Ejercicio 19 : Alquiler de propiedades

Se desea diseñar e implementar una **plataforma para gestión de reservas** de **propiedades** que llamaremos OOBnB.
En la misma, los **usuarios** pueden gestionar sus **inmuebles** para su **alquiler** así como también realizar **reservas** sobre estos.
De los usuarios se conoce el **nombre**, la **dirección** y el **DNI**.
Cada usuario posee propiedades que desea alquilar, de las cuales se guarda la **dirección**, un **nombre descriptivo** y el **precio** que se desea cobrar por noche.
Además, los usuarios pueden realizar reservas sobre cualquiera de las propiedades disponibles.

Implementar la siguiente funcionalidad:

Consultar la disponibilidad de una propiedad en un período específico:
dada una propiedad, una fecha inicial y una fecha final, se debe determinar si la propiedad está disponible el período indicado.

Crear una reserva:
Un usuario puede realizar una reserva para un período de tiempo determinado.
Si la propiedad está disponible, se crea la reserva y la propiedad pasa a estar ocupada durante ese período.
Si no lo está, la reserva no será creada.

Calcular el precio de una reserva:
Dada una reserva, se debe poder calcular su precio.
El mismo se obtiene multiplicando la cantidad de noches por el precio por noche.

Cancelar una reserva:
Se debe permitir cancelar una reserva.
En este caso, la propiedad pasa a estar disponible durante el período de tiempo indicado en la reserva.
Esta operación sólo es permitida si el período de la reserva no está en curso.

Calcular los ingresos de un propietario:
Se debe calcular la retribución a un propietario, la cual es el 75% de la suma de precio totales de las reservas incluidas en un período específico de tiempo.

Sugerencia: Para el manejo de los períodos de reserva se sugiere
añadir un nuevo método a la interfaz **DateLapse** definida en el ejercicio anterior (**ejercicio de Intervalos de tiempo**).
A modo de sugerencia, la especificación del mismo puede ser la siguiente:
// Retorna true si el período de tiempo del receptor se superpone con el recibido por parámetro.
public boolean overlaps (DateLapse anotherDateLapse){}

Tareas:

1) Modelar Dominio:
a) Realice la lista de conceptos candidatos, teniendo en cuenta los métodos vistos en la teoría.
b) Grafique el modelo de dominio usando UML.
c) Actualice el modelo de dominio incorporando los atributos a los conceptos.
d) Agregue asociaciones entre conceptos, indicando para cada una de ellas la categoría a la que pertenece, de acuerdo a lo explicado en la teoría, y demás atributos, según sea necesario.

2) Modelado e Implementación:
a) Diagrama de clases UML a partir del modelo de dominio realizado en el punto anterior.
b) Implemente en Java la funcionalidad requerida.

3)Pruebas Automatizadas:
a) Diseñe los casos de prueba teniendo en cuenta los conceptos de valores de borde y particiones equivalentes vistos en la teoría para verificar la disponibilidad de una propiedad en una fecha, reservar y cancelar una reserva.
b) Implemente utilizando JUnit los tests automatizados diseñados en el punto anterior.