

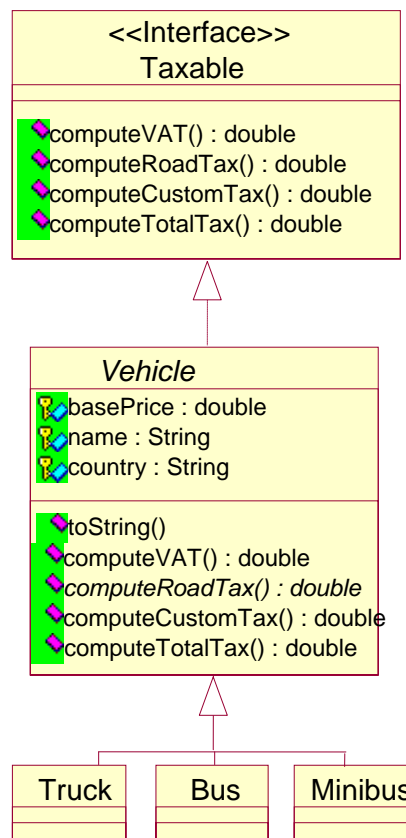
Interfaces

Problem 1

A car dealer sales domestic and foreign vehicles like trucks, buses and minibuses. To compute a vehicle's price the dealer has to add some requested taxes to the vehicle's base price. The taxes the dealer's sales management system should consider are the followings:

- VAT: 19% from vehicle's price
- Road tax: 3% for minibuses, 4% for buses and 5% for trucks
- The Customs tax: 10% for the vehicles made abroad.

In order to reuse the code, the system's model introduces the class `Vehicle`. It abstracts three classes: `Truck`, `Bus`, and `Minibus` by factorizing the common data structure and behaviour of them. The system behaviour regarding tax application is modeled as a four methods interface `Taxable`. The class diagram of the problem is presented in the following figure:



`Vehicle` implements three methods and offers the fourth, `computeRoadTax()`, as an abstract method. This way it gives up the method implementation to its subclasses. From this reason, the class `Vehicle` is an abstract class.

Requirement. Implement the above class diagram and write a test class that prints for at least 3 vehicles the following information:

- vehicle's type: "Truck", "Bus" or "Minibus"
- manufacture's name
- manufacture country
- base price of the vehicle
- the total sum of taxes that have to be paid