Carpool

Lehrziele:

- Vererbung in C#
- Wiederholung v. Assoziationen

Ihr Programm verwaltet einen Fuhrpark (Carpool), der verschiedene Fahrzeuge enthalten kann.

Ihre Aufgabe ist es, die Klasse Carpool und einige Fahrzeugklassen inkl. einer Basisklasse zu modellieren und die Vererbungshierarchie von Vehicles (Fahrzeugen) so zu implementieren, dass die Unittests bestanden werden und die Anforderungen erfüllt sind.

Klassen:

Carpool:

Diese Klasse verwaltet einen Fuhrpark mit einer Garage. Es kann angenommen werden, dass eine Garage genau 5 Stockwerke mit je 10 Parkplätzen aufweist.

- 1 Feld: Vehicle-Array zum Speichern der aktuellen Fahrzeuge. (1/2dimensional)
- Property **Count:** Liefert die Gesamtanzahl an Fahrzeugen zurück. (readonly)
- Die Methode ***bool AddVehicle(Vehicle)*** parkt ein Fahrzeug am nächsten freien Parkplatz und liefert *true* zurück, wenn dies erfolgreich ausgeführt wurde, ansonsten *false*.
- Die Methode ***bool AddVehicleAtSpot(Vehicle, Floor, Parkingspace)*** parkt ein Fahrzeug an der angegebenen Stelle. Ist dies nicht möglich, da diese bereits besetzt ist, wird *false* zurückgegeben, ansonsten *true*.
- Die Methode ***Vehicle GetVehicleAtSpot(Floor, Parkingspace)*** liefert ein Fahrzeug zurück, dass am gegebenen Platz steht, steht dort keines, wird *null* zurückgegeben.
- Die Methode ***Vehicle[] GetVehiclesOnFloor(Floor)*** liefert ein Array von Fahrzeugen zurück, die sich auf einem bestimmten Stockwerk befinden. *(Achtung! Sorgen Sie dafür, dass im Array keine Lücken vorkommen!)*
  Ist der Parameter *Floor* ungültig, soll *null* zurückgegeben werden.

Vehicle:

Als abstrakte Basisklasse verwaltet ***Vehicle*** die Daten, die alle Fahrzeuge gemeinsam haben.

Daten:

- Farbe – Enum (Red, Yellow, Green, Blue, Black, White)
- IsDriving - Bool
- Auslieferungsdatum – DateTime
- Gefahrene Zeit in Minuten – Double
- Maximale Geschwindigkeit – Int
- Property Age (readonly)
- Überschreiben Sie die ToString()-Methode (siehe Unittests)
- Die Methode ***StartDriving(),*** um loszufahren. -> Bool
- Die Methode ***StopDriving(),*** um stehen zu bleiben. -> Bool
- Die Methode ***GetType(),*** die jedes Child selbst implementiert. -> String (lt. Unittest)

Car:

Die Klasse *Car* implementiert die Methoden und Properties eines Autos.

- Es wird der Konstruktor der Basisklasse verwendet, um Farbe, Auslieferungsdatum und Maximalgeschwindigkeit zu speichern.
- ToString() wird überschrieben (+Verwendung der Basisklasse)
- ***GetType()*** wird von der Basisklasse implementiert.

Individuelle Felder:

- AreDoorsOpen – Bool (default: false)
- Methode ***OpenDoors()*** öffnet die Türen). -> Bool
- Methode ***CloseDoors()*** schließt die Türen). -> Bool

Motorcycle:

Die Klasse *Motorcycle* implementiert die Methoden und Properties eines Motorrads.

- Es wird der Konstruktor der Basisklasse verwendet, um Farbe, Auslieferungsdatum und Maximalgeschwindigkeit zu speichern.
- ToString() wird überschrieben (+Verwendung der Basisklasse)
- ***GetType()*** wird von der Basisklasse implementiert.

Individuelle Felder:

- TimesHonked – Int
- Methode ***Honk()*** zum Hupen. -> void