
	IES LAGUNA DE TOLLÓN	
	CURSO 2024 - 2025	
	Módulo: Programación 1º DAM	
	Simulacro de examen bloque 1	

Actividad 1



Imagina un sistema para gestionar los préstamos de libros en una biblioteca. El sistema debe permitir gestionar tanto libros como socios y realizar el seguimiento de los préstamos de libros a los socios.

1. Clases a implementar:

- Producto:** Clase abstracta que debe tener los atributos idProducto (String) y nombre (String). El método disponibilidad(), que será implementado por las clases hijas para verificar si el producto está disponible.
- Libro:** esta clase debe ser una subclase de Producto y debe tener los atributos atributoISBN (String), titulo (String), autor (String), añoPublicacion (int) y disponible (boolean). Además de los métodos para obtener y modificar estos atributos, debe incluir un método que marque al libro como no disponible cuando se realice un préstamo.
- Socio:** Esta clase debe tener los atributos idSocio (String), nombre (String), direccion (String), telefono (String) y una lista de libros prestados. El socio debe poder realizar los métodos prestarLibro(Libro libro) y devolverLibro(Libro libro).
- Biblioteca:** La biblioteca debe gestionar los libros disponibles y los socios. Esta clase debe implementar la interfaz Gestionable (ver abajo) que define los métodos agregarLibro(Producto producto) y agregarSocio(Socio socio). Los métodos clave deben incluir:
 - agregarLibro(Producto producto): agrega un producto al inventario.
 - agregarSocio(Socio socio): agrega un socio al sistema.
 - listarLibrosDisponibles(): lista todos los libros disponibles.
 - prestarLibro(String idSocio, String ISBNLibro): permite a un socio pedir prestado un libro, lanzando una excepción si el libro no está disponible o si el socio no existe.
 - devolverLibro(String idSocio, String ISBNLibro): permite a un socio devolver un libro.

2. Excepciones personalizadas: Crea una excepción personalizada LibroNoDisponibleException que se lanzará si el libro solicitado no está disponible. Además, crea otra excepción SocioNoExistenteException que se lanzará si el socio no está registrado.

3. Interfaces: Crea una interfaz Gestionable con los métodos agregarLibro(Producto producto) y agregarSocio(Socio socio).

	IES LAGUNA DE TOLLÓN	
	CURSO 2024 - 2025	
	Módulo: Programación 1º DAM	
	Simulacro de examen bloque 1	

Actividad 2

Imagina que estás desarrollando un sistema para controlar la asistencia en una escuela. El sistema debe crear registros de asistencia y permitir gestionar estudiantes y clases. Además, debe guardar y cargar los registros de asistencia desde archivos.

1. Clases a implementar:

- Persona (clase abstracta):** Esta clase debe tener los atributos id (String), nombre (String) y apellido (String). Debe incluir un método abstracto registrarAsistencia(String clase, String fecha).
- Estudiante (subclase de Persona):** Esta clase debe heredar de Persona y agregar los atributos curso (String) y asistencias (una lista de objetos de la clase Asistencia). Debe implementar el método registrarAsistencia para registrar si el estudiante está presente o ausente en una clase.
- Profesor (subclase de Persona):** Esta clase debe heredar de Persona y agregar los atributos materia (String) y clasesImpartidas (una lista de objetos Clase). También debe implementar el método registrarAsistencia.
- Asistencia:** Esta clase debe tener los atributos fecha (String), estado (String), y idClase (String). Esta clase puede ser utilizada tanto por Estudiante como por Profesor.
- Clase:** Esta clase debe contener la información de una clase específica, incluyendo nombreClase (String), codigoClase (String), y una lista de estudiantes y profesores.



2. Métodos importantes:

- mostrarAsistenciaPorEstudiante(String idEstudiante): muestra todos los registros de asistencia de un estudiante.
- mostrarAsistenciaPorClase(String codigoClase): muestra la lista de estudiantes y su estado de asistencia para una clase.
- guardarAsistenciasEnArchivo(String nombreArchivo): guarda los registros de asistencia en un archivo.
- cargarAsistenciasDesdeArchivo(String nombreArchivo): carga los registros de asistencia desde un archivo.

3. Archivos:

Implementa la funcionalidad para guardar y cargar las asistencias en un archivo de texto. Los datos se deben guardar en el archivo en el siguiente formato:

idEstudiante;fecha;estado

	IES LAGUNA DE TOLLÓN	
	CURSO 2024 - 2025	
	Módulo: Programación 1º DAM	
	Simulacro de examen bloque 1	

Actividad 3

Diseña una aplicación para gestionar una tienda online. Los usuarios pueden comprar productos y realizar pagos. Implementa la interfaz **Vendible** y maneja las transacciones mediante un sistema de pago.

1. Clases a implementar:



- Producto (implementa la interfaz Vendible):** Esta clase debe tener los atributos idProducto (String), nombre (String), precio (double), y cantidadDisponible (int). El método disponibilidad() debe verificar si el producto está disponible.
- Vendible (interfaz):** Esta interfaz debe contener el método disponibilidad(), que será implementado por los productos para verificar su disponibilidad.
- Usuario:** Esta clase debe tener los atributos idUsuario (String), nombre (String), correoElectronico (String), y una lista de productos en el carrito. Los métodos clave deben ser:
 - agregarAlCarrito(Producto producto)
 - eliminarDelCarrito(Producto producto)
 - realizarPago(double total): Permite al usuario realizar el pago y vaciar el carrito.
- Carrito de Compras:** Esta clase debe tener una lista de productos y un método calcularTotal(). También debe incluir un método vaciarCarrito().
- Tienda:** La tienda debe gestionar una lista de productos y una lista de usuarios registrados. Los métodos clave deben incluir:
 - agregarProducto(Producto producto)
 - registrarUsuario(Usuario usuario)
 - mostrarInventario(): Muestra todos los productos disponibles en la tienda.

2. Métodos importantes:

- realizarPago(String idUsuario): Permite a un usuario realizar un pago por el total de su carrito de compras.
- guardarCarritoEnArchivo(String idUsuario, String nombreArchivo): Guarda el carrito de compras de un usuario en un archivo.
- cargarCarritoDesdeArchivo(String idUsuario, String nombreArchivo): Carga el carrito de un usuario desde un archivo.

3. Excepciones personalizadas:

Crea una excepción ProductoNoDisponibleException que se lanzará cuando un usuario intente agregar más productos al carrito de los que están disponibles en el inventario.

	IES LAGUNA DE TOLLÓN	
	CURSO 2024 - 2025	
	Módulo: Programación 1º DAM	
	Simulacro de examen bloque 1	

Actividad 4

Imagina un sistema de gestión bancaria que permite a los usuarios realizar transacciones en sus cuentas. Los usuarios pueden tener varias cuentas y las cuentas pueden ser de diferentes tipos: corriente y de ahorro.

1. Clases a implementar:

- Cuenta (interfaz de transacciones):** Esta clase debe ser **abstracta** e implementar la interfaz Transaccionable que contiene los métodos depositar(double cantidad) y retirar(double cantidad).
- CuentaCorriente (subclase de Cuenta):** Esta clase debe implementar la lógica específica para cuentas corrientes.
- CuentaAhorros (subclase de Cuenta):** Esta clase debe implementar la lógica específica para cuentas de ahorro.
- Usuario:** Esta clase debe tener los atributos idUsuario (String), nombre (String), apellido (String) y una lista de cuentas bancarias. Los métodos clave incluyen agregarCuenta(Cuenta cuenta) y listarCuentas().

2. Métodos importantes:

- realizarTransaccion(String idUsuario, String numeroCuenta, double cantidad, String tipoTransaccion): Permite realizar depósitos o retiros en una cuenta.
- guardarCuentasEnArchivo(String nombreArchivo): Guarda la información de todas las cuentas en un archivo de texto.
- cargarCuentasDesdeArchivo(String nombreArchivo): Carga la información de las cuentas desde un archivo de texto.