Access Inherited Members with Protected

While working on `Vehicle` and `Sedan`, you may have seen this error:

```
Sedan.cs(11,13): error CS0200: Property or indexer 'Vehicle.Wheels' cannot
be assigned to -- it is read only
```

Remember `public` and `private`? A `public` member can be accessed by any code outside of the enclosing class. A `private` member can only be accessed by code within the same class.

The above error comes up because either:

- There is no setter for `Vehicle.Wheels`, or
- The setter is private

How do we fix this problem? Making the setter public is not secure. Making it private is too restrictive – we only want the subclass `Sedan` to access the property. C# has another access modifier to solved that: `protected`!

A *protected* member can be accessed by the current class and any class that inherits from it. In this case, if the setter for `Vehicle.Wheels` is protected, then any `Vehicle`, `Truck`, and `Sedan` instance can call it.

☑Instructions

1.

In `Vehicle`, add a `protected` setter to each of these properties:

- `LicensePlate`
- `Speed`
- `Wheels`

Hint

The format of an automatic property with `get` and `protected` `set` is:

```
public bool IsFake
{ get; protected set; }
```