Segundo Parcial
14/06/2022

Tener en cuenta que:

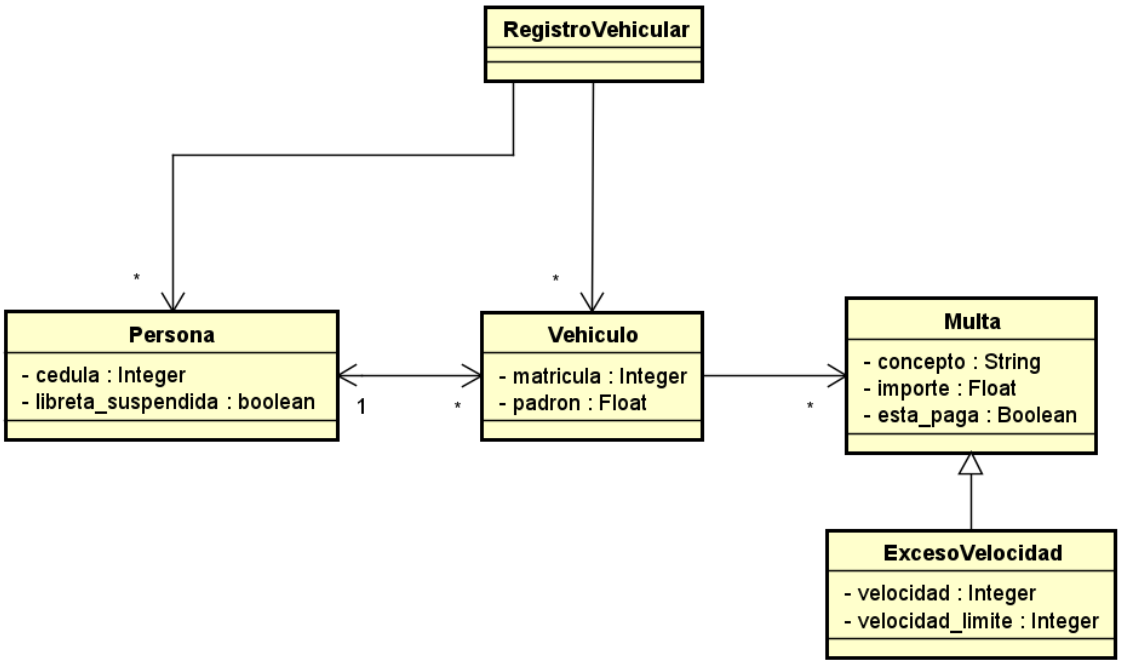
- La duración del parcial es de 3 horas.
- Se debe realizar en computadora, para este fin se deberá seguir los siguientes pasos:
    1. Crear una carpeta dentro de su computadora de nombre segundoparcial2022.
    2. Dentro de esta carpeta se deben crear todos los módulos necesarios para la resolución del problema planteado.
- Culminado las horas del parcial se deberá entregar a través del Moodle en la tarea "Entrega Segundo Parcial". Se espera que se suba un Zip de la carpeta "segundoparcial2022" creada en el paso 1. Puede usar de nombre del zip segundoparcial2022.zip.
- Tenga en cuenta que no se corrige código con errores del intérprete de Python.
- La prueba es individual, para la parte teórica es sin material, para la parte práctica se puede usar material.
- Se contestan dudas solo en la primer hora y media del parcial.

Ejercicio 1

Se desea desarrollar un sistema que permita a las intendencias informatizar el registro vehicular. Dentro de esta realidad se identifican las personas que son titulares de vehículos, los vehículos y por último las multas que los vehículos pueden recibir por infracciones cometidas. Dentro de las multas se desea mantener un registro de si fue paga o no y para el caso de multas de exceso de velocidad se desea registrar la velocidad del vehículo en el momento de la infracción como la velocidad limite establecida.

En caso de que una persona disponga mas de 3 multas de exceso de velocidad en cualquier de sus vehículos, se debe suspender la licencia de conducir y no se debe permitir registrar nuevos vehículos.

Se propone el siguiente diagrama de clases parcial que modela la realidad a implementar:



Con esta realidad en mente se debe implementar la clase RegistroVehicular que tenga las siguientes operaciones:

- **def registrar_vehiculo(self, matricula, nro_padron, cedula_titular)**
  - Se debe registrar el vehículo identificado por su matrícula asociado a la persona identificada por su cédula. En caso de que la persona titular no exista debe crearse en esta operación. Si el vehículo este asociado a otro titular debe retornar un error. Si la persona tiene la libreta suspendida (supero las 3 multas en otros vehículos, calculado en la siguiente operación) debe retornar un error.
  - Lanza la excepción InformacionInvalida si algún argumento obligatorio es vacío o si la persona tiene la libreta suspendida.
  - Lanza la excepción EntidadYaExiste en caso de que el vehículo es registrado a otro titular.
- **def registrar_multa_vehiculo(self, matricula, concepto, importe, es_exceso_velocidad, velocidad_vehiculo, velocidad_limite)**
  - Registra una multa al vehículo identificado por su matrícula. El campo es_exceso_velocidad es un boolean que indica si la infracción fue por exceso de velocidad, en caso de ser True se reciben los argumentos velocidad_vehiculo y velocidad_limite.
  - En caso de super las 3 multas por exceso de velocidad en cualquiera de sus vehículos se debe suspender la libreta del titular.
  - Lanza la excepción EntidadNoExiste, en caso de que no se encuentre el vehículo con la matricula indicada.
  - Lanza la excepción InformaciónInvalida si algún argumento obligatorio es vacío.
- **def pagar_multa_vehiculo(self, matricula, concepto, importe)**
  - Se registra el pago de una multa para el vehículo identificado por la matricula, de cierto concepto e importe. En caso de que exista más de una multa, con el mismo concepto e importe se debe pagar la primera.
  - Lanza la excepción EntidadNoExiste, en caso de que no se encuentre el vehículo o la multa.
- **def traspaso_vehiculo(self, cedula_titular, matricula, cedula_nuevo_titular)**
  - Esta operación traslada un vehículo con su historial de multas de un titular identificado con cedula_titular a otra persona identificada por cedula_nuevo_titular. No se debe permitir realizar la transferencia si el dueño actual tiene multas sin pagar del vehículo a hacer traspaso o si el nuevo titular tiene la libreta suspendida.
  - Lanza la excepción EntidadNoExiste en caso de que no se encuentre el vehículo o las personas involucradas.
  - Lanza la excepción InformaciónInvalida en caso de que la transacción no pueda realizarse por las precondiciones indicadas.

Consideraciones

- Todas las clases sin ser las excepciones y la clase RegistroVehicular se deben colocar en el paquete entities. La clase RegistroVehicular se debe colocar en la raíz del proyecto. Las excepciones se deben colocar en el paquete exceptions.
- Se debe preservar el encapsulamiento de todas las entidades, para las cuales se deben usar getters (todas las propiedades) y setters (si son realmente necesarios).
- Se debe crear un modulo por clase.
- Se debe crear el módulo registro_vehicular.py una operación “main” que pruebe el correcto funcionamiento de las operaciones implementadas. Se espera al menos una prueba de cada operación.