**Sistema de Registro y Administración de Empleados**

**Objetivo:**

Desarrollar un sistema que permita registrar y mostrar información de empleados, así como calcular las bonificaciones de los empleados según su departamento, todo esto utilizando el paradigma de la Programación Orientada a Objetos (POO) en Python.

**Descripción:**

Necesitamos una clase llamada "Empleado" que modele la información básica de un empleado de una empresa. Esta clase deberá tener los siguientes atributos:

nombre: Nombre completo del empleado.

ID: Número identificador del empleado.

salario: Salario base del empleado.

departamento: Departamento al que pertenece el empleado.

Además, la clase "Empleado" debe tener un método calcular\_bonificacion que, cuando se llame, calcule y muestre la bonificación que le corresponde según su departamento.

**El programa deberá hacer lo siguiente en un bucle:**

Solicitar al usuario que ingrese la información requerida para cada atributo del empleado.

Instanciar un objeto de la clase Empleado con la información proporcionada.

Solicitar al usuario una acción a realizar. Si el usuario ingresa la palabra "bonificacion", el programa debe llamar al método calcular\_bonificacion del empleado. El bucle debe continuar hasta que el usuario decida salir.

Requerimientos:

**Utilizar el paradigma de la Programación Orientada a Objetos.**

Implementar el método \_\_init\_\_ para inicializar los atributos de la clase Empleado.

Implementar el método calcular\_bonificacion dentro de la clase.

Solicitar al usuario si desea calcular la bonificación o salir.

**Sistema de Reservas en un Restaurante**

Objetivo:

Desarrollar un sistema que permita registrar y mostrar información de las reservas, así como asignar mesas según el número de comensales, utilizando el paradigma de la Programación Orientada a Objetos (POO) en Python.

**Descripción:**

Se necesita una clase llamada "Reserva" que modele la información de las reservas en un restaurante. Esta clase deberá tener los siguientes atributos:

nombre: Nombre del titular de la reserva.

telefono: Número de teléfono de contacto.

num\_comensales: Número de personas en la reserva.

mesa: Número de mesa asignada.

La clase "Reserva" debe tener un método asignar\_mesa que, al ser llamado, asigne un número de mesa según el número de comensales.

**El programa deberá:**

Solicitar al usuario que ingrese la información requerida para cada atributo de la reserva.

Instanciar un objeto de la clase Reserva con la información proporcionada.

Solicitar al usuario una acción a realizar. Si el usuario ingresa la palabra "mesa", el programa debe llamar al método asignar\_mesa de la reserva. El bucle debe continuar hasta que el usuario decida salir.

**Requerimientos:**

Utilizar el paradigma de la Programación Orientada a Objetos.

Implementar el método \_\_init\_\_ para inicializar los atributos de la clase Reserva.

Implementar el método asignar\_mesa dentro de la clase.

Solicitar al usuario si desea asignar una mesa o salir.

**Sistema de Administración de Biblioteca**

Objetivo:

Desarrollar un sistema que permita registrar y mostrar los libros, así como prestar y devolver libros, utilizando el paradigma de la Programación Orientada a Objetos (POO) en Python.

Descripción:

Se necesita una clase llamada "Libro" que modele la información de un libro en la biblioteca. Esta clase deberá tener los siguientes atributos:

titulo: Título del libro.

autor: Autor del libro.

ISBN: Número identificador del libro.

estado: Estado del libro (disponible, prestado).

La clase "Libro" debe tener un método prestar que cambie el estado del libro a "prestado" y un método devolver que cambie el estado a "disponible".

El programa deberá:

Solicitar al usuario que ingrese la información requerida para cada atributo del libro.

Instanciar un objeto de la clase Libro con la información proporcionada.

Solicitar al usuario una acción a realizar. Si el usuario ingresa la palabra "prestar", se llama al método prestar, y si ingresa "devolver", se llama al método devolver. El bucle debe continuar hasta que el usuario decida salir.

Requerimientos:

Utilizar el paradigma de la Programación Orientada a Objetos.

Implementar el método \_\_init\_\_ para inicializar los atributos de la clase Libro.

Implementar los métodos prestar y devolver dentro de la clase.

Solicitar al usuario si desea prestar, devolver un libro o salir.

**Sistema de Administración de Biblioteca**

Objetivo:

Desarrollar un sistema que permita registrar y mostrar los libros, así como prestar y devolver libros, utilizando el paradigma de la Programación Orientada a Objetos (POO) en Python.

Descripción:

Se necesita una clase llamada "Libro" que modele la información de un libro en la biblioteca. Esta clase deberá tener los siguientes atributos:

titulo: Título del libro.

autor: Autor del libro.

ISBN: Número identificador del libro.

estado: Estado del libro (disponible, prestado).

La clase "Libro" debe tener un método prestar que cambie el estado del libro a "prestado" y un método devolver que cambie el estado a "disponible".

El programa deberá:

Solicitar al usuario que ingrese la información requerida para cada atributo del libro.

Instanciar un objeto de la clase Libro con la información proporcionada.

Solicitar al usuario una acción a realizar. Si el usuario ingresa la palabra "prestar", se llama al método prestar, y si ingresa "devolver", se llama al método devolver. El bucle debe continuar hasta que el usuario decida salir.

Requerimientos:

Utilizar el paradigma de la Programación Orientada a Objetos.

Implementar el método \_\_init\_\_ para inicializar los atributos de la clase Libro.

Implementar los métodos prestar y devolver dentro de la clase.

Solicitar al usuario si desea prestar, devolver un libro o salir.

**Sistema de Control de Estacionamiento**

Objetivo:

Desarrollar un sistema que permita registrar y mostrar información de vehículos en un estacionamiento, así como asignar y liberar espacios, usando POO en Python.

Descripción:

Se requiere una clase llamada "Vehiculo" que modele la información de un vehículo en el estacionamiento. Esta clase tendrá atributos como:

placa: Placa del vehículo.

modelo: Modelo del vehículo.

color: Color del vehículo.

espacio: Espacio asignado en el estacionamiento.

La clase "Vehiculo" debe tener un método asignar\_espacio que asigne un espacio y un método liberar\_espacio que libere el espacio asignado.

El programa deberá:

Solicitar al usuario que ingrese la información requerida para cada atributo del vehículo.

Instanciar un objeto de la clase Vehiculo con la información proporcionada.

Solicitar al usuario una acción a realizar. Si el usuario ingresa "asignar", el programa llama al método asignar\_espacio, y si ingresa "liberar", llama al método liberar\_espacio. El bucle debe continuar hasta que el usuario decida salir.

Requerimientos:

Utilizar el paradigma de la Programación Orientada a Objetos.

Implementar el método \_\_init\_\_ para inicializar los atributos de la clase Vehiculo.

Implementar los métodos asignar\_espacio y liberar\_espacio.

Solicitar al usuario si desea asignar un espacio, liberar un espacio o salir.

**Sistema de Reservas para Vuelos**

Objetivo:

Desarrollar un sistema que permita a los usuarios reservar vuelos, seleccionar asientos y confirmar sus reservas, utilizando el paradigma de la Programación Orientada a Objetos en Python.

Descripción:

Se necesita una clase llamada "Reserva" que modele la información de un vuelo. Esta clase tendrá atributos como:

pasajero: Nombre del pasajero.

num\_vuelo: Número de vuelo.

destino: Destino del vuelo.

asiento: Asiento seleccionado.

La clase "Reserva" debe tener un método seleccionar\_asiento que permita al pasajero escoger un asiento y un método confirmar\_reserva que confirme la reserva realizada.

El programa deberá:

Solicitar al usuario que ingrese la información requerida para cada atributo de la reserva.

Instanciar un objeto de la clase Reserva con la información proporcionada.

Solicitar al usuario una acción a realizar. Si el usuario ingresa "seleccionar", el programa llama al método seleccionar\_asiento, y si ingresa "confirmar", llama al método confirmar\_reserva. El bucle debe continuar hasta que el usuario decida salir.

Requerimientos:

Utilizar el paradigma de la Programación Orientada a Objetos.

Implementar el método \_\_init\_\_ para inicializar los atributos de la clase Reserva.

Implementar los métodos seleccionar\_asiento y confirmar\_reserva.

Solicitar al usuario si desea seleccionar un asiento, confirmar su reserva o salir.

**Sistema de Gestion de Pacientes en una Clínica Dental**

Objetivo:

Desarrollar un sistema que permita registrar pacientes, asignarles citas y gestionar tratamientos dentales, usando el paradigma POO en Python.

Descripción:

Se necesita una clase llamada "Paciente" que modele la información de un paciente en la clínica dental. Esta clase tendrá atributos como:

nombre: Nombre del paciente.

historial\_medico: Número de historial médico.

tratamiento: Tratamiento dental asignado.

cita: Fecha y hora de la cita.

La clase "Paciente" debe tener un método asignar\_tratamiento que asigne un tratamiento dental al paciente y otro método reagendar\_cita para cambiar la fecha/hora de la cita.

El programa deberá:

Solicitar al usuario que ingrese la información requerida para cada atributo del paciente.

Instanciar un objeto de la clase Paciente con la información proporcionada.

Solicitar al usuario una acción a realizar. Si el usuario ingresa "tratamiento", el programa llama al método asignar\_tratamiento, y si ingresa "reagendar", llama al método reagendar\_cita. El bucle debe continuar hasta que el usuario decida salir.

Requerimientos:

Utilizar el paradigma de la Programación Orientada a Objetos.

Implementar el método \_\_init\_\_ para inicializar los atributos de la clase Paciente.

Implementar los métodos asignar\_tratamiento y reagendar\_cita.

Solicitar al usuario si desea asignar un tratamiento, reagendar una cita o salir

**Sistema de Control de Alumnos en una Escuela**

Objetivo:

Desarrollar un sistema que permita registrar alumnos, asignarles materias y calificaciones, utilizando el paradigma POO en Python.

Descripción:

Se necesita una clase llamada "Alumno" que modele la información de un alumno en la escuela. Esta clase tendrá atributos como:

nombre: Nombre del alumno.

matricula: Número de matrícula.

materias: Lista de materias inscritas.

calificaciones: Calificaciones en las materias.

La clase "Alumno" debe tener un método inscribir\_materia que agregue una materia a la lista del alumno y otro método asignar\_calificacion para asignar una calificación a una materia.

El programa deberá:

Solicitar al usuario que ingrese la información requerida para cada atributo del alumno.

Instanciar un objeto de la clase Alumno con la información proporcionada.

Solicitar al usuario una acción a realizar. Si el usuario ingresa "inscribir", el programa llama al método inscribir\_materia, y si ingresa "calificar", llama al método asignar\_calificacion. El bucle debe continuar hasta que el usuario decida salir.

Requerimientos:

Utilizar el paradigma de la Programación Orientada a Objetos.

Implementar el método \_\_init\_\_ para inicializar los atributos de la clase Alumno.

Implementar los métodos inscribir\_materia y asignar\_calificacion.

Solicitar al usuario si desea inscribir una materia, asignar una calificación o salir.

**Sistema de Reservas de Habitaciones en un Hotel**

Objetivo:

Desarrollar un sistema que permita a los usuarios reservar habitaciones, elegir fechas y realizar pagos, usando POO en Python.

Descripción:

Se necesita una clase llamada "Reserva" que modele la información de un cliente en el hotel. Esta clase tendrá atributos como:

nombre: Nombre del cliente.

ID: Número identificador del cliente.

habitacion: Tipo de habitación reservada.

fechas: Fechas de reserva.

La clase "Reserva" debe tener un método elegir\_habitacion que permita seleccionar el tipo de habitación y otro método realizar\_pago para efectuar el pago.

Requerimientos:

Usar POO.

Método \_\_init\_\_.

Métodos elegir\_habitacion y realizar\_pago.

Solicitar acciones al usuario.

Sistema de Control de Equipos en un Gimnasio

Objetivo:

Desarrollar un sistema que permita registrar equipos, asignarles estado y registrar su mantenimiento, usando POO en Python.

Descripción:

Se necesita una clase "Equipo" con atributos:

nombre: Nombre del equipo.

ID: Identificación del equipo.

estado: Estado actual del equipo.

mantenimientos: Fechas de mantenimientos realizados.

La clase "Equipo" debe tener un método cambiar\_estado y otro método agregar\_mantenimiento.

Requerimientos:

Usar POO.

Método \_\_init\_\_.

Métodos cambiar\_estado y agregar\_mantenimiento.

Solicitar acciones al usuario.

Por supuesto, aquí te dejo 8 ejemplos adicionales:

Sistema de Reservas de Habitaciones en un Hotel

Objetivo:

Desarrollar un sistema que permita a los usuarios reservar habitaciones, elegir fechas y realizar pagos, usando POO en Python.

Descripción:

Se necesita una clase llamada "Reserva" que modele la información de un cliente en el hotel. Esta clase tendrá atributos como:

nombre: Nombre del cliente.

ID: Número identificador del cliente.

habitacion: Tipo de habitación reservada.

fechas: Fechas de reserva.

La clase "Reserva" debe tener un método elegir\_habitacion que permita seleccionar el tipo de habitación y otro método realizar\_pago para efectuar el pago.

Requerimientos:

Usar POO.

Método \_\_init\_\_.

Métodos elegir\_habitacion y realizar\_pago.

Solicitar acciones al usuario.

Sistema de Control de Equipos en un Gimnasio

Objetivo:

Desarrollar un sistema que permita registrar equipos, asignarles estado y registrar su mantenimiento, usando POO en Python.

Descripción:

Se necesita una clase "Equipo" con atributos:

nombre: Nombre del equipo.

ID: Identificación del equipo.

estado: Estado actual del equipo.

mantenimientos: Fechas de mantenimientos realizados.

La clase "Equipo" debe tener un método cambiar\_estado y otro método agregar\_mantenimiento.

Requerimientos:

Usar POO.

Método \_\_init\_\_.

Métodos cambiar\_estado y agregar\_mantenimiento.

Solicitar acciones al usuario.

Sistema de Control de Citas en una Veterinaria

Objetivo:

Desarrollar un sistema que permita registrar mascotas, asignarles citas y registrar atenciones médicas, usando POO en Python.

Descripción:

Se necesita una clase "Mascota" con atributos:

nombre: Nombre de la mascota.

tipo: Tipo de animal.

citas: Fechas de citas agendadas.

atenciones: Registro de atenciones médicas.

La clase "Mascota" debe tener un método agendar\_cita y otro método agregar\_atencion.

Requerimientos:

Usar POO.

Método \_\_init\_\_.

Métodos agendar\_cita y agregar\_atencion.

Solicitar acciones al usuario.

Sistema de Control de Stock en una Tienda de Ropa

Objetivo:

Desarrollar un sistema que permita registrar prendas, gestionar su stock y realizar ventas, usando POO en Python.

Descripción:

Se necesita una clase "Prenda" con atributos:

nombre: Nombre de la prenda.

codigo: Código de prenda.

tallas: Tallas disponibles.

stock: Stock actual.

La clase "Prenda" debe tener un método vender\_prenda y otro método reponer\_stock.

Requerimientos:

Usar POO.

Método \_\_init\_\_.

Métodos vender\_prenda y reponer\_stock.

Solicitar acciones al usuario.

Sistema de Control de Clientes en un Banco

Objetivo:

Desarrollar un sistema que permita registrar clientes, gestionar sus cuentas y realizar operaciones bancarias, usando POO en Python.

Descripción:

Se necesita una clase "Cliente" con atributos:

nombre: Nombre del cliente.

cuenta: Número de cuenta.

saldo: Saldo actual.

transacciones: Registro de transacciones.

La clase "Cliente" debe tener un método realizar\_retiro y otro método realizar\_deposito.

Requerimientos:

Usar POO.

Método \_\_init\_\_.

Métodos realizar\_retiro y realizar\_deposito.

Solicitar acciones al usuario.