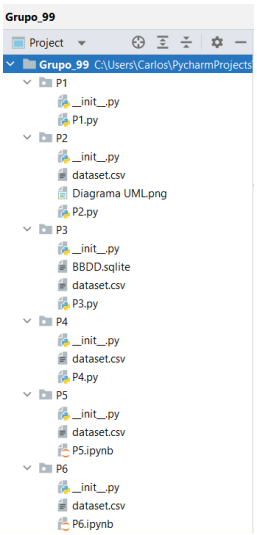
Trabajo de Python

Instrucciones para la entrega



- Se debe entregar un proyecto de PyCharm cuyo nombre sea la numeración del grupo.
- El proyecto debe contener seis paquetes denominados P1, P2, P3, P4, P5 y P6; uno por problema planteado en este enunciado.
- En cada paquete se añadirán los ficheros de código .py y .ipynb necesarios para resolver los problemas planteados.
- Se incluirán en cada paquete los ficheros utilizados/solicitados: imágenes PNG, ficheros de datos CSV, ficheros SQLite.
- La estructura del proyecto debe ser similar a la de la imagen adjunta.
- **IMPORTANTE:** El código de los programas **debe incluir comentarios** que describan lo realizado en cada ejercicio, así como detalles de la solución.

Enunciado general

Se trata de resolver en Python los seis problemas que se plantean y para ello a cada grupo se le ha asignado un dataset en formato CSV en el aula virtual. Tomando como inspiración dicho conjunto de datos debe resolver los problemas planteados en este trabajo.

Los dataset se han seleccionado de la web kaggle.com y se facilita el enlace que permite consultar la información que allí hay publicada sobre los datos que contienen dichos dataset.

Problema 1

Escriba un programa que permita gestionar el almacenamiento de los registros del dataset en un diccionario de Python.

El programa presentará un menú con las distintas opciones disponibles. A continuación, el usuario seleccionará una de las opciones, se realizará la entrada de datos correspondiente a dicha opción, el procesamiento de datos adecuado, una salida por pantalla que informe del resultado de la operación y finalmente volverá a mostrar el menú de opciones.

Los registros se almacenarán en un diccionario donde la clave debe ser el campo o campos (clave en forma de tupla) que se utilicen como clave en el dataset. El valor asociado será una tupla con todos los valores del registro, incluida la clave.

```
diccionario = {
        clave1 : (campo11, campo12, campo13),
        clave2 : (campo21,campo22,campo23),

}
```

La entrada y salida de datos debe ser adecuada a los nombres y tipos de datos de los campos del dataset correspondiente.

Las opciones de dicho programa deben ser:

1. Agregar un nuevo registro
2. Buscar un registro por su clave y mostrar sus valores
3. Buscar un registro por su clave, editarlo y mostrar sus valores.
4. Borrar un registro a partir de su clave
5. Listar todos los registros en formato de tabla
6. Salir

Problema 2

Tomando como base el dataset proporcionado debe implementar las clases necesarias para reflejar dicho conjunto de datos. Adjuntar al proyecto un diagrama UML de las clases diseñadas en formato imagen (en draw.io se puede exportar a imagen PNG).

Implemente también una clase denominada Almacén que hará el papel de almacén de datos. El almacenamiento de los datos se implementará mediante **una lista** de objetos de la clase que represente a los registros del dataset.

La clase Almacén deberá tener los siguientes métodos, teniendo en cuenta que los nombres de los métodos se deben adaptar a la temática del conjunto de datos (altaPelicula, bajaVehículo, listadoFlores, agruparPorPaís).

1. **__init__**: Deberá inicializar la lista de objetos.
2. **altaObjeto(objeto)** : Deberá insertar un objeto en la lista, teniendo en cuenta que no se admiten duplicados. Objetos duplicados son aquellos que tienen el mismo valor en el campo que se tome como clave. Debe devolver la cadena “OK” si se ha insertado o “DUPLICADO” si no se ha insertado por este motivo.
3. **bajaObjeto(clave)** : Deberá borrar un objeto de la lista a partir de su clave. Debe devolver la cadena “OK” si se ha localizado y eliminado el objeto, o “NO LOCALIZADO” si no se ha localizado.
4. **listadoObjetos** : Deberá obtener un listado en formato de tabla de todos los datos almacenados. Utilizar caracteres de tabulación para separar las columnas de datos y caracteres de salto de línea para separar los datos de cada objeto. El listado obtenido tendrá una fila de encabezamiento con los nombres de los campos.
5. **agruparPorCampo( )** : Método que procesará los datos almacenados y realizará la agrupación por uno de los campos del dataset (por ejemplo: país, código postal, marca, …). Devolverá un listado con dos columnas: la primera corresponderá al campo utilizado para la agrupación y la segunda a otro campo seleccionado en el que se utilizará una función de agregación (por ejemplo: suma, producto, valor medio, …)
6. **fromCSV(ruta)**: Método que recibe la ruta y el nombre de un fichero CSV que debe contener los datos de los objetos que se almacenan. Se leerá cada registro del fichero creando el objeto correspondiente e insertándolo en la lista sin duplicados.
7. **toCSV(ruta)**: Método que recibe la ruta y el nombre de un fichero que se escribirá en formato CSV con el contenido de todos los objetos del almacén.

Para dar por válido el problema se debe acompañar un pequeño programa principal que pruebe todos los métodos de la clase Almacén.

Problema 3

Diseñe una base de datos SQLite donde exista una tabla con la estructura del dataset, coincidiendo el nombre de los campos y con el tipo de datos adecuado en cada caso.

A continuación, crear un programa que realice las siguientes operaciones secuencialmente:

1. Utilizar las sentencias adecuadas para crear la base de datos SQLite con la tabla diseñada.

2. Leer el dataset en formato CSV e insertar todos los registros en la tabla.

3. Realizar una sentencia "SELECT" con una cláusula "WHERE" compleja que seleccione varias filas (entre 5 y 20) y muestre el resultado por pantalla.

4. Realizar una sentencie "UPDATE" para actualizar el valor de un campo con una cláusula "WHERE" que seleccione varias filas (entre 5 y 20).

5. Realizar una sentencia "SELECT" que calcule realice el agrupamiento de datos por una columna y realice una operación de agregación (suma, media …) sobre otra columna.

6. Realizar una sentencia "DELETE" con una cláusula "WHERE" que seleccione varias filas (entre 5 y 20) para su eliminación.

Se valorará positivamente la complejidad y utilidad de las sentencias SQL diseñadas.

Problema 4

Para este problema utilizar como estructura de datos un diccionario como el del problema 1 que contenga al menos cinco pares (clave, valor).

```
diccionario = {
       clave1 : (campo11, campo12, campo13),
       clave2 : (campo21,campo22,campo23),

}
```

Se trata de diseñar dos funciones que utilicen funciones de orden superior (filter, map, reduce), funciones lambda y comprensión de listas, para procesar el diccionario y obtener un resultado que debéis definir.

Cualquier proceso que utilice bucles no se considerará válida. Se pueden utilizar resultados auxiliares o intermedios.

Se valorará positivamente la complejidad de las funciones diseñadas.

Problema 5

Utilice para este ejercicio el conjunto de datos en formato CSV del dataset asignado.

Escriba un notebook que haga las siguientes operaciones con el anterior conjunto de datos:

1. Cargue en un dataframe de pandas todos los datos.

2. Realice una operación donde se seleccionen varias columnas relevantes, pero mostrando solo algunas de las filas, es decir realizando un filtrado en los valores de al menos dos columnas que no tienen por qué coincidir con las que se muestren.

3. Construya y muestre un nuevo dataframe que agregue al original una nueva columna calculada numérica a partir de una o varias de las ya existentes. Posteriormente agrupar los datos por algún campo relevante y mostrar el resultado aplicando algún criterio de ordenación.

Problema 6

Utilice para este ejercicio el conjunto de datos en formato CSV del dataset asignado.

Escriba un notebook que haga las siguientes operaciones con el anterior conjunto de datos:

1. Genere una gráfica con la librería Seaborn https://seaborn.pydata.org/ y escoger el tipo más adecuado (barras, histograma, puntos, sectores) que sea útil para el análisis de los datos del dataset. Incluir comentarios que ayuden a interpretar la gráfica generada.

2. Guardar en un fichero “grafica.png” la gráfica generada.