Ejercicio 1 Carrera de Bicicletas

Objetivo

El ejercicio consiste en programar mediante el uso del patrón de diseño Factoría Abstracta una carrera de bicicletas, las cuales serán de distintos tipos, ya sea carretera o montaña.

Descripción de la solución

Para solucionar el problema mediante el patrón Factoría Abstracta me he tenido que basar en varios conceptos:

1. La clase Bicicleta va a ser una clase abstracta de la que heredarán dos clases BicicletaMontana y BicicletaCarretera, con esto creamos el concepto de Bicicleta no el objeto concreto.
2. Las bicicletas van a ser de dos tipos uno llamado BicicletaMontana y otro llamado BicicletaCarretera estos tipos van a estar compuestos por piezas, es decir, van a tener un ArrayList de piezas. BicicletaMontana y BicicletaCarretera son dos clases concretas que van a realizar el trabajo de objetos concretos compuestos.
3. La clase Pieza es una clase abstracta que implementa un método toString() que nos dará la información de que tipo de Pieza tenemos y de la que heredan los tipos de pieza que voy a tener, en este caso Cuadro, Manillar y Ruedas. Estas últimas también son abstractas ya que su función es representar el concepto del objeto concreto.
4. Las clases CuadroMontana, CuadroCarretera, etc… son objetos concretos que heredan de Cuadro, Manillar y Ruedas. La funcionalidad de estas clases no es más que darle sentido lógico al programa y por ello no tienen métodos ni variables, simplemente sirven para crear una estructura lógica que puede ser usada en un entorno de desarrollo, pudiéndose añadir los métodos y variables que se consideren.
5. Para cada Pieza se ha creado su factoría correspondiente y cada una de estas heredan de una interfaz llamada FactoriaPieza que representa el concepto de factoría de una pieza pero que no sirve para crear objetos concretos. Cada una de las factorías de objetos concretos llaman al constructor del objeto de ese tipo.
6. Por último tenemos una interfaz Carrera que es implementada por factorías concretas, como serían FactoriaCarrera que se encarga de crear una carrera dependiendo del tipo que le pasen, FactoriaCarreraCarretera que crea una carrera con bicicletas del tipo Carretera y por último FactoriaCarreraMontana que crea una carrera con bicicletas del tipo Montana.
7. Estas dos últimas clases se encargan de crear, con el constructor del tipo apropiado ya sea Montaña o Carretera, tantas bicicletas como se indiquen en una variable N. Para ello tienen que crear las piezas que componen a la bicicleta y añadírselas al objeto BicicletaTipo creado, para que posteriormente sean añadidas al ArrayList.

8. Una Carrera lo he representado mediante un ArrayList<Bicicletas> el cual va a ser usado por dos Hebras distintas una HebraCarreraCarretera y otra HebraCarreraMontana las cuales se encargan de retirar las bicicletas de la carrera cumpliendo con los porcentajes de 20% de Carrera y Montaña y 10% en general.