear Pedido. El sistema pregunta cuántos productos va a agregar, pide ID de producto y cantidad. Verifica stock; si no hay suficiente, lanza `StockInsuficienteException` o un mensaje apropiado. Si se confirma, descuenta stock y crea el pedido en la colección de pedidos.
- Selecciona “7” para Salir. El programa finaliza.