AGENCIA INMOBILIARIA – ANTEPROYECTO

1. ArrayList (estructuras de datos)

Crearemos un ArrayList que usaremos para mostrar la lista de viviendas disponibles y que también aprovecharemos para indicar que viviendas están en venta y cuales están alquiladas.

2. Enumeraciones

Utilizaremos las enumeraciones para elegir a que barrio de la ciudad pertenece cada vivienda. Una vivienda puede pertenecer a la zona del ***Brillante***, a ***Valdeolleros***, ***Fátima, Cañero, Ciudad Jardín,*** *a la* ***Judería*** *o al* ***Centro***.

3. Herencia

Existirán varios tipos de ***Viviendas***. Entre ellas encontramos ***Casas, Pisos*** *y* ***Casas Rurales***. Todas las viviendas contarán con atributos como, por ejemplo, **metrosCuadrados**, **numeroHabitaciones**, **numeroBaños**, **añosAntigüedad, precio** y **zona** a la que pertenecen. Se identificarán por un **codigoEdificio**, que es unívoco y autogenerado. Sin embargo, las casas tendrán atributos que nos indicarán, por ejemplo, si son **adosadas** o no y el **numeroDePlantas**. Por otra parte, para los pisos necesitaremos saber, por ejemplo, la **planta** en la que están situados y si tienen **ascensor** o no. Por último, las casas rurales son algo diferentes. Las casas rurales no pueden ser adosadas ni tienen ascensor, pero a diferencia de las casas y los pisos (que pueden venderse o alquilarse), las casas rurales **solo podrán ser alquiladas**.

4. Interfaces

Implementaremos la interfaz **calcularAlquiler** para conocer el precio al que ascendería el alquiler teniendo en cuenta las características de nuestra vivienda. También usaremos la interfaz **comparable** para mostrar las viviendas de forma ordenada por su código de vivienda, lo que las separará en Casas, Pisos y Casas Rurales.

5. Flujos de datos

Utilizaremos los flujos de datos para escribir y leer en nuestro fichero de almacenamiento. Además serán utilizados para la entrada y salida de datos.

6. Ficheros

Almacenaremos la lista de edificios en un fichero, el cual podremos guardar cuando realicemos cambios en dicha lista.

7. Excepciones

Si un edificio ya está registrado en la lista, aparecerá la excepción ***ViviendaExistenteException***. Si alguno de los datos de la vivienda, como por ejemplo, numeroHabitaciones, es negativo, saltará la excepción ***ValorInvalidoException***. Además de estas, si queremos añadir una vivienda a nuestra lista, y no ha sido creada, saltará la excepción **ViviendaNoExistenteException**. Por último, tenemos la excepción **CodigoNoValidoException**, que nos indica si nuestro código unívoco se corresponde con el patrón o no.

8. Expresiones regulares

Necesitaremos de expresiones regulares para validar el código único de la vivienda, que será representado por 4 números seguidos y una letra mayúscula. Si la vivienda se trata de una casa, la letra será C, y así con los demás tipos de viviendas.

9. Fechas

Utilizaremos las fechas para calcular la antigüedad de las viviendas.

10. GUI

Todo el programa se desarrollará mediante un entorno gráfico para que haya un cómodo funcionamiento y una visualización más atractiva. En cuanto al comportamiento, se mostrará cuando una vivienda es alquilada (Casa, Piso, CasaRural), lo cual se podrá ver recorriendo el array y se controlará a través de una variable booleana *alquilado*. También se indicará cuando una vivienda ha sido vendida (Casa, Piso) y se verá del mismo modo, al recorrer el array controlándose con una variable booleana *vendido*.