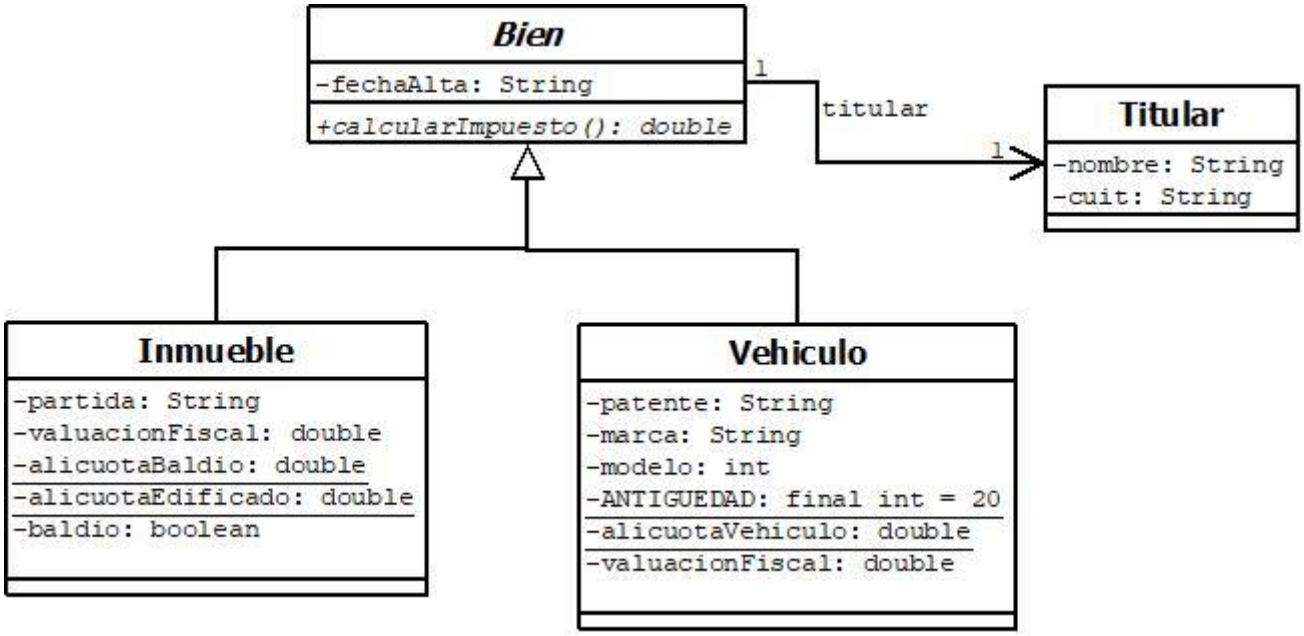
1° PARCIAL DE ALGORITMICA Y PROGRAMACION II - 2023

1) La Municipalidad necesita un sistema para gestionar el cobro de impuesto anual a los titulares de bienes inmuebles y patentes de vehículos. Para el cálculo del impuesto en los bienes inmuebles se toma su valuación fiscal y se lo multiplica por la alícuota correspondiente a baldío o edificado según el estado del mismo. Para los automotores se toma la valuación fiscal y se lo multiplica por la alícuota del vehículo. Si el modelo del vehículo tiene 20 años o más de antigüedad no paga impuesto.

Diagrama de Clases



Crear todas las clases con sus atributos y constructores, gets y sets, equals y toString. Implementar los métodos que correspondan.

Nota: puede utilizar **`LocalDate.now().getYear()`** para obtener el año actual.

2) Realizar un programa donde pruebe:

a) Cargar en un arreglo bienes inmuebles y vehículos para distintos titulares.

b) Cargar las siguientes alícuotas para los cálculos:

alicuotaBaldio = 0.006
alicuotaEdificado = 0.002
alicuotaVehiculo = 0.004

c) Recorrer el arreglo y mostrar el bien, el impuesto a cobrar y el total de los impuesto a cobrar de todos los bienes. Incluir por lo menos un bien inmueble edificado y uno baldío y un vehículo con un modelo menor a 20 años de antigüedad y otro con un modelo mayor a 20 años de antigüedad.

d) Recorrer el arreglo y mostrar el bien, el impuesto a cobrar y el total de los impuesto a cobrar de todos los bienes que pertenece a un titular determinado.

3) Dada una lista simplemente enlazada, agregar los siguientes métodos y realizar un programa que pruebe los diferentes casos que se pueden presentar.

a)

```
/**
 * Crea una nueva lista y copia todos sus elementos
 *
 * @return lista nueva con una copia de la lista actual
 *
 */
public SinglyLinkedList<E> duplicate()
```

b)

```
/**
 * Retorna la lista con todos los elementos de la lista l insertados a partir de
 * la posición pos.
 *
 * Considere la opción de utilizar el método duplicate() o clone()
 *
 * No utilizar métodos que insertan de a un elemento como por ejemplo  addPos(E, p)
 *
 * Por ejemplo:
 *
 * Dada la lista: {A, B, C, D} y la lista l = {X, Y}
 *
 * addPos(l, 2) => {A, B, X, Y, C, D}
 *
 * addPos(l, 0) => {X, Y, A, B, C, D}
 *
 * addPos(l, 4) => {A, B, C, D, X, y}
 *
 * @param SinglyLinkedList<E> l : lista a insertar
 * @param SinglyLinkedList<E> pos : posición a partir de donde se inserta
 *
 *
 * @return lista original más todos los elementos de la lista l insertados a
 *         partir de la posición pos
 *
 * @exception Si los índices están fuera de rango lanza la excepción
 *               IndexOutOfBoundsException
 *
 */
public void addList(SinglyLinkedList<E> l, int pos) throws IndexOutOfBoundsException
```

IMPORTANTE:

Subir al Campus solamente los archivos con extensión .java.