Hochschule Deggendorf

Objektorientierte Programmierung

Einleitung, Motivation, Definition

Prof. Dr. Peter Jüttner

Frage

Hochschule Deggendorf

Was wissen Sie schon über Objektorientierte Programmierung wissen?

Frage

Hochschule Deggendorf

… Begriffe

… Anwendung

… Ziele

… …

Ziele der Vorlesung

Hochschule Deggendorf

1. **Studenten kennen die wichtigsten Prinzipien der Objektorientierten Programmierung**
2. **Die Studenten beherrschen die wichtigsten Begriffe der Objektorientierten Programmierung**
3. **Schwerpunkt auf Codierung in C++ setzen**
4. **Die Studenten haben 1. 2. und 3. in Übungen und Beispielen aktiv erfahren**

Zeitplan

Hochschule Deggendorf

- **Donnerstag**
- **11:30 – 13:00 Vorlesung**
- **14:00 – 15:30 Übung (2 Gruppen)**

Sprechstunde Dozent:

- **Mi. 8.00 – 9.30 Uhr (nach Voranmeldung)**
- **auch nach Vereinbarung**
- **vor und nach der Vorlesung**

Fragen

Falls Sie Fragen haben …

fragen Sie bitte

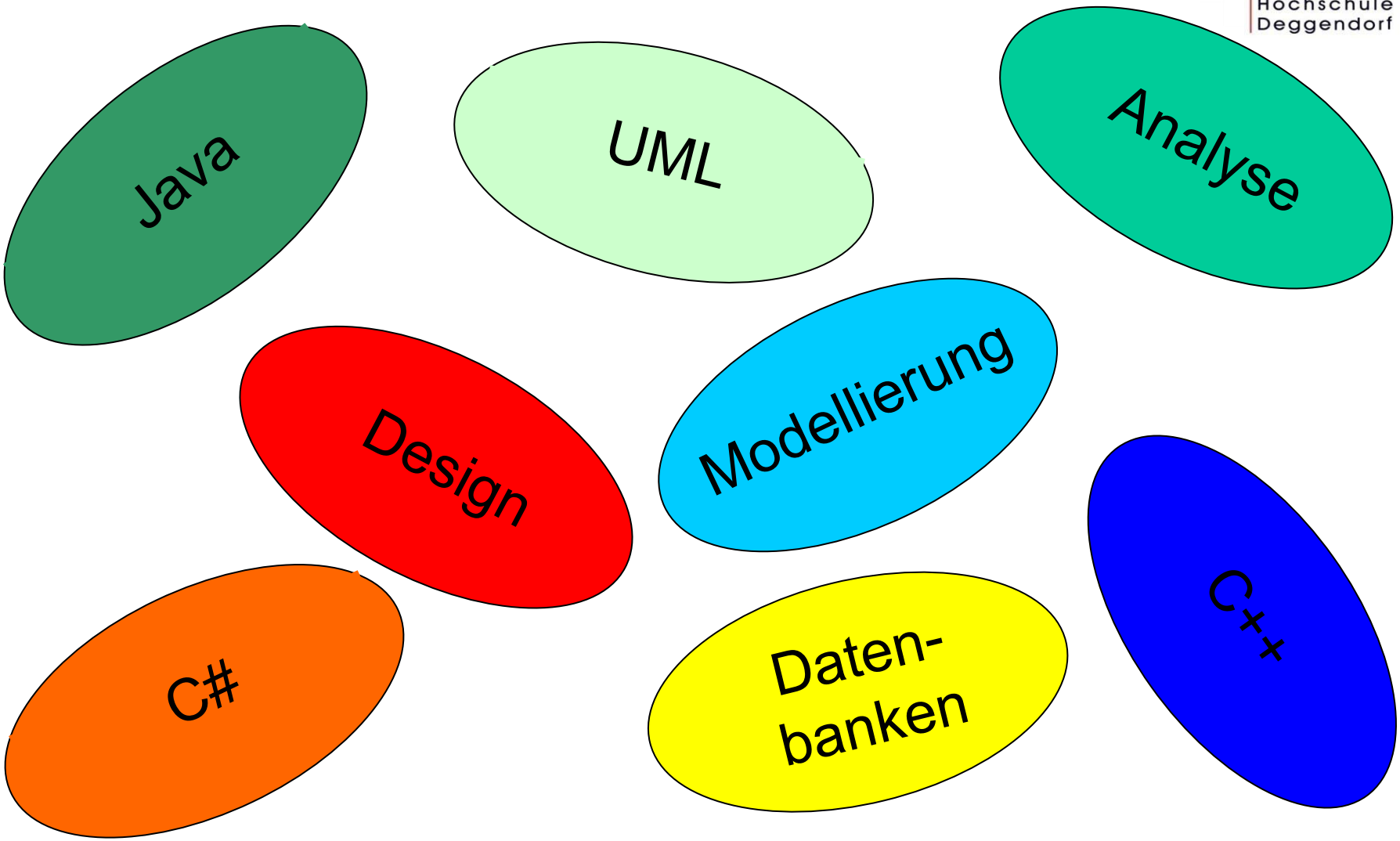
- **sofort**
- **oder vor / nach der Vorlesung**
- **oder in der Sprechstunde**
- **oder nach Terminvereinbarung**

Klausur

Hochschule Deggendorf

- 90 Min.
- Unterlagen erlaubt
- Klausuraufgaben gemäß Vorlesungsübungen und Programmierübungen
- Vorlesungs-, Übungsinhalt und Transfer
- Probeklausur

Objektorientierung



Inhalte

Hochschule Deggendorf

1. Inhalt / Abgrenzung zu anderen Vorlesungen
2. Geschichte / Motivation der OOP / Literatur
3. Methodik – Grundprinzipien und Vergleich mit funktionaler Programmierung
4. Anmerkungen zu C++
5. Grundbegriffe der Objektorientierung
   - Datenkapselung
   - Abstrakten Datentypen
   - Objekt
   - Klasse

Abgrenzung

Hochschule Deggendorf

- **Objektorientierte Programmierung**
  Programmiertechnischen Mittel um objektorientierte Programme zu schreiben (codieren) ➔ **Schwerpunkt dieser Vorlesung**

- **Objektorientierte Analyse**
  Eine auf die Objektorientierung ausgerichtete Methode Probleme zu analysieren ➔ Schwerpunkt der Vorlesung **Software Engineering**

- **Objektorientiertes Design**
  Methoden Objektorientierte Programme zu entwerfen ➔ Schwerpunkt der Vorlesung **Software Engineering**

- **Objektorientierte Datenbanken**
  Methoden Datenbanken objektorientiert zu organisieren

Abgrenzung

- **UML (Unified Modelling Language)**
  teil-formale, weitgehend objektorientierte Modellierungssprache für Software(systeme), verwendet zur Beschreibung von Anforderungen, Analyse und Design. Aus UML Modellen kann z.T. direkt lauffähiger Code generiert werden ➔ Schwerpunkt der Vorlesung **Software Engineering**

- **C#**
  Von Microsoft ab 2001 entwickelte objektorientierte Programmier-sprache basierend auf Konzepten von C++, Java, SQL und Delphi

Abgrenzung

- **Java**
  - Von Sun Microsystems in den 90er Jahren entwickelte objektorientierte Programmiersprache als ein Bestandteil der Java-Technologie.
  - Java-Programme werden nicht in Maschienencode, sondern in Bytecode übersetzt. Dieser wírd dann in einer speziellen Umgebung ausgeführt, die als Java-Laufzeitumgebung oder Java-Plattform bezeichnet wird.
  - Der wichtigster Bestandteil der Java-Laufzeitumgebung wiederum ist die Java Virtual Machine (Java-VM), die den Bytecode interpretiert und nur bei Bedarf kompiliert.
  - Java-Programme sind damit plattformunabhängig, das heißt sie laufen in aller Regel ohne weitere Anpassungen auf verschiedenen Computern und Betriebssystemen, für die eine Java-VM existiert.

Geschichte

Hochschule Deggendorf

- **Erste Ansätze in den 70er Jahren des vorigen Jahrhunderts:**
  - Abstrakte Datentypen
  - Programmiersprache Simula (Dahl, Nygaard)
- **Durchbruch in den 80er Jahren:**
  - Programmiersprachen Smalltalk, Eiffel und insbesondere C++ (als objektorientierte Erweiterung der weit verbreiteten Sprache C
- **Weitere Verbreitung in den 90er Jahren: Internet**
  - Java, C#

Geschichte

- **Heute ist die Objektorientierung *die* Methode (Programmierstil, Programmierparadigma), komplexe Softwaresysteme zu entwerfen und zu beherrschen.**

- **Sehr weit verbreitet in der Entwicklung**
  - PC-basierter Software
  - zunehmend auch für eingebettete Echtzeitsystem

    (z.B.Automobilelektronik)

Geschichte

Hochschule Deggendorf

- **Warum ein neues Programmierparadigma?**

Geschichte

Hochschule Deggendorf

- **Erwartungen an die Objektorientierung**
  - „näher“ an der menschlichen Denkweise
  - bessere „Problemnähe“, d.h. Lösung in der Software spiegelt das reale Problem wider
  - bessere Strukturierung
  - bessere Wiederverwendbarkeit von Software
  - leichtere Verständlichkeit

➔Reduktion von Entwicklungs-, Wartungs- und Fehlerkosten

Geschichte

Hochschule Deggendorf

- **Nachteile**
  - (geringfügig) höherer Resourcenverbrauch (Speicherplatz und Laufzeit)

  ➔ bei Einsatz in resourcenkritischen Umgebungen (z.B. Eingebettete Echtzeitsysteme im Auto) können unter Umständen nicht alle Möglichkeiten der Objektorientierung genutzt werden.

Literatur Objektorientierte Programmierung

Hochschule Deggendorf

... gibt`s wie Sand am Meer ...

Literatur Objektorientierte Programmierung

Bernhard Lahres, Gregor Raýman: Praxisbuch Objektorientierung. Galileo Computing, ISBN 3-89842-624-6 (Frei verfügbar auf der Verlags-Webseite).

Harold Abelson, Gerald Jay Sussman, Julie Sussman: Structure and Interpretation of Computer Programs. The MIT Press, ISBN 0-262-01153-0.

Heide Balzert: Objektorientierte Systemanalyse. Spektrum Akademischer Verlag, Heidelberg 1996, ISBN 3-8274-0111-9.

Grady Booch: Object-Oriented Analysis and Design with Applications. Addison-Wesley, ISBN 0-8053-5340-2.
Peter Eeles, Oliver Sims: Building Business Objects. John Wiley & Sons, ISBN 0-471-19176-0.

Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides: Design Patterns: Elements of Reusable Object Oriented Software. Addison-Wesley, ISBN 0-201-63361-2.

Paul Harmon, William Morrissey: The Object Technology Casebook. Lessons from Award-Winning Business Applications. John Wiley & Sons, ISBN 0-471-14717-6.

Ivar Jacobson: Object-Oriented Software Engineering: A Use-Case-Driven Approach. Addison-Wesley, ISBN 0-201-54435-0.
Bertrand Meyer: Object-Oriented Software Construction. Prentice Hall, ISBN 0-13-629155-4.

Bernd Oestereich: Objektorientierte Programmierung mit der Unified Modeling Language. Oldenbourg, ISBN 3-486-24319-5.

James Rumbaugh, Michael Blaha, William Premerlani, Frederick Eddy, William Lorensen:
Object Oriented Modeling and Design. Prentice Hall, ISBN 0-13-629841-9.

Geschichte

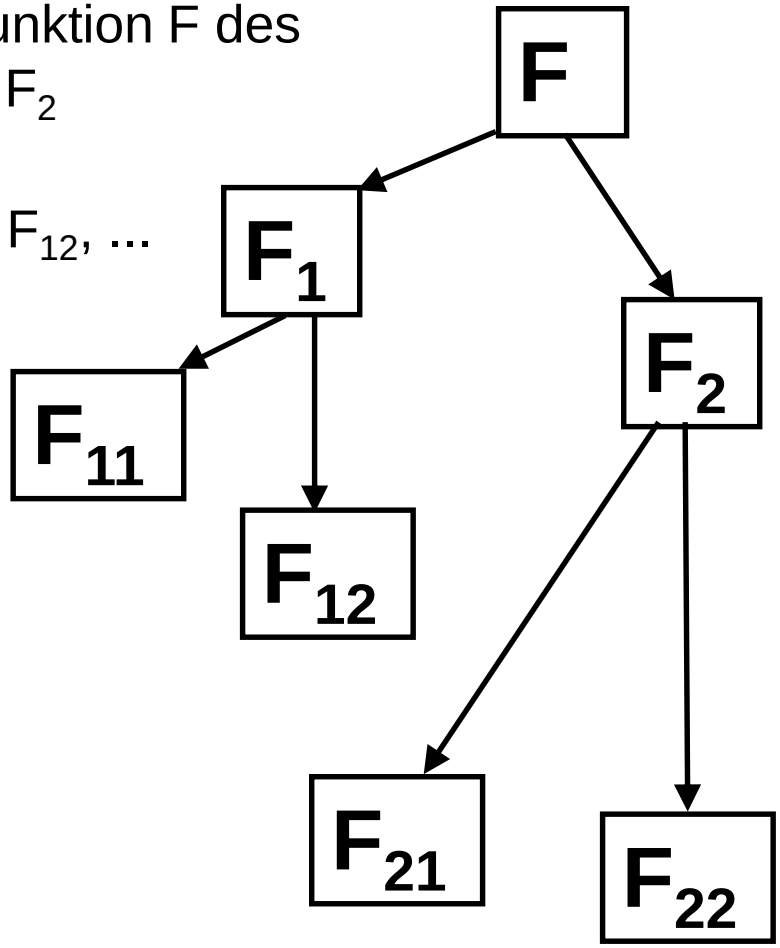
Hochschule Deggendorf

Zum Schluss dieses Abschnitts …

Noch Fragen ??

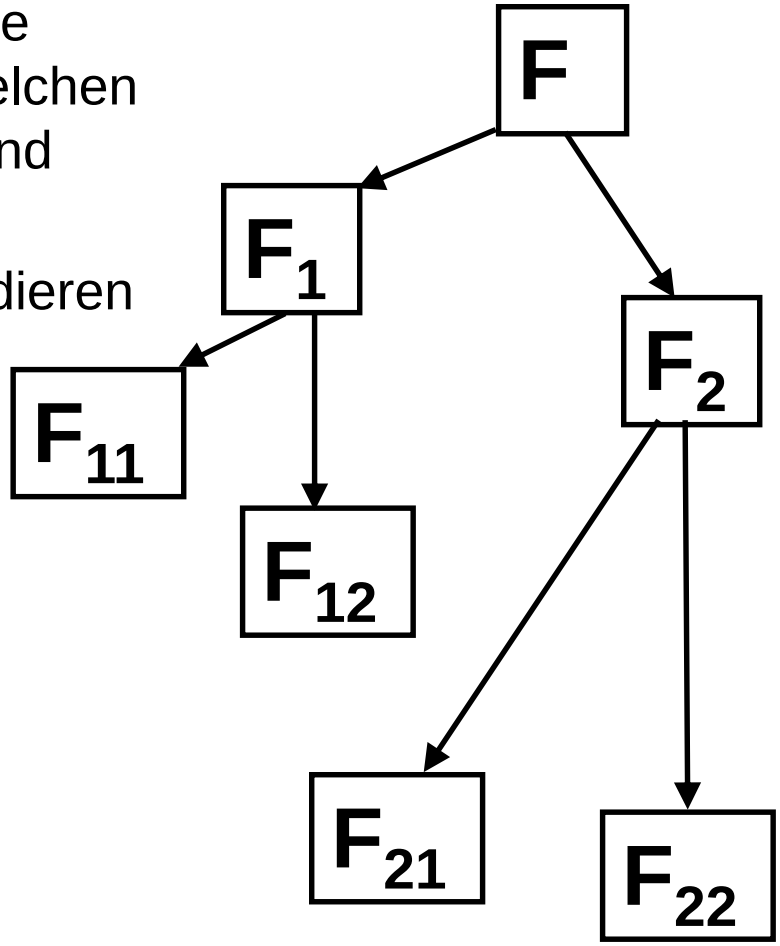
Methodik – Funktionale Programmierung

- **Programm berechnet aus Eingabedaten ein Ausgabe**
  - schrittweise Zerlegung der Gesamtfunktion F des Programms in Teilfunktionen F_1 und F_2
  - Teilfunktionen werden wiederum zerlegt in weitere Teilfunktionen F_{11}, F_{12}, ...
  - „Atomare“ (nicht weiter zerlegte) Teilfunktionen werden codiert
  - Übergeordnete Funktionen rufen atomare Teilfunktionen auf
  - Gesamtfunktion ruft integrierte Teilfunktionen auf



Methodik – Funktionale Programmierung

- **Designansatz (vereinfacht)**
  - in welche Teilfunktionen lässt sich die Gesamtfunktion strukturieren, auf welchen Daten arbeitet die Gesamtfunktion und die Teilfunktionen?
  - Lassen sich Teilfunktionen direkt codieren oder ist eine weitere Zerlegung in Teilfunktionen notwendig?



Methodik – Funktionale Programmierung

Hochschule Deggendorf

Beispiel: Berechnung der Einkommenssteuer

Problembeschreibung:

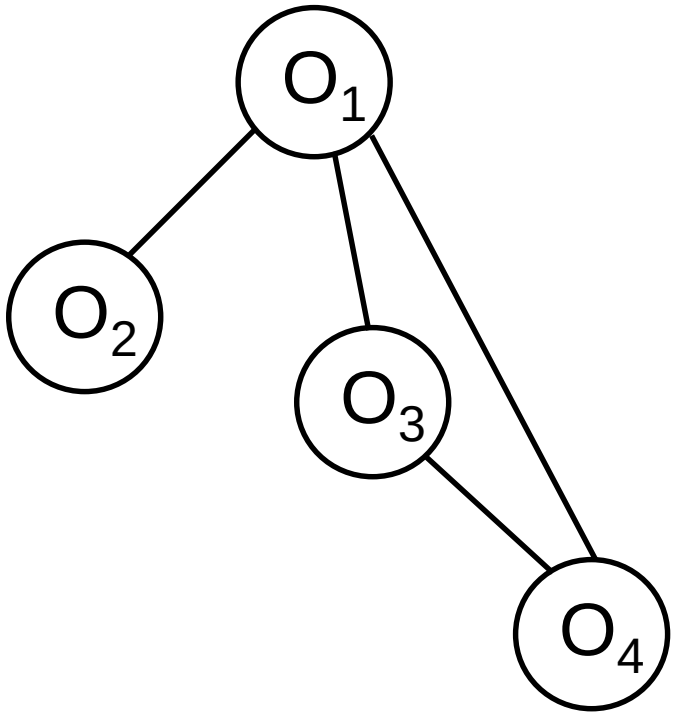
„ ... die Einkommenssteuer setzt sich zusammen aus der normalen Einkommensteuer und dem Solidaritätszuschlag ...“

➔ **Lösung:**

Funktion „Einkommenssteuer“ setzt sich zusammen aus den Funktionen „Normale Einkommenssteuer“ und „Solidaritätszuschlag“. Als Ergebnis der Funktion wird die Summe der Teilfunktionen gebildet.

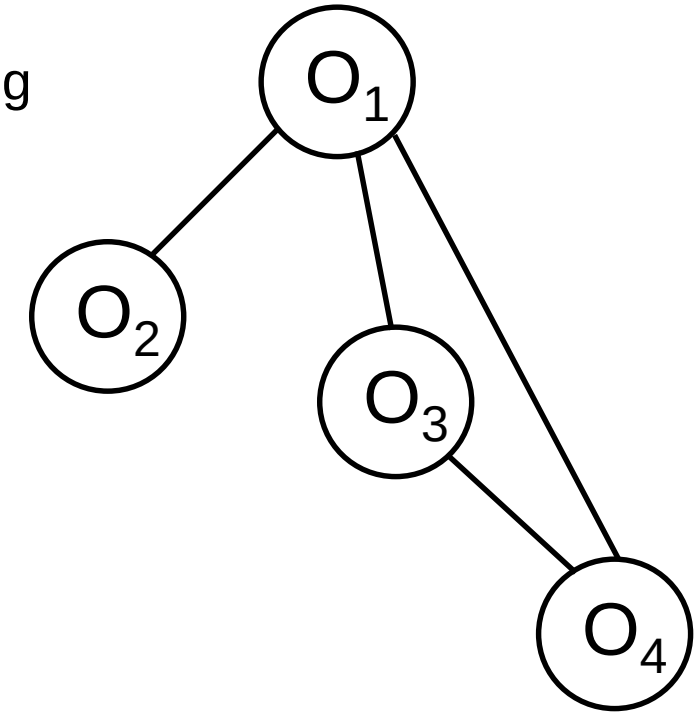
Methodik – Objektorientierung

- **Programm berechnet aus Eingabedaten ein Ausgabe**
  - welche Objekte (der Problemwelt) spielen bei der Berechnung der Ausgabe eine Rolle ?
  - wie hängen diese Objekte zusammen ?
  - wann entstehen diese Objekte ? (gleichzeitig, nacheinander)
  - wie ist die Lebensdauer der Objekte ? (permanent, temporär)
  - was ist die Kardinalität der Objekte ? (gibt es nur eins, mehrere, unendlich viele)
  - wie arbeiten diese Objekte zusammen ?
  - wie lassen sich Objekte klassifizieren ?
  - ...



Methodik – Objektorientierung

- **Designansatz (vereinfacht)**
  - Hauptworte in der Problembeschreibung (Spezifikation) suchen
  - Daraus Kandidaten für Objekte und (später) Klassen identifizieren
  - Beziehungen zwischen Objekten und Klassen festlegen („enthält“, „benötigt“, „kommuniziert“)
  - Klassen codieren
  - Objekte instanziieren und Kommunikation codieren



Methodik – Objektorientierung

Hochschule Deggendorf

Beispiel: Steuerung der Blinkfunktion in einem Kraftfahrzeug.

Problembeschreibung:
„ ... der linke Blinker soll blinken, wenn der Blinkhebel nach unten gedrückt wird ... „

➔ **Lösung:**
Objekte: Blinkhebel, linker_Blinker, Blinkhebel kommuniziert mit dem Objekt linker_Blinker

Übung

Einsatzfelder der Objektorientierung

Beschreibung:
Viele Programme lassen sich mittels eines objektorientierten Ansatzes leichter implementieren als mit einem rein funktionalen Ansatz

Aufgabe:
1.Überlegen Sie, in welchen Problemfeldern ein funktionaler Ansatz berechtigt sein kann und warum!
2.In welchen Problemfeldern könnte ein objektorientierter der bessere sein und warum?

Zeit:
10 Minuten, arbeiten Sie ggf. zusammen mit einem Partner