Consider the following definition for a data structure called `MotorVehicle`:

```plaintext
# A MotorVehicle is a model, year, color, and price

data MotorVehicle:
    | vehicle( model :: String, 
    year :: Number, 
    color :: String, 
    price :: Number )

end
```

To make instances of this structure, I would write:

```plaintext
___________ = ______________________________________________________________________

___________ = ______________________________________________________________________
```

Choose one of your above instances, and note which dot-accessors you would use to access each of its fields:

```plaintext
____________________________________________________________________________________

____________________________________________________________________________________

____________________________________________________________________________________

____________________________________________________________________________________
```
Which of the following are functions that could be written based on the data definition for `MotorVehicle`? Check all that apply

☐ # same-license : MotorVehicle, String -> Boolean
   # Consumes a MotorVehicle and String, produces true if the
   # given MotorVehicle’s license plate is the same as the
   # given String

☐ # how-old : MotorVehicle, Number -> Number
   # consumes a MotorVehicle and a year. Produces the age of
   # the vehicle by subtracting its year from the given year.

☐ # more-expensive : MotorVehicle, MotorVehicle -> Boolean
   # consumes two MotorVehicle and produces true if the first
   # MotorVehicle is more expensive than the second

☐ # is-under-warranty : MotorVehicle -> Boolean
   # Consumes a MotorVehicle, produces true if the given
   # MotorVehicle has a mileage of less than 100,000 miles

☐ # paint-job : MotorVehicle -> MotorVehicle
   # Consumes a MotorVehicle and produces a MotorVehicle which
   # is the same as the given MotorVehicle, but painted red