Taller de Java ­ Práctica de polimorfismo

Respuestas:

1. Puedo asegurar esto dado a que los elementos que pueden ser afinados son los que implementan la interfaz Afinable.
2. El método afinar() pertenece a la interfaz Afinable. Solo aquellos que la implementen podrán ser afinados.
3. Para poder aprovechar el polimorfismo que ofrece Java. Puedo definir un objeto del tipo Instrumento e instanciarlo como cualquiera de los otros instrumentos que han sido definidos en la orquesta, además de definir métodos que sean comunes a todos los instrumentos existentes sin importar su tipo. Por ejemplo, un Banjo. Dado a que un Banjo es un Instrumento de cuerda, que a su vez es un instrumento, es posible de hacer.

Instrumento instrumento = **new** Banjo();

System.***out***.println(instrumento.tocar());

1. Esas dos subclases me permiten especializar la herencia. Un Banjo y un Laud son instrumentos de cuerda, mientras que una Flauta, Tuba u Oboe son elementos de viento. Si en el futuro quisiera añadir algún método exclusivo de alguno de los dos tipos (por ejemplo, un método para reemplazar una cuerda) puedo hacerlo en la subclase de dichos instrumentos. Si no tuviera esas subclases y quisiera crear dicho método, debería definirlo tanto en Banjo como en Laud (y teniendo el cuidado de no agregárselos a Flauta, Tuba y Oboe). Además, si en un futuro implemento más clases como Guitarra o Bajo, debería añadir ese método a cada una de esas clases.
2. Se hace de esa forma porque la clase Instrumento (y las dos subclases que heredan de la misma en el siguiente nivel de la jerarquía) es una clase abstracta. Los métodos abstractos solo definen su firma en la clase abstracta y luego se sobrecargan en la clase que hereda de la misma, así el intérprete de java sabrá a cuál de todos los métodos llamar, dependiendo de la forma en que lo instancié.
3. El constructor va a crear el objeto Orquesta. Los setters y los getters me permitirán pasarle a la orquesta una lista de instrumentos, o bien, obtenerla desde la misma orquesta. Los métodos add, se encargarán de agregar elementos de viento o de cuerdas a la orquesta. Los métodos tocarVientos y tocarCuerdas se encargarán de tocar únicamente los instrumentos que pertenezcan a esas clases en particular. Mientras que tocar, se encargará de tocar todos los instrumentos pertenecientes a la orquesta.
4. Con los setters puedo pasarle a la orquesta una lista de instrumentos de viento o de cuerdas (o de ambas) que haya sido definida fuera de la orquesta para que pasen a formar parte de la misma. Con los getters, puedo guardar los elementos de la orquesta en una lista nueva (distinguiendo siempre entre la clase a la que pertenecen).
5. Debo definir setters y getters para el tono en la clase Instrumento, pero se deberán implementar en cada una de las subclases correspondientes, dado a que la clase Instrumento es abstracta.