

## Zadanie – Samochody w C# z konstruktorami i walidacją danych

1. **Utwórz klasę `car`** z następującymi prywatnymi polami:
  - `brand` (marka samochodu, typu `string`)
  - `model` (model samochodu, typu `string`)
  - `year` (rok produkcji, typu `int`)
  - `mileage` (przebieg w kilometrach, typu `double`)
  - `fuelLevel` (poziom paliwa w litrach, typu `double`)
2. **Dodaj właściwości publiczne** dla każdego pola, z następującą logiką walidacji:
  - `Brand` i `Model` nie mogą być puste ani `null`; w przeciwnym razie ustaw domyślnie `"Unknown"`.
  - `Year` nie może być wcześniejszy niż 1886 (rok pierwszego samochodu) i nie większy niż bieżący rok. W przeciwnym razie ustaw domyślnie bieżący rok.
  - `Mileage` nie może być ujemny; jeśli podano ujemną wartość, ustaw 0.
  - `FuelLevel` nie może być ujemny ani większy niż pojemność baku (np. 80 litrów).
3. **Dodaj konstruktor domyślny**, który ustawia przykładowe wartości:
  - Marka: `"Toyota"`
  - Model: `"Corolla"`
  - Rok: 2020
  - Przebieg: 0
  - Poziom paliwa: 50
4. **Dodaj konstruktory przeciążone:**
  - Konstruktor jednoparametrowy – ustawia tylko markę, reszta wartości domyślna.
  - Konstruktor dwuparametrowy – ustawia markę i model.
  - Konstruktor trzyparametrowy – ustawia markę, model i rok.
  - Konstruktor pięcioparametrowy – ustawia wszystkie pola.

## 5. Dodaj metody do klasy `Car`:

- `PrintInfo()` – wyświetla wszystkie dane samochodu.
- `Drive(double distance)` – zwiększa przebieg o podaną wartość, zmniejsza paliwo o `distance * 0.1` litra; jeśli paliwa brak, wyświetl komunikat.
- `Refuel(double liters)` – zwiększa paliwo, nie przekraczając maksymalnej pojemności baku (80 litrów).

## 6. W metodzie `Main()`:

- Utwórz kilka obiektów klasy `Car` przy użyciu różnych konstruktorów.
- Wywołaj metody `PrintInfo()` dla wszystkich samochodów.
- Przetestuj metody `Drive()` i `Refuel()` oraz sprawdź logikę walidacji danych