Agregación\_Composición.  
**La agregación y la composición son relaciones entre dos clases. Una clase se compone de campos, estos campos pueden ser de tipo de dato simple o pueden ser referencias (tipo de dato: clase), es decir al ser inicializados serán objetos internos del objeto principal.**  
proyecto: **automobile**

Genere un proyecto con clase que incluya el método main, el nombre de la clase será; automobileMain. Agregue las siguientes 4 clases.

Defina la clase **Motor** con los atributos: **private** **boolean** **encendido**. **Constructor público** sin parámetros: inicia el **encendido** en **false**. Método **público void turnOn()**, coloca el **encendido** en **true**; Método **público** **void** **turnOff()**, coloca el **encendido** en **false**. Método **público void display()**, muestra en pantalla los valores de los atributos.

Defina la clase **Wheel** con los atributos: **private boolean** **rolling**, **private short speed**. **Constructor** **público** sin parámetros: inicia el **rolling** en **false, speed** en 0, **private short SPEEDLIMIT**. Método **público** **void** **roll**(short speed), coloca el **rolling** en **true si el parámetro speed es positivo, y asigna al campo speed el valor del parámetro speed**; Método **público** **void** **break**(), coloca el **rolling** en false y **speed** en cero. Método **público void display()**, muestra en pantalla los valores de los atributos.

Defina la clase **Tank** con los atributos: **private final int** **CAPACITY** y **private int level. Constructor** **público** con dos parámetros (**int CAPACITY, int level**) para iniciar **CAPACITY** y **level**: en ningún caso **level** superará a **CAPACITY,** ni será negativo**.** Método **público** **void** **downLevel**(**int km**), decrementa **level** en una unidad por cada 12 km; Método **público** **void** **UpLevel**(**int units**), incrementa **level** en **units.** Método **público void display()**, muestra en pantalla los valores de los atributos.

La clase **Automobile**, tiene los atributos motor y wheels(4) y tank.Defina la clase **Automobile** con los **atributos privados**: motor, wheels(4) y tank. Un constructor que recibe dos parámetros int **CAPACITY** e int **level**. El **constructor público,** inicializa **motor, wheels y tank**. Método **público void start(short speed, int km),** llama al método **turnOn() del objeto motor,** llama al método **roll(short speed) de los objetos Wheel,** llama al método **downLevel(int km) del objeto tank;** Método público **stop(int units),** el método llama al método **turnOff() del objeto motor,** llamaal método **break()** de cada objeto en wheels, y llama al método **UpLevel(int units)** **del objeto tank**; Método público **void display(),** llama a los métodos **display()** de **motor, wheels y tank.**

En el main de la clase AutomobileMain, inicialice un objeto **Automobile automobile**. Haga que automobile llame al método **display()** y observe los valores iniciales del automobile. Haga que automobile llame al método **start()** solicitando los valores correspondientes de sus parámetros vía teclado**,** haga que automobile llame al método **display()** y observe los valores de automobile.Haga que automobile llame al método **stop()** solicitando el valor del parámetro **units** vía teclado**,** haga que automobile llame al método **display()** y observe los valores de automobile.