



Unidades Didácticas 4, 5 y 6: Ejercicio de feedback 2

Informática 1

1. Objetivo

Este ejercicio tiene como finalidad aplicar los conocimientos adquiridos en programación en Python, enfatizando en la programación orientada a objetos y el manejo de datos. Se espera que al finalizar el proyecto, los estudiantes hayan reforzado sus habilidades técnicas y de trabajo en equipo, logrando desarrollar una aplicación funcional que resuelva una necesidad real en la gestión de una autoescuela.

2. Enunciado

Se le ha encomendado la tarea de desarrollar un programa en Python para gestionar una autoescuela. El objetivo es crear una aplicación que permita organizar y manejar la información de alumnos, profesores, clases, vehículos, permisos y facturación, sin utilizar frameworks complejos ni bases de datos externas; solo se utilizarán archivos y estructuras básicas de Python.

El ejercicio tiene como propósito que aplique sus conocimientos de programación orientada a objetos, manejo de archivos y estructuras de datos en Python para crear un sistema eficiente, fácil de usar y visualmente amigable, que permita llevar un seguimiento completo de cada alumno desde su registro hasta la obtención de su permiso de conducir, incluyendo clases impartidas, exámenes y cálculo de pagos.

Instrucciones:

1. Clase Registro

La clase Registro se encargará de almacenar y gestionar la información básica de los alumnos.

- **Atributos:**
 - num_registro: string (formato "Año/Numero").
 - nombre: string.
 - primer_apellido: string.
 - segundo_apellido: string.
 - dni: string.
 - fecha_nacimiento: string (formato "DD-MM-AAAA").
 - fecha_registro: string (formato "DD-MM-AAAA").
 - permiso_opta: string (ej. "B", "A").
- **Métodos:**
 - `__init__`: Para inicializar los atributos.
 - `mostrar_datos()`: Para mostrar la información del alumno en formato legible.

2. Clase Alumno

La clase Alumno hereda de la clase Registro y contiene información adicional relacionada con el alumno.

- **Atributos:**
 - domicilio: string.
 - municipio: string.
 - provincia: string.
 - telefono1: string.
 - telefono2: string (opcional).
 - correo: string.
 - num_clases: int.
 - profesor: string.
 - examenes_teoricos: list (fecha y hora de los exámenes teóricos).
 - examenes_circulacion: list (fecha y hora de los exámenes de circulación).
 - total_anticipos: float.
- **Métodos:**
 - `__init__`: Inicializa con los atributos del alumno.
 - `agregar_examen_teorico()`: Permite añadir nuevas fechas para los exámenes teóricos.
 - `agregar_examen_circulacion()`: Permite añadir nuevas fechas para los exámenes de circulación.
 - `agregar_clases(num)`: Añade el número de clases realizadas.
 - `mostrar_info_completa()`: Muestra toda la información completa del alumno.

3. Clase Profesor

La clase Profesor se encarga de almacenar la información de los profesores y del vehículo que utilizan.

- **Atributos:**
 - nombre: string.
 - primer_apellido: string.
 - segundo_apellido: string.
 - vehiculo: string (matrícula).
 - tipo_vehiculo: string (tipo de vehículo).

- itv: string (fecha de caducidad de la ITV).
- gastos_combustible: dict (clave: fecha, valor: coste del gasto).
- **Métodos:**
 - `__init__`: Inicializa los atributos.
 - `registrar_gasto_combustible(fecha, costo)`: Agrega un gasto de combustible al historial.
 - `mostrar_gastos_combustible()`: Muestra los gastos de combustible acumulados.
 - `mostrar_info_profesor()`: Muestra la información del profesor y el vehículo.

4. Clase Clase

La clase Clase gestionará las sesiones entre el alumno y el profesor.

- **Atributos:**
 - alumno: string (DNI del alumno).
 - profesor: string (DNI del profesor).
 - matricula_vehiculo: string.
 - fecha_hora: string.
- **Métodos:**
 - `__init__`: Inicializa la clase con los datos requeridos.
 - `mostrar_info_clase()`: Muestra la información de la clase realizada.

5. Clase Permiso

La clase Permiso gestionará la información de los tipos de licencias disponibles en la autoescuela.

- **Atributos:**
 - tipo_permiso: string (ej. "B", "A").
 - precio_matricula: float.
 - clases_incluidas: int.
 - precio_por_clase: float.
 - precio_examen: float.
 - precio_renovacion: float.
- **Métodos:**

- `__init__`: Inicializa los atributos del permiso.
- `mostrar_info_permiso()`: Muestra la información completa del permiso.

6. Clase Factura

La clase Factura gestionará la generación de facturas para cada alumno.

- **Atributos:**

- `alumno`: string (DNI del alumno).
- `precio_matricula`: float.
- `num_clases_incluidas`: int.
- `num_clases_dadas`: int.
- `precio_clase`: float.
- `num_exámenes`: int.
- `precio_examen`: float.
- `num_renovaciones`: int.
- `precio_renovacion`: float.
- `anticipos`: float.

- **Métodos:**

- `__init__`: Inicializa los atributos.
- `calcular_total()`: Calcula el total de la factura (matrícula + clases + exámenes + renovaciones).
- `calcular_iva()`: Calcula el 21% de IVA.
- `calcular_total_con_iva()`: Calcula el total con IVA incluido.
- `calcular_saldo_pendiente()`: Resta los anticipos del total con IVA.
- `generar_factura()`: Muestra la factura completa, detallando todos los conceptos.

7. Clase Anticipo

La clase Anticipo almacenará y gestionará los pagos anticipados de los alumnos.

- **Atributos:**

- `alumno`: string (DNI del alumno).
- `fecha`: string.

- concepto: string.
- cantidad: float.
- **Métodos:**
 - `__init__`: Inicializa los atributos.
 - `mostrar_info_antipico()`: Muestra la información completa del anticipo realizado.

Funcionalidades Adicionales:

1. **Interfaz en Consola:**
 - Cree un menú interactivo para que el usuario pueda seleccionar entre registrar alumnos, profesores, clases, anticipos o generar facturas.
 - Use bucles y funciones para organizar el flujo del programa.
2. **Persistencia de Datos:**
 - Utilice archivos **JSON** o **CSV** para almacenar la información de los alumnos, profesores, clases, permisos y facturas, de modo que los datos se puedan cargar al iniciar el programa y se guarden al finalizar.
3. **Manejo de Errores:**
 - Incluya manejo de excepciones para evitar errores en la entrada de datos (ej. fechas mal formateadas, valores incorrectos, etc.).
4. **Formato de Salida:**
 - Asegúrese de que las facturas y otros informes se muestren de manera clara en la consola y se puedan guardar en un archivo **.txt** o **.pdf**.

3. Instrucciones de entrega

- **Extensión:** Un único archivo .zip donde quedará exportado el Proyecto.
- **Nombre del fichero:** EjercicioFeedback2_NPdelAlumno
- **Nombre de los documentos:** EjercicioFeedback2_NPdelAlumno
- **Formato de entrega:** .zip

WELCOME
TO
UAX

UAX

Universidad
Alfonso X el Sabio

GRACIAS

UAX.COM