RESOLUCION DEL MARRON2: ObjectDB

1. Para el primer punto lo que hicimos fue añadir el booleano que se pide en el enunciado a la lógica de negocio
2. Para el segundo apartado hemos creado una clase llamada Flight_objectdbAccess, que será la que gestione todo lo de la base de datos, hemos creado los métodos de insertar vuelos y vuelos concretos, encontrar vuelos y vuelos concretos por su clave principal(código) y métodos para que devuelva una lista de todos los vuelos concretos agrupados por cada vuelo al que perteneces, esto será útil para el tercer apartado. Además de esto hemos creado funciones para eliminar vuelos concretos y por ultimo un método para actualizar la información del vuelo concreto, también útil para la 3ª parte, ya que necesitaremos actualizar la cantidad de asientos de los vuelos.
3. En la 3ª parte, se nos pedía que al comprar un billete de un asiento, este se actualizase. Para esto, hemos utilizado una función de la clase que gestiona la base de datos, que agrupa los vuelos concretos dependiendo de su vuelo principal. Después, hemos llamado este método en otro método de la lógica de negocio para que ahora en la interfaz la información de los vuelos concretos no saliese de la lista que se creaba en la constructora de la clase de la lógica de negocio, sino que ahora saliese directamente de la base de datos.

En la tercera parte hemos tenido un problema, y es que hemos intentado agrupar los vuelos concretos por su vuelo pero al hacer esto la JComboBox salía vacía, y sin agruparlo para cada vuelo salían todos los vuelos concretos.

Esto nos ha quitado bastante tiempo, y sumándole los problemas para entender como conectar la base de datos con la lógica y la interfaz, además de crear todos los nuevos métodos , hemos invertido unas 5 horas en el proyecto en total.