**Nivel Básico**

1. **Clase Empleado:**
   * Diseña una clase con atributos como nombre, puesto y salario.
   * Agrega métodos para aumentar el salario en un porcentaje dado.
   * Crea objetos y prueba aumentando el salario.
2. **Clase Fracción:**
   * Crea una clase con atributos numerador y denominador.
   * Implementa un método para simplificar la fracción.
   * Añade métodos para sumar, restar, multiplicar y dividir fracciones.

**Nivel Intermedio**

1. **Sistema de Notas:**
   * Crea una clase Estudiante con atributos nombre y notas (un arreglo o lista).
   * Implementa métodos para agregar notas y calcular el promedio.
   * Crea una clase Curso que contenga varios estudiantes, con métodos para calcular el promedio general del curso.
2. **Clases Abstractas - Figura Geométrica:**
   * Diseña una clase abstracta Figura con métodos abstractos como calcularArea() y calcularPerimetro().
   * Crea subclases como Cuadrado, Rectángulo y Círculo que implementen estos métodos.
3. **Sistema de Inventarios:**
   * Crea una clase Producto con atributos como nombre, precio y cantidadDisponible.
   * Diseña una clase Inventario que gestione varios productos.
   * Implementa métodos para agregar, vender y verificar stock de productos.

**Nivel Avanzado**

1. **Interfaces - Vehículos de Transporte:**
   * Define una interfaz Vehiculo con métodos como arrancar(), detener() y acelerar().
   * Implementa esta interfaz en clases como Bicicleta, Moto, y Coche.
   * Crea un programa que permita simular el comportamiento de diferentes vehículos.
2. **Gestión de Biblioteca:**
   * Diseña una clase Libro con atributos como título, autor y estado (prestado o disponible).
   * Crea una clase Usuario con atributos como nombre y una lista de libros prestados.
   * Implementa una clase Biblioteca que permita registrar usuarios, libros y gestionar préstamos y devoluciones.
3. **Polimorfismo - Tienda de Mascotas:**
   * Diseña una clase base Animal con atributos generales y un método hacerSonido().
   * Implementa subclases como Perro, Gato y Pez con sonidos únicos.
   * Simula una tienda que venda y muestre información de diferentes animales.
4. **Sistema de Gestión de Pedidos:**
   * Diseña una clase Producto con atributos como nombre, precio y cantidad.
   * Crea una clase Pedido que gestione una lista de productos comprados.
   * Implementa métodos para calcular el total del pedido y generar una factura.
5. **Simulador de Videojuego:**
   * Implementa una clase Personaje con atributos como nombre, salud y nivel.
   * Crea una clase Enemigo que tenga métodos para atacar al personaje.
   * Usa herencia para crear subclases como Guerrero y Hechicero con habilidades únicas.

**Reto Extra (Avanzado)**

**Sistema de Aerolíneas:**

* Diseña una clase Vuelo con atributos como codigoVuelo, origen, destino, y capacidad.
* Implementa una clase Pasajero con atributos como nombre y numeroDeAsiento.
* Crea métodos para reservar asientos, cancelar reservas y mostrar los detalles del vuelo.

EJERCICIOS MODULO2

**Nivel Básico**

1. **Clase Persona:**
   * Crea una clase Persona con atributos nombre, edad y nacionalidad.
   * Implementa métodos getters y setters para cada atributo.
   * Crea un método saludar() que imprima "Hola, soy [nombre]".
   * Prueba tu clase creando un objeto y utilizando sus métodos.
2. **Clase Rectángulo:**
   * Diseña una clase con atributos largo y ancho.
   * Implementa un método para calcular el área y el perímetro.
   * Crea un método para imprimir las dimensiones del rectángulo.
   * Haz pruebas creando objetos y utilizando los métodos.
3. **Clase CuentaBancaria:**
   * Define una clase con atributos como titular, saldo, numeroDeCuenta.
   * Implementa métodos depositar(monto) y retirar(monto) para actualizar el saldo.
   * Agrega restricciones para evitar retiros que dejen saldo negativo.

**Nivel Intermedio**

1. **Herencia - Vehículo:**
   * Diseña una clase base Vehiculo con atributos como marca, modelo y velocidad.
   * Crea clases derivadas como Coche y Moto que añadan atributos específicos (por ejemplo, puertas en Coche y tipoDeMotor en Moto).
   * Sobrescribe métodos en las clases derivadas para mostrar detalles específicos de cada vehículo.
2. **Polimorfismo - Animal:**
   * Diseña una clase abstracta Animal con un método abstracto hacerSonido().
   * Crea subclases como Perro y Gato que implementen el método de forma distinta.
   * Utiliza polimorfismo para iterar a través de un arreglo de animales y llamar al método hacerSonido() de cada objeto.
3. **Gestión de Productos:**
   * Implementa una clase Producto con atributos como nombre, precio, y cantidad.
   * Crea una clase Inventario que gestione una lista de productos.
   * Implementa métodos para agregar productos, eliminar productos y buscar un producto por nombre.

**Nivel Avanzado**

1. **Interfaces - Sistema de Pagos:**
   * Define una interfaz Pago con métodos como pagar() y calcularImpuestos().
   * Crea clases como PagoConTarjeta y PagoEnEfectivo que implementen la interfaz.
   * Calcula los pagos y los impuestos según cada método de pago.
2. **Clase Genérica - Contenedor:**
   * Diseña una clase genérica que funcione como un contenedor para objetos de cualquier tipo.
   * Incluye métodos agregarElemento(), eliminarElemento() y obtenerElemento().
   * Prueba la clase con tipos de datos diferentes como String, Integer, o tus propias clases.
3. **Patrón Singleton - Configuración Global:**
   * Implementa una clase Configuracion usando el patrón Singleton.
   * Asegúrate de que solo pueda existir una instancia de esta clase y proporciona un método para obtenerla.
4. **Sistema de Gestión Académica:**
   * Diseña un sistema con clases como Estudiante, Profesor y Curso.
   * Permite que los estudiantes se inscriban en cursos, y que los profesores asignen notas.
   * Usa colecciones para relacionar estudiantes con sus cursos.
5. **Sistema de Biblioteca (Avanzado):**
   * Implementa un sistema con clases Libro, Usuario y Prestamo.
   * Permite registrar usuarios y libros, y gestionar el préstamo y devolución de libros.
   * Usa estructuras como HashMap para gestionar el inventario de libros.

EJERCICIOS MODULO 3

### ****1. Gestión de Agenda de Contactos****

**Problema:** Diseña una aplicación para almacenar y gestionar una lista de contactos personales, incluyendo datos como nombre, teléfono, correo electrónico y dirección.

* **Clases sugeridas:** Contacto, Agenda.
* Métodos:
  + Agregar nuevos contactos.
  + Buscar contactos por nombre o número de teléfono.
  + Mostrar todos los contactos ordenados alfabéticamente.

### ****2. Sistema de Gestión de Tickets de Soporte****

**Problema:** Implementa un sistema para gestionar los tickets de soporte de clientes con información como estado del ticket (Abierto, En Progreso, Cerrado) y descripción del problema.

* **Clases sugeridas:** Cliente, Ticket, SistemaSoporte.
* Métodos:
  + Crear un nuevo ticket asociado a un cliente.
  + Cambiar el estado de un ticket.
  + Mostrar los tickets abiertos.

### ****3. Sistema de Reservas para Restaurantes****

**Problema:** Diseña un sistema para gestionar reservas de mesas en un restaurante con datos de clientes, fecha y hora de la reserva.

* **Clases sugeridas:** Mesa, Reserva, Restaurante.
* Métodos:
  + Reservar una mesa.
  + Mostrar la disponibilidad de mesas en una fecha específica.
  + Cancelar reservas.

### ****4. Simulador de Tienda Online****

**Problema:** Implementa una plataforma de e-commerce para gestionar productos y pedidos con información como precios y cantidades.

* **Clases sugeridas:** Producto, Carrito, Pedido.
* Métodos:
  + Agregar productos al carrito de compras.
  + Calcular el total del carrito.
  + Generar un pedido al confirmar la compra.

### ****5. Sistema de Gestión de Estacionamiento****

**Problema:** Diseña un sistema que permita administrar un estacionamiento con espacios de diferentes tamaños y vehículos compatibles.

* **Clases sugeridas:** Vehiculo, EspacioEstacionamiento, Estacionamiento.
* Métodos:
  + Registrar la entrada y salida de vehículos.
  + Mostrar el número de espacios disponibles por tamaño.

### ****6. Gestión de Inventario para una Librería****

**Problema:** Implementa un sistema para gestionar el inventario de una librería con información como título, autor, ISBN, y stock de libros.

* **Clases sugeridas:** Libro, Libreria.
* Métodos:
  + Agregar nuevos libros al inventario.
  + Vender un libro, reduciendo el stock.
  + Buscar libros por título o autor.

### ****7. Sistema de Gestión de Alquiler de Vehículos****

**Problema:** Diseña un sistema para gestionar reservas y disponibilidad de vehículos en una empresa de alquiler.

* **Clases sugeridas:** Vehiculo, Cliente, Reserva.
* Métodos:
  + Registrar la reserva de un vehículo.
  + Mostrar la disponibilidad de vehículos por categoría.
  + Cancelar una reserva.

### ****8. Simulación de Banco****

**Problema:** Implementa un sistema bancario para gestionar cuentas de clientes con funcionalidades como transferencia de dinero y consulta de saldo.

* **Clases sugeridas:** CuentaBancaria, Cliente, Banco.
* Métodos:
  + Crear cuentas bancarias (corriente o de ahorro).
  + Transferir dinero entre cuentas.
  + Consultar el saldo de una cuenta.

### ****9. Sistema de Clases Virtuales****

**Problema:** Diseña un sistema que permita gestionar clases virtuales, incluyendo el registro de estudiantes y el envío de notificaciones.

* **Clases sugeridas:** Profesor, Estudiante, ClaseVirtual.
* Métodos:
  + Crear clases virtuales con fecha y hora.
  + Registrar estudiantes en una clase.
  + Enviar notificaciones a los estudiantes registrados.

### ****10. Sistema de Gestión de Farmacia****

**Problema:** Diseña un sistema para gestionar medicamentos en una farmacia, con funcionalidad para procesar recetas y verificar disponibilidad.

* **Clases sugeridas:** Medicamento, Farmacia, Receta.
* Métodos:
  + Agregar medicamentos al inventario.
  + Verificar la disponibilidad de un medicamento.
  + Procesar recetas médicas y disminuir el stock.

EJERCICIOS MODULO 4

¡Entendido, Luis! Aquí te presento ejercicios diferentes y variados que combinan **POO**, **patrones de diseño**, y **estructuras de datos**, ¡nada de repetir! 😉

### ****1. Sistema de Gestión de Transporte Público****

**Contexto:** Diseña un sistema para gestionar las rutas de autobuses y pasajeros de una ciudad.

* **POO:** Clases como Autobus, Ruta, Pasajero.
* **Patrón Observer:** Notifica a los pasajeros registrados si hay cambios en los horarios o rutas.
* **Estructuras de Datos:** Usa HashMap para asociar rutas con autobuses y pasajeros.

**Requerimientos:**

* Agregar rutas, autobuses y pasajeros.
* Actualizar las rutas y notificar a los pasajeros.
* Consultar la lista de autobuses disponibles para una ruta específica.

### ****2. Sistema de Gestión de Hospital****

**Contexto:** Diseña un sistema para gestionar pacientes, doctores y consultas en un hospital.

* **POO:** Clases como Paciente, Doctor, Consulta.
* **Patrón Strategy:** Calcula el costo de una consulta dependiendo de la especialidad del doctor.
* **Estructuras de Datos:** Usa PriorityQueue para priorizar las consultas según su urgencia.

**Requerimientos:**

* Registrar pacientes y doctores.
* Programar consultas y priorizarlas según el nivel de urgencia.
* Calcular el costo total de las consultas según el tipo.

### ****3. Sistema de Inventario para una Tienda de Tecnología****

**Contexto:** Implementa un sistema para gestionar el inventario de dispositivos electrónicos.

* **POO:** Clases como Producto, Proveedor, Inventario.
* **Patrón Factory:** Crea dispositivos de diferentes tipos (teléfonos, laptops, accesorios).
* **Estructuras de Datos:** Usa una LinkedList para gestionar el historial de pedidos de productos.

**Requerimientos:**

* Registrar productos y proveedores.
* Realizar pedidos a los proveedores y actualizar el inventario.
* Consultar el historial de pedidos y su estado.

### ****4. Sistema de Reservas para Gimnasios****

**Contexto:** Diseña un sistema que permita a los clientes reservar clases y equipos en un gimnasio.

* **POO:** Clases como Cliente, ClaseGimnasio, Equipo.
* **Patrón Singleton:** Usa un Gimnasio que gestione una única instancia para todas las reservas.
* **Estructuras de Datos:** Usa HashSet para evitar que un cliente se registre dos veces en una misma clase.

**Requerimientos:**

* Registrar clientes y clases disponibles.
* Reservar equipos y verificar disponibilidad.
* Mostrar las clases programadas para cada cliente.

### ****5. Sistema de Gestión de Parques de Atracciones****

**Contexto:** Diseña un sistema para gestionar las atracciones y los visitantes de un parque temático.

* **POO:** Clases como Atraccion, Visitante, Ticket.
* **Patrón Observer:** Notifica a los visitantes si una atracción está fuera de servicio.
* **Estructuras de Datos:** Usa HashMap para asociar atracciones con los visitantes que las han reservado.

**Requerimientos:**

* Registrar atracciones y visitantes.
* Permitir reservar atracciones.
* Notificar automáticamente a los visitantes cuando una atracción no esté disponible.

### ****6. Sistema de Gestión de Ventas de Autos****

**Contexto:** Diseña un sistema para gestionar las ventas de autos en una concesionaria.

* **POO:** Clases como Auto, Vendedor, Cliente.
* **Patrón Factory:** Genera autos según su tipo (sedán, SUV, deportivo).
* **Estructuras de Datos:** Usa una ArrayList para gestionar el inventario de autos disponibles.

**Requerimientos:**

* Registrar autos y clientes.
* Vender un auto y actualizar el inventario.
* Consultar el historial de ventas realizadas.

### ****7. Sistema de Gestión de Eventos Culturales****

**Contexto:** Diseña un sistema para gestionar eventos culturales, sus asistentes y organizadores.

* **POO:** Clases como Evento, Asistente, Organizador.
* **Patrón Strategy:** Calcula el costo de entrada según el tipo de evento (conferencia, concierto, teatro).
* **Estructuras de Datos:** Usa una TreeMap para ordenar los eventos por fecha.

**Requerimientos:**

* Registrar eventos y asistentes.
* Calcular el costo total para un asistente según los eventos a los que se haya inscrito.
* Mostrar la lista de eventos ordenados cronológicamente.

### ****8. Sistema de Gestión de Mensajería****

**Contexto:** Diseña un sistema para gestionar el envío y recepción de mensajes entre usuarios en una aplicación.

* **POO:** Clases como Usuario, Mensaje, Conversacion.
* **Patrón Singleton:** Usa una clase ServidorMensajeria para gestionar todas las conversaciones.
* **Estructuras de Datos:** Usa una Queue para manejar la lista de mensajes pendientes de enviar.

**Requerimientos:**

* Registrar usuarios y sus conversaciones.
* Enviar y recibir mensajes entre usuarios.
* Mostrar las conversaciones de un usuario.

### ****9. Sistema de Alquiler de Herramientas****

**Contexto:** Diseña un sistema para gestionar el alquiler de herramientas en una tienda.

* **POO:** Clases como Herramienta, Cliente, Alquiler.
* **Patrón Observer:** Notifica a los clientes cuando una herramienta reservada esté disponible.
* **Estructuras de Datos:** Usa un HashMap para asociar herramientas con sus reservas.

**Requerimientos:**

* Registrar herramientas disponibles para alquiler.
* Registrar reservas de clientes.
* Notificar automáticamente cuando una herramienta vuelva a estar disponible.

### ****10. Sistema de Gestión de Residuos Urbanos****

**Contexto:** Diseña un sistema para gestionar la recolección de residuos en una ciudad.

* **POO:** Clases como Zona, Camion, Recoleccion.
* **Patrón Strategy:** Calcula el tiempo estimado de recolección dependiendo del tipo de residuos (orgánicos, reciclables, peligrosos).
* **Estructuras de Datos:** Usa una HashMap para asociar zonas con sus camiones asignados.

**Requerimientos:**

* Registrar zonas y camiones.
* Programar las recolecciones para cada zona.
* Calcular el tiempo estimado para completar la recolección.

EJERCICIOS MODULO 5

¡Entendido, Luis! Aquí tienes ejercicios con una narrativa clara de la problemática y los requerimientos, completamente nuevos y sin repetición:

### ****1. Sistema de Gestión de Colegios****

**Narrativa:** Una institución educativa necesita un sistema para gestionar el proceso de inscripción de estudiantes, el registro de notas y el cálculo del promedio por clase. Además, deben poder generar reportes que incluyan la clasificación de los estudiantes según su rendimiento.

**Requerimientos:**

* Permitir la inscripción de nuevos estudiantes en diferentes clases.
* Registrar notas para los estudiantes en cada clase.
* Calcular el promedio de notas por clase y el promedio general de cada estudiante.
* Generar reportes de clasificación según el rendimiento académico de los estudiantes.

### ****2. Sistema de Gestión de Cines****

**Narrativa:** Un cine busca modernizar el proceso de reservas para películas. Los clientes deberían poder consultar funciones disponibles, reservar asientos y recibir notificaciones si una función se cancela o se modifica.

**Requerimientos:**

* Registrar las funciones disponibles, incluyendo fecha, hora y sala asignada.
* Permitir a los clientes reservar asientos para una función específica.
* Mostrar la disponibilidad de asientos por función.
* Notificar a los clientes cuando se cancele o modifique una función.

### ****3. Sistema de Gestión de Tiendas de Alimentos****

**Narrativa:** Una cadena de supermercados necesita automatizar el proceso de administración de productos, promociones y ventas. Desean implementar un sistema que permita a los clientes crear listas de compras y obtener descuentos aplicables.

**Requerimientos:**

* Permitir registrar productos en el sistema, incluyendo precios y cantidades disponibles.
* Crear listas de compras para los clientes.
* Aplicar descuentos dependiendo de promociones activas.
* Generar una factura detallada al confirmar la compra, indicando el ahorro obtenido por las promociones.

### ****4. Sistema de Reservas de Consultorios Médicos****

**Narrativa:** Una clínica necesita un sistema para gestionar la reserva de citas médicas, médicos disponibles y pacientes que deseen consultar. El sistema debe manejar horarios y evitar que haya conflictos en las reservas.

**Requerimientos:**

* Registrar médicos disponibles y sus horarios de consulta.
* Permitir a los pacientes reservar citas médicas según los horarios disponibles.
* Mostrar las citas programadas para un día específico.
* Notificar automáticamente a los pacientes si el médico cancela o reprograma una cita.

### ****5. Sistema de Gestión de Talleres de Reparación****

**Narrativa:** Un taller mecánico necesita un sistema para gestionar el ingreso y reparación de vehículos. Los clientes deberían poder realizar solicitudes de servicios y obtener presupuestos estimados.

**Requerimientos:**

* Permitir el registro de vehículos ingresados al taller, incluyendo detalles del problema.
* Gestionar las solicitudes de servicios y asignarlas a mecánicos disponibles.
* Generar presupuestos según el tipo de servicio solicitado.
* Notificar a los clientes cuando el vehículo esté listo para ser retirado.

### ****6. Sistema de Administración de Huertos Urbanos****

**Narrativa:** Una comunidad que gestiona huertos urbanos necesita un sistema para realizar un seguimiento de las plantas cultivadas, asignar parcelas y registrar el mantenimiento realizado por los usuarios.

**Requerimientos:**

* Registrar las especies de plantas cultivadas y sus necesidades específicas.
* Asignar parcelas a los usuarios para el cultivo.
* Registrar las actividades de mantenimiento realizadas (riego, poda, fertilización).
* Generar reportes sobre el estado de crecimiento y salud de las plantas.

### ****7. Sistema de Gestión de Servicios de Mensajería****

**Narrativa:** Una empresa de envíos necesita un sistema para gestionar paquetes, clientes y entregas, asegurando un seguimiento en tiempo real del estado del paquete.

**Requerimientos:**

* Registrar paquetes incluyendo peso, dimensiones y destino.
* Permitir asignar paquetes a mensajeros para su entrega.
* Actualizar el estado del paquete (en tránsito, entregado).
* Permitir a los clientes consultar el historial de seguimiento de sus paquetes.

### ****8. Sistema de Gestión de Albergues para Animales****

**Narrativa:** Un refugio para animales necesita un sistema para gestionar la información de los animales acogidos, las adopciones y los registros de vacunación.

**Requerimientos:**

* Registrar los animales acogidos, incluyendo especie, edad y estado de salud.
* Permitir registrar las adopciones realizadas por los clientes interesados.
* Registrar y consultar el historial de vacunaciones para cada animal.
* Generar reportes sobre el número de animales acogidos y adoptados.

### ****9. Sistema de Gestión de Eventos Deportivos****

**Narrativa:** Una organización deportiva necesita un sistema para registrar participantes en competencias, programar eventos y clasificar a los ganadores según sus puntuaciones.

**Requerimientos:**

* Registrar a los participantes y sus equipos en competencias.
* Programar eventos incluyendo fecha, hora y lugar.
* Registrar las puntuaciones obtenidas por los participantes.
* Generar un cuadro de clasificaciones según las puntuaciones.

### ****10. Sistema de Control de Estaciones Climáticas****

**Narrativa:** Una empresa que monitorea el clima necesita un sistema que registre datos de estaciones meteorológicas y genere reportes detallados sobre el clima en diferentes regiones.

**Requerimientos:**

* Registrar datos meteorológicos como temperatura, humedad y velocidad del viento.
* Permitir agregar nuevas estaciones climáticas y actualizar sus lecturas.
* Generar reportes diarios, semanales y mensuales sobre las condiciones climáticas por región.
* Alertar sobre eventos climáticos extremos (tormentas, olas de calor, etc.).