Holke Toni
Professur für Didaktik der Informatik

Input – Programmierparadigmen 2024/2025
UML - Einführung

Tu Dresden 19.12.2024

Notwendigkeit OOP

- **Structs in C:** Dienen zur Zusammenfassung von Datenelementen unterschiedlicher Typen. Sie sind rein datenorientiert und bieten keine direkte Möglichkeit, Funktionen einzubinden. Was fehlt? Vereinigung von Daten mit Funktionen (Daten und Funktionalität)
- **Mangel bei Structs:** Die Trennung von Daten und Funktionalität kann zu redundantem Code und einer schwierigeren Verwaltung von Objekten führen.
- **Lösung:** Klassen in objektorientierten Sprachen: Bieten eine umfassendere Möglichkeit, Daten und die zugehörigen Operationen (Methoden) in einem einzigen Konstrukt zu kapseln. Dies führt zu einer besseren Organisation und Wiederverwendbarkeit von Code.

```c
struct Person {
    char name[50];
    int age;
    void printInfo() {
        printf(format: "Name: %s, Alter: %d\n", name, age);
    }
};



int main()
{
    struct Person person1 = {.name: "Max Mustermann", .age: 30};
    person1.printInfo();
    return 0;
}
```

```cpp
class Person {
    std::string_view m_name;
    int m_age;

public:
    void printInfo() const {
        std::cout << "Name: " << m_name << ", Alter: " << m_age << std::endl;
    }

    Person(const std::string_view name, const int age)
        : m_name(name), m_age(age) {}

};

int main()
{
    const Person person1 = {name: "Max Mustermann", age: 30};
    person1.printInfo();
    return 0;
}

```

Quelle: Eigene Darstellungen

Begriffe OOP

OOP = Das objekt-orientierte Paradigma organisiert den Code um "Objekte" herum, die Daten und Methoden (Funktionen) enthalten, die auf diese Daten angewendet werden können. (siehe Vorlesung Folie: 33)

Klasse = Baupläne für Objekte, die deren Struktur und Verhalten definieren

Objekt = Instanzen von Klassen, die Daten und Methoden besitzen

Attribute = Eigenschaften von Objekten

Kardinalitäten = Beziehungen zwischen Klassen

Darstellung von OOP mit UML

UML = Standardisierte Notation zur grafischen Darstellung OO-Sachverhalte

Beinhaltet:

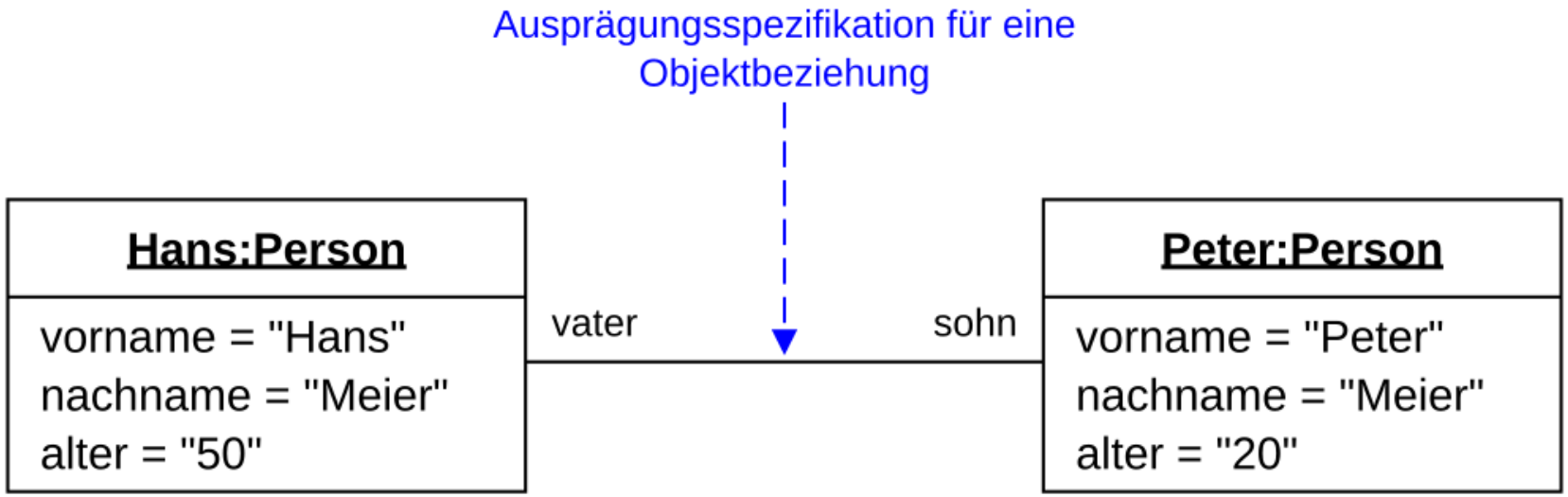
- Sieben Strukturdiagramme
- Sieben Verhaltensdiagramme

Programm: Draw.io

Strukturdiagramme	Verhaltensdiagramm
Klassendiagramm	Aktivitätsdiagramm
Kompositionsdiagramm	Use-Case-Diagramm
Komponentendiagramm	Interaktionsübersichtdiagramm
Verteilungsdiagramm	Kommunikationsdiagramm
Objektdiagramm	Sequenzdiagramm
Paketdiagramm	Zeitverlaufsdiagramm
Profildiagramm	Zustandsdiagramm

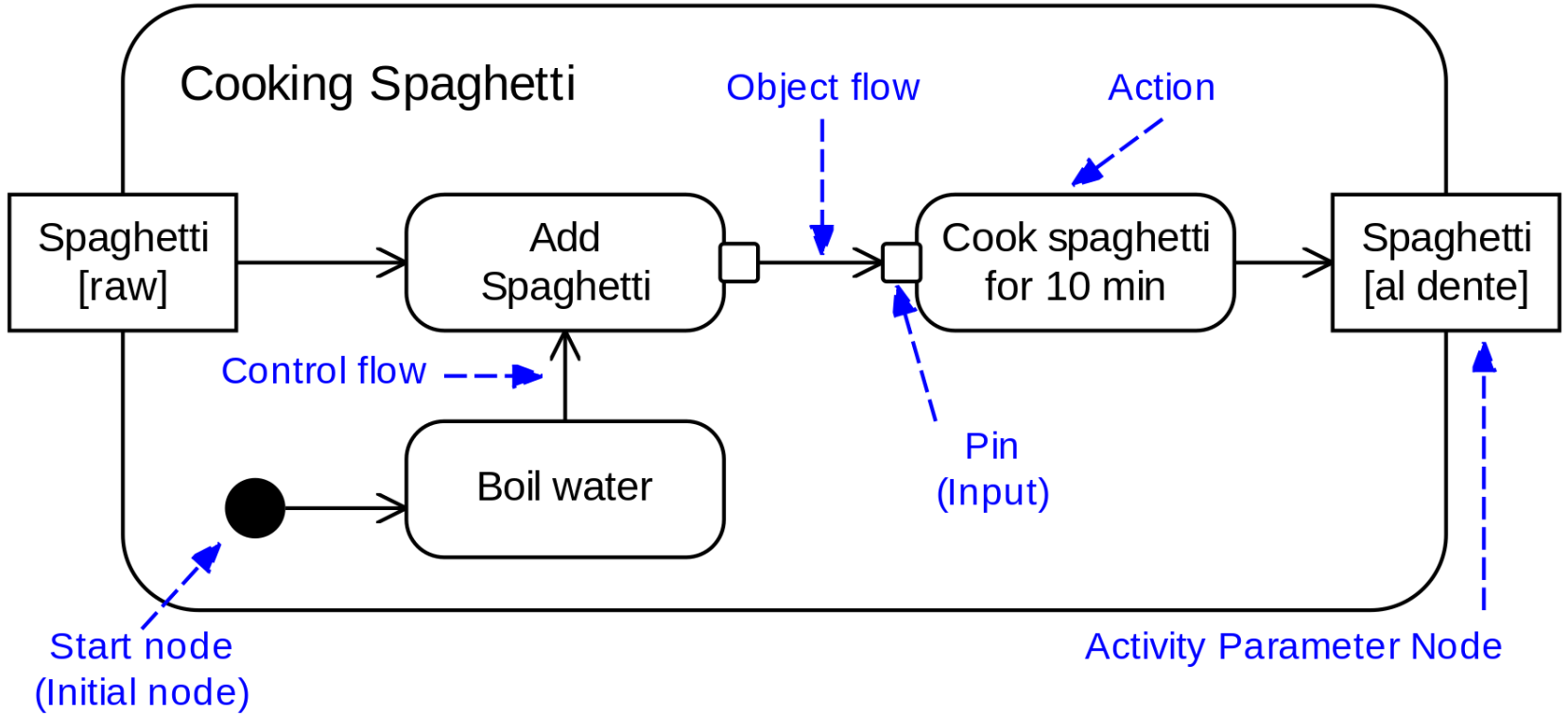
UML – Diagrammarten einige Bsp.

Objektdiagramm = Ein Objektdiagramm zeigt die Instanzen von Objekten und ihre Beziehungen zu einem bestimmten Zeitpunkt



UML – Diagrammarten einige Bsp.

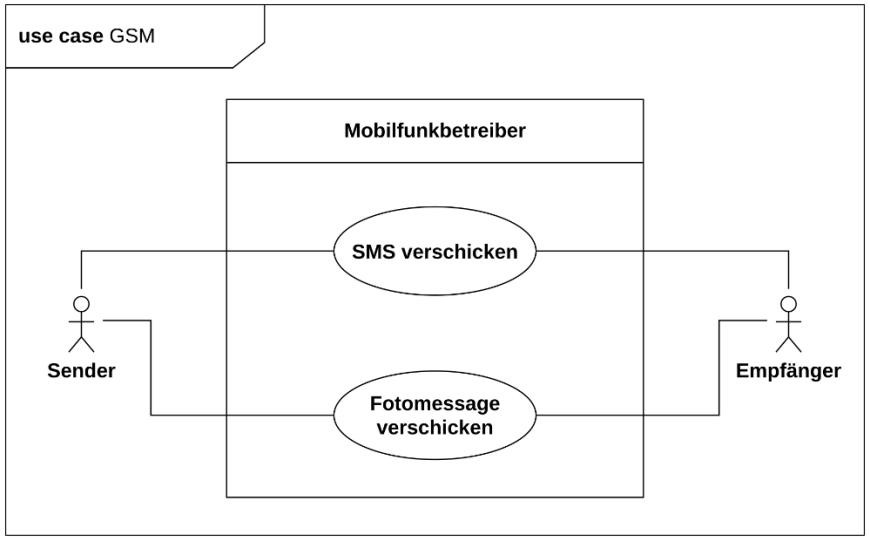
Aktivitätsdiagramm = Ein Aktivitätsdiagramm visualisiert den Ablauf und die Bedingungen von Aktivitäten und Prozessen im System



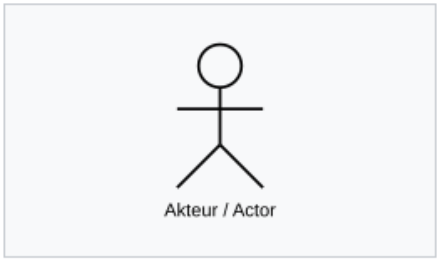
https://de.wikipedia.org/wiki/Unified_Modeling_Language#/media/Datei:Uml-Activity-Beispiel2.svg

UML – Diagrammarten einige Bsp.

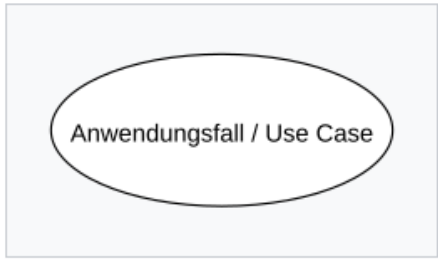
Use-Case-Diagramm = Ein Use-Case-Diagramm beschreibt die Interaktionen zwischen Akteuren und den Use Cases des Systems



Der **Systemkontext** wird durch **Systemgrenzen** in Form von Rechtecken gekennzeichnet.



Akteure werden als „Strichmännchen" dargestellt, welche sowohl Personen wie Kunden oder Administratoren als auch ein System darstellen können.

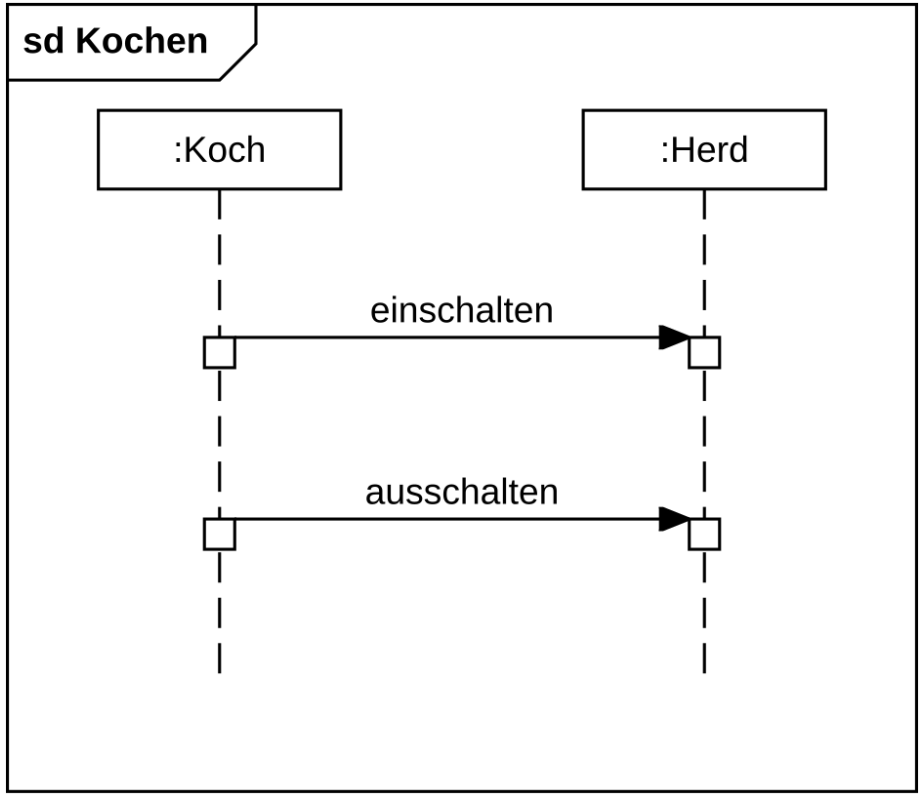


Anwendungsfälle werden in Ellipsen dargestellt. Sie müssen (z. B. in einem Kommentar oder einer eigenen Datei) beschrieben werden.

https://de.wikipedia.org/wiki/Unified_Modeling_Language#/media/Datei:Uml-UseCase-Beispiel3.svg

UML – Diagrammarten einige Bsp.

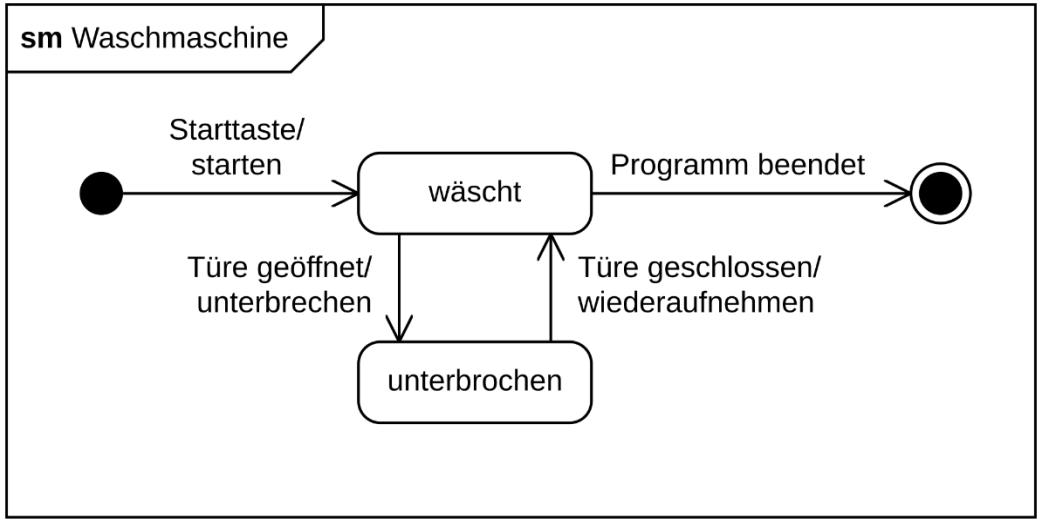
Sequenzdiagramme = Ein Sequenzdiagramm stellt die zeitlich geordnete Kommunikation zwischen Objekten dar.



https://de.wikipedia.org/wiki/Unified_Modeling_Language#/media/Datei:UmlSequenzdiagramm-1.svg

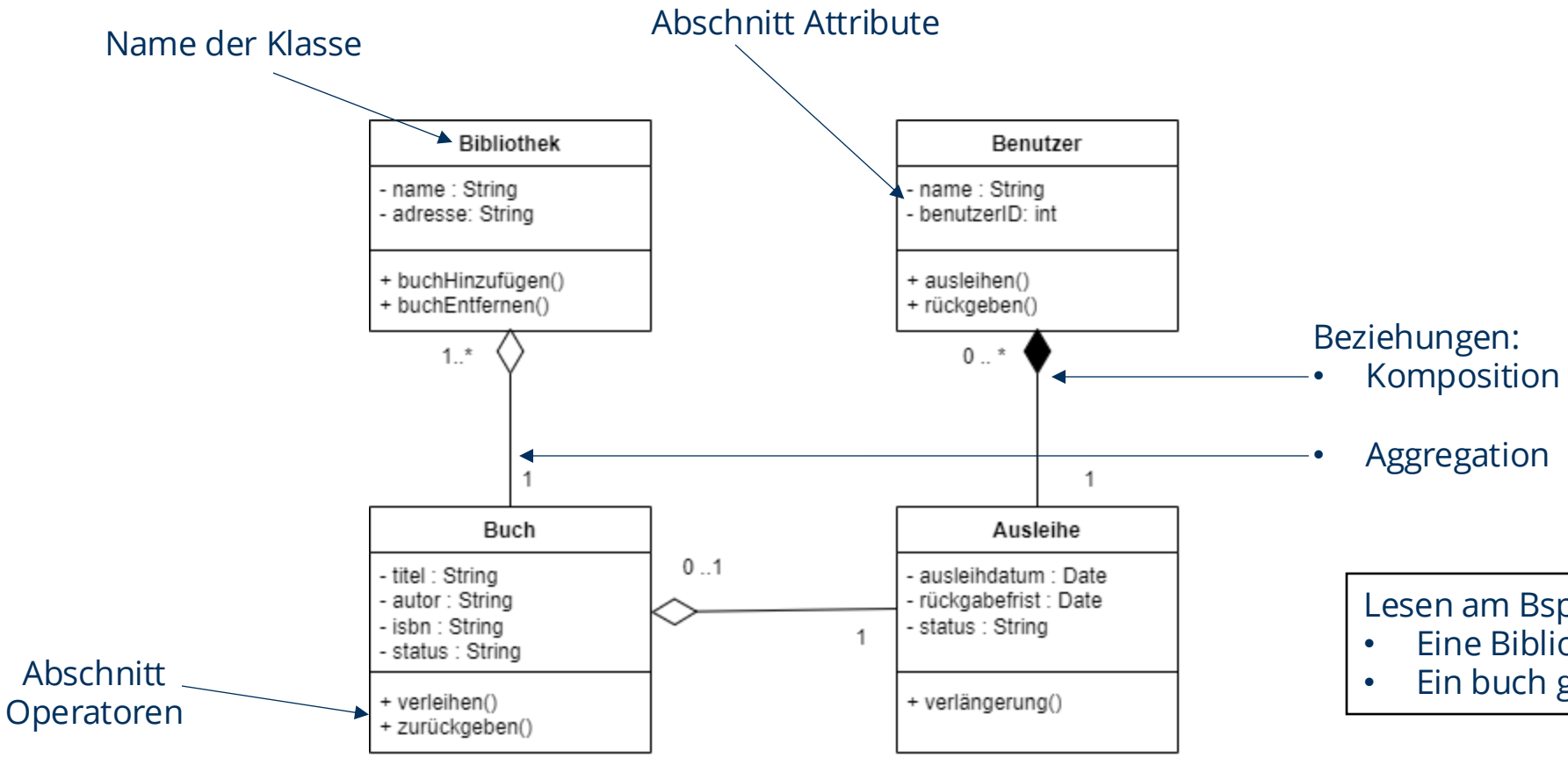
UML – Diagrammarten einige Bsp.

Zustandsdiagramm = Ein Zustandsdiagramm beschreibt die Zustände eines Objekts und die Übergänge zwischen diesen Zuständen



https://de.wikipedia.org/wiki/Zustandsdiagramm_(UML)#/media/Datei:Uml-Zustandsdiagramm-5.svg

UML – Klassendiagramme Begriffe



Beziehungen:

- Komposition
- Aggregation

Lesen am Bsp der Klassen Bibliothek und Buch:

- Eine Bibliothek verwaltet viele Bücher. (1..*)
- Ein buch gehört zu genau einer Bibliothek. (1)

Quelle: Eigene Darstellung

UML – Klassendiagramm Beispiel: 3.3 (Auto)

3.3 Klasse Fahrzeug und einfache Objekte

Schreiben Sie ein C++-Programm in dem die Klasse „Fahrzeug“ mit den privaten Membervariablen Personen (int), Tankvolumen (float), Verbr_pro_100km (float), aktTankinhalt (float) und kmStand (float) deklariert sind.

Neben dem Konstruktor sollen die Memberfunktionen enthalten sein:

float Fahren (float km); (Rückgabe: aktueller Tankinhalt; Achtung! Darf nicht kleiner 0 werden!!)

float Tankinhalt(void);

float Tanken (float liter); (Rückgabe: aktueller Tankinhalt)

float Reichweite(void);

float get_kmStand(void);

Erzeugen Sie die folgenden Objekte: PKW, LKW, Bus etc.

Testen Sie Ihre Implementierung mit einem kleinen Hauptprogramm, dass alle Methoden an zwei Objekten mindestens einmal aufruft!

Quelle: Übungsskript Prof. Luntovskyy

UML – Klassendiagramm Beispiel: 3.3 (Auto)

Quelle: Eigene Darstellung

3.3 Klasse Fahrzeug und einfache Objekte

Schreiben Sie ein C++-Programm in dem die Klasse „Fahrzeug“ mit den privaten Membervariablen Personen (int), Tankvolumen (float), Verbr_pro_100km (float), aktTankinhalt (float) und kmStand (float) deklariert sind.

Neben dem Konstruktor sollen die Memberfunktionen enthalten sein:

float Fahren (float km); (Rückgabe: aktueller Tankinhalt; Achtung! Darf nicht kleiner 0 werden!!)

float Tankinhalt(void);

float Tanken (float liter); (Rückgabe: aktueller Tankinhalt)

float Reichweite(void);

float get_kmStand(void);

Erzeugen Sie die folgenden Objekte: PKW, LKW, Bus etc.

Testen Sie Ihre Implementierung mit einem kleinen Hauptprogramm, dass alle Methoden an zwei Objekten mindestens einmal aufruft!

Umwandlung UML

- Empfehlung CLion als Tool für C++ mit OOP und Klassen
- Eine cross-platform IDE für C und C++
- Motto:
  - Harness the power. Cut the complexity.
  - Free 30-day trial…
  - Free for students and teachers!
- URL: https://www.jetbrains.com/clion/

Noch fragen?