

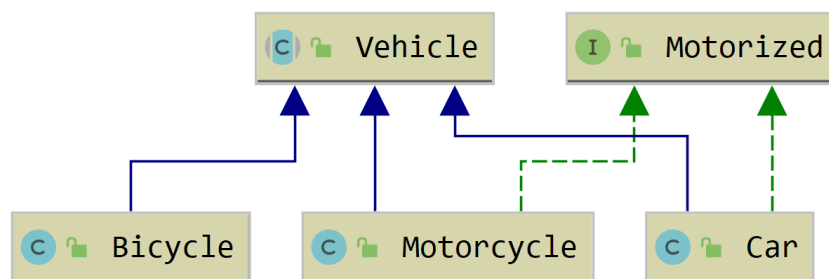
## 1 Vehicle

Die Klasse kapselt die Anzahl an Rädern und stellt öffentliche Arrays für Marken verschiedener Fahrzeugtypen bereit. Implementiere die Methoden und den Konstruktor.

Ein **Vehicle** braucht nicht unbedingt eine Marke zu haben(also **null** ist hier ok), die Räderanzahl sollte aber positiv sein. Falls dem nicht so ist, wird eine **IllegalArgumentException** geworfen.

## 2 Motorized

Das Interface stellt Methoden zur Verwaltung von motorisierten Dingen zur Verfügung. Zu keinem Zeitpunkt darf ein Objekt eine Motorleistung  $< 0$  haben.



## 3 Bicycle

Die Klasse repräsentiert Fahrräder, die in dieser Laborübung keinen Motor besitzen, also das Interface nicht implementieren. Implementiere mindestens:

- `public Bicycle(String brand)`  
Erzeugt ein Fahrrad mit 2 Rädern
- `public Bicycle(String brand, int wheels)`  
Ist die Radanzahl nicht 2 oder 3 wird eine **IllegalArgumentException** geworfen.

## 4 Motorcycle

Die Klasse repräsentiert Motorräder<sup>1</sup>. Implementiere mindestens:

- `public Motorcycle(String brand, double power)`  
Erzeugt ein Motorrad mit 2 Rädern und der übergebenen Motorleistung. Der Motor ist aus.
- `public Motorcycle(String brand, int wheels, double power, boolean started)`  
Erzeugt ein Motorrad mit der übergebenen Motorleistung und möglicherweise laufendem Motor. Ist die Radanzahl nicht 2 oder 3 wird eine **IAE** geworfen.

<sup>1</sup>Je nach Anwendungsfall könnte man entscheiden, ob **Motorcycle** **Bicycle** extended

## 5 Car

Die Klasse repräsentiert Autos. Implementiere mindestens:

- `public Car(String brand, double power)`  
Erzeugt ein Auto mit 4 Rädern und der übergebenen Motorleistung. Der Motor ist aus.
- `public Car(String brand, double power, boolean started)`  
Erzeugt ein Auto mit 4 Rädern, der übergebenen Motorleistung und möglicherweise laufendem Motor.