Coding Challenges: OOP

Pillars of Object Oriented Programming

Challenge I

Create a class called `Vehicle`. The `Vehicle` class should have the following:

- An `enum` publicly available called `Medium` with the values LAND, AIR, WATER
- A property called `Speed`. This should be read-only outside of the class. This is not allowed to be negative
- A property called `Name`. This should be read-only outside of the class
- A property called `Occupants`. This is not allowed to be negative
- A constant called `Capacity`. This must be at least 1
- A variable called `Media`, which is a list of `Mediums`. This should only be accessible by the class and any of its children

Challenge II

Add the following methods to `Vehicle`:

- A constructor that sets `Speed`, `Occupants`, `Name`, and `Capacity` to default values
- A method called `Accelerate` which changes `Speed` by the specified amount
- An abstract method called `Turn`
- A method called `ToString` that prints out the current `Speed`, all `Media`, the `Name`, and the number of `Occupants`

Challenge III

Create a class called `Car` that extends `Vehicle`. `Car` should have the following:

- A variable called `SteeringRotation`. This should not be accessible outside of the class. It must be between -1440 and 1440, inclusive
- A property called `TireRotation`. This should be read-only outside of the class. It must be between -90 and 90, inclusive
- A property called `Gear`. This should be read-only outside of the class
- A constructor that calls the base constructor of `Vehicle` and then sets the two rotation values to 0 and adds the LAND `Medium` to `Media`
- Implement `Turn` to rotate the `SteeringRotation` by the specified amount, in degrees, and the `TireRotate` by 1/16th the specified amount, in degrees.

Challenge IV

Make the following modifications to `Vehicle`:

- Implement constructor chaining to allow for values set by the constructor to be input instead

Make the following modifications to `Car`:

- Implement constructor chaining to allow inputs for values that `Vehicle's` constructor sets
- Override the `Accelerate` function to call the base implementation, and then set the `Gear` via the following equation: `Gear` = `(Integer)Math.ceil(0.4444 * (Math.sqrt(Speed)))`