# Train-Manager

Es soll eine einfache Zugverwaltung implementiert werden. Ein Zug besteht aus mehreren Wagons (Carriage). Es gibt sowohl Passagierwagons (PassengerCar) als auch Transportwagons (CargoCar). Diese drei Wagonklassen sind mittels Vererbung sauber objektorientiert zu realisieren.

Die Klasse Train verwaltet eine Liste mit den Wagons des Zuges und stellt verschiedene Methoden zur Verfügung.

Sie benötigen die Klassen Carriage, PassengerCar, CargoCar und Train.

**Hinweis**: Die leere Klasse Carriage ist bereits in der Vorlage enthalten. In dieser Klasse sind einige Konstanten vorgegeben, die für verschiedene Methoden benötigt werden!

## Klasse Carriage:

Diese abstrakte Klasse ist die Basisklasse für PassengerCar und CargoCar.

Private Felder und jeweils readonly Properties:

**emptyWeight** (als Double): Leergewicht des Wagons (ohne Fracht oder Passagiere)
**carriageNumber** (als Integer): Wagonnr.

Methoden/Konstruktor:

- **Konstruktor:** Nimmt die Werte für emptyWeight und carriageNumber entgegen und setzt die internen Felder.

  **Gültigkeitsprüfung carriageNumber:** Die carriageNumber muss im Konstruktor auf Gültigkeit geprüft werden, wenn die übergebene Nummer nicht gültig ist, wird das interne Feld carriageNumber auf 1 gesetzt.

  Die Nr. muss aus 8 Ziffern bestehen! Die Gültigkeit der Nr. kann überprüft werden, indem die Quersumme aller 8 Ziffern gebildet wird, wobei für jede Ziffer mit ihrer Position multipliziert wird. (D.h. die 1. Ziffer + 2. Ziffer * 2 + 3. Ziffer *3 + … + 8. Ziffer * 8)
  Diese Quersumme muss auf 0 enden (d.h. 0 als letzte Stelle beinhalten)!

  Beispiel CarriageNumber: **63251679**
  6*1+3*2+2*3+5*4+1*5+6*6+7*7+9*8 = 200

  Die letzte Stelle von 200 ist 0, daher ist diese Nr. gültig.

- **double GetProfit():** Diese Methode berechnet den Gewinn, der durch diesen Wagon erzielt wird. Da die Berechnung je Wagontyp unterschiedlich ist, muss diese Methode von jeder konkreten Wagonklasse (PassengerCar, CargoCar) selbst implementiert werden.

- **double GetFullWeight():** Diese Methode berechnet das Gesamtgewicht des Wagons (Leergewicht+Fracht/Passagiergewicht. Da die Berechnung je Wagontyp unterschiedlich ist, muss diese Methode von jeder konkreten Wagonklasse selbst implementiert werden.

## Klasse CargoCar:

Diese Klasse ist erbt von der Klasse Carriage.

Private Felder und Properties (read/write):

**cargoWeight** (als Double): Frachtgewicht in Tonnen.
**pricePerTon** (als Double): Wieviel Umsatz wird pro Tonne Fracht erzielt.

Methoden/Konstruktor:

- **Konstruktor:** Nimmt die Werte für emptyWeight, carriageNumber, cargoWeight und pricePerTon entgegen und setzt die internen Felder (ohne Codeverdopplung).

- **double GetProfit():** Liefert den Gewinn den dieser Wagon erzielt. Beachten sie auch die Kosten pro Frachtwagen (Konstante in Klasse Carriage).

- **double GetFullWeight():** Diese Methode berechnet das Gesamtgewicht des Wagons (Leergewicht+Frachtgewicht)

## Klasse PassengerCar:

Diese Klasse ist erbt von der Klasse Carriage.

Private Felder und Properties (read/write):

**numberOfPassengers** (als Integer): Anzahl der Passagiere im Wagon.
**pricePerTicket** (als Double): Wieviel Umsatz wird pro Passagier in diesem Wagon erzielt (→ Tickets kosten pro Wagon unterschiedlich viel (z.B. 1. Klasse, 2. Klasse…))

Methoden/Konstruktor:

- **Konstruktor:** Nimmt die Werte für emptyWeight, carriageNumber, numberOfPassenger und pricePerTicket entgegen und setzt die internen Felder (ohne Codeverdopplung).

- **double GetProfit():** Liefert den Gewinn den dieser Wagon erzielt. Beachten sie auch die Kosten pro Passagierwagen (Konstante in Klasse Carriage).

- **double GetFullWeight():** Diese Methode berechnet das Gesamtgewicht des Wagons (Leergewicht+Passagiergewicht). Für die Berechnung des Passagiergewichts ist die Kostante für das durchschnittliche Gewicht pro Person aus Klasse Carriage zu verwenden.

## Weitere Vorgaben für PassengerCar und CargoCar:

Beachte, dass die maximale Anzahl der Passagiere bzw. das maximale Frachtgewicht pro Wagon nicht überschritten werden darf (Konstante in der Klasse Carriage vorhanden!). Wenn den Feldern zu viel zugewiesen werden sollte, so soll der Wert automatisch auf das Maximum korrigiert werden.

## Klasse Train:

Private Felder und Properties (read):

**List<Carriage>** carriageList: In der Klasse Train soll eine Collection (Liste) von Wagons implementiert werden. Zusätzlich soll eine readonly Property CarriageList erstellt werden, welche die Liste zurückliefert.
**maxTrainWeight** (als Double): Maximal zulässiges Gesamtgewicht des Zuges (Leergewicht+Beladung der Wagons).
**CountOfPassengerCars:** Berechnete **Property**, welche die Anzahl der Passagierwagons in der Liste als Integer zurückliefert.

Methoden/Konstruktor (Methodenköpfe dieser Klasse sind bereits in der Vorlage vorgegeben):

- **Konstruktor:** Nimmt den Wert für das Feld maxTrainWeight entgegen uns setzt diesen.

- `public double GetTrainWeight();` Berechnet das Gesamtgewicht des Zuges. D.h. die Summe der Gewichte aller Wagons (Leergewicht und Fracht bzw. Passagiergewicht!)

- `public Carriage GetMostProfitableCarriage();` Liefert jenen Wagon zurück, der am meisten Gewinn erzielt. Wenn es keinen Wagon gibt soll null zurückgeliefert werden!

- `public int GetAmoutOfPassengersInTrain();` Liefert die Anzahl aller Passagiere im Zug. D.h. die Summe aller Passagiere der Passagierwagons (in der Liste).

- `public bool AddCarriage(Carriage newCar);` Ein neuer Wagon wird (sortiert) in die Liste eingefügt, wenn das maximale Zuggesamtgewicht (Feld maxTrainWeight) durch den neuen Wagon nicht überschritten wird.
  Die Wagons sollen absteigend sortiert nach dem Gewicht in der Liste geführt werden. Der schwerste Wagon soll ganz vorne stehen!
  Die Methode returniert true, wenn der neue Wagon erfolgreich eingefügt wurde, sonst false.

- `public bool AddPassengersToCar(int carriageNumber, int newPassengers);`
  Diese Methode lässt weitere Passagiere in einen bestimmten Passagierwagon einsteigen. Als erster Parameter wird die Nummer des Wagons übergeben, in den die Passagiere einsteigen sollen. Als zweiter Parameter wird die Anzahl der zusäztlichen Passagiere übergeben.
  Beachte! Durch die neuen Passagiere erhöht sich das Gewicht des Wagons und er muss möglicherweise umgereiht werden (schwerere Wagons müssen vor den leichteren gereiht werden!)
  Die Methode returniert true, wenn die Passagiere erfolgreich eingestiegen sind. Falls nicht alle Passagiere in den Wagon / den Zug passen, so steigt keiner ein und es wird false zurückgeliefert. (Auch Zuggesamtgewicht beachten!)
  Falls Wagon nicht gefunden wird, wird ebenfalls false zurückgeliefert!
  Es kann davon ausgegangen werden dass die Wagonnr. eindeutig sind!