Proyecto: Pub

Segunda parte

En el Pub se sirven también distintos tipos bebidas que debes representar en una jerarquía de clases que partan de la clase **Drink**, teniendo en cuenta las especificaciones que se redactan a continuación. En algunas de estas clases será necesario incluir información (atributos) y operaciones (métodos) que permitan hacer una gestión adecuada de las bebidas.

Los tipos de bebidas que se sirven son **cervezas**, **vinos**, **agua**, **refrescos** (soda drinks) y **licores**. El agua y los refrescos incluyen su *sabor* y si tienen *burbujas* o no. Además, para los refrescos es preciso guardar el *porcentaje de zumo* que contienen, y para el agua, su *origen* y si es o no de *manantial*.

Las cervezas se caracterizan por dos valores de tipo doublé, su *IBU* (International Bittering Unit), y su *ABV* (Alcohol By Volume), y por su *origen*. Además, pueden ser **Lager** o **Ale**, en función del tipo de levadura utilizada en su proceso de fermentación. Las Ale son las más demandadas y deben guardar la *marca*. Las Lager, en cambio, incluyen su *color*. Entre las Lager, se distinguen las **Pilsener** y las **DarkLager**. Las de tipo Pilsener guardan la *temperatura* a la que deben servirse y las DarkLager si la cervecería las tiene en *botella* o en *barril*.

El **vino** recoge si tiene o no *denominación de origen* y es *espumoso* o no. Por último, los **licores** se caracterizan por su *graduación* y sus *años*.

En todas las clases se añadirán los métodos *get* y *set* necesarios para acceder/modificar su información privada. El método *toString* debe redefinirse en todas las clases, para ir añadiendo incrementalmente la nueva información privada añadida en cada clase (bastará con introducir dicha información separada por un espacio en blanco). Además, en las clases de cerveza se quiere que este método añada al String que devuelve, la vía hereditaria (*inheritancePath*), desde las clase a la que pertenece la instancia que ha invocado el método hasta la clase **Beer**.

Lógicamente, la cervecería necesita conocer el precio de las bebidas, valor que cambia con bastante frecuencia.

En este momento, las cervezas Lager tienen un precio base de 2€, pero las DarkLager incrementan ese precio en el 1% de su IBU si están en barril, añadiendo un euro más si están en botella. El precio de las Ale es de 3€ más 0,05 por cada ABU.

El precio del vino es 3€ si es espumoso y 2€ si no lo es.

Los licores se venden por copas y en general cuestan 3€ + 0,25 por año.

En el caso del agua y los refrescos el precio es 3,5 euros.

La cervecería ha puesto recientemente un servicio de entrega a domicilio. De momento, sólo reparte a domicilio comida, agua y refrescos. Para calcular el tiempo del envío de los pedidos, utiliza la distancia a recorrer (en kilómetros) y el tiempo empleado para ello, estimando que el tiempo de en un servicio será de 3 minutos por kilómetro, a lo que sumar el tiempo de preparación. El tiempo de preparación varía si se trata de bebidas (agua o refresco) o de una comida. Se añadirán 2 minutos en el caso de la bebida y 10 minutos en el caso de la comida.

Tareas a realizar:

1. Completa el diagrama de clases UML de la primera parte del proyecto añadiendo la jerarquía de clases de **Drink** (en StarUML). Presta atención a las características (*atributos*) y funcionalidades (*métodos*) que hay que establecer en cada clase. Analiza cómo deben ser los métodos (abstractos o concretos) y ponlos explícitamente sólo en las clases que los necesiten. Coloca las clases de la jerarquía de bebidas en el paquete **packDrink** que estará incluido en el paquete **packFeed** (ten en cuenta que tendrás que cambiar de paquete la clase **Drink**).
2. Incluye en la jerarquía obtenida, al menos ***dos nuevas clases*** (las que tengan sentido y en el nivel adecuado), aportando su descripción (en cuanto a características y funcionalidad).
3. Implementa la jerarquía de la clase **Drink** siguiendo el diseño de UML creado en la tarea anterior.
4. Actualiza la implementación realizada anteriormente con lo visto en la asignatura (por ejemplo, el método *equals*).
5. Haz las modificaciones y ampliaciones de la clase **Pub** para que contemple y pruebe las funcionalidades que se solicitan a continuación:
   5.1. **sortDrinks**, ordena la lista de bebidas en función de su nombre. Utiliza para ello las interfaces vistas y utilizadas en la asignatura.
   5.2. **cheapestBeer**: devuelve la cerveza más barata que haya en la cervecería.
   5.3. Ahora haciendo uso de tu imaginación, incluye **dos métodos nuevos** que trabajen con la lista de bebidas.
6. Incluye en la clase PubCheck las instrucciones necesarias para verificar el correcto funcionamiento de los últimos métodos incluidos en la jerarquía de clases:
   6.1. Si recoge adecuadamente la información de todos los tipo de bebidas.
   6.2. Si calcula bien el precio de todos los tipos de bebida.
   6.3. Si ordena bien las bebidas de la cervecería por su nombre.
   6.4. Si obtiene bien la cerveza más barata.