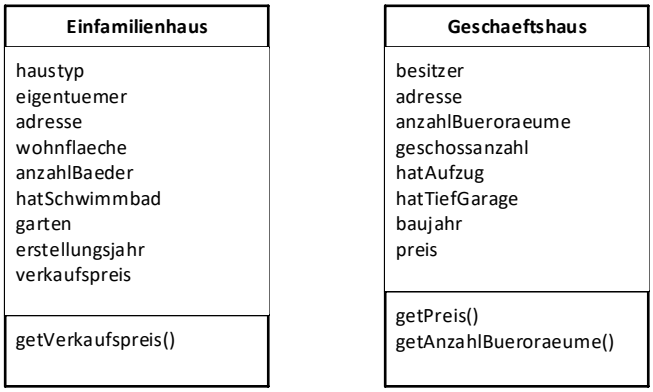
Programmieren 2 – Live Übung

Vererbung

Aufgabe 1: Generalisieren – Gemeinsamkeiten erkennen

Für die Immobilienfirma Exklusiv & Teuer soll ein Softwaresystem entwickelt werden. Eine erste Analyseaktivität ergab folgende zwei Klassen:



a) Vergleichen Sie die beiden Klassen. Was fällt Ihnen auf? Achten Sie dabei nicht nur auf die Namen, sondern auf die Semantik, die dahinter steckt.

b) Entwerfen Sie eine Vererbungsstruktur, indem Sie eine Oberklasse für die beiden Klassen mit geeigneten Namen finden. Finden Sie dazu sowohl für die Oberklasse als auch für die beiden entstehenden Unterklassen geeignete Namen für die Attribute und Methoden.

Aufgabe 2: Umsetzung der Klassen

Implementieren Sie die die drei Klassen Ihrer Vererbungsstruktur aus Aufgabe 1. Berücksichtigen Sie dabei nur eine Auswahl der oben angeführten Attribute. Folgender Programmablauf sollte möglich sein:

```
Daten zum Einfamilienhaus
Haustyp: Stadthaus
Besitzer: Müller
Verkaufspreis: 300000
Daten zum Geschäftshaus
Anzahl Büroräume: 10
Besitzer: Winter
Verkaufspreis: 1500000
```

Aufgabe 3: Konstruktoren

Schreiben Sie zu den drei Klassen die jeweils passenden Konstruktoren. Berücksichtigen Sie dabei folgende Aspekte:

- Der Konstruktor der Oberklasse soll zwei Aufrufparameter besitzen, nämlich der Besitzer sowie der Verkaufspreis.
- Dem Konstruktor der Unterklasse Einfamilienhaus soll drei Aufrufparameter haben, nämlich der Besitzer, der Verkaufspreis sowie der Haustyp.
- Der Konstruktor der Unterklasse Geschäftshaus soll drei Aufrufparameter besitzen, nämlich der Besitzer, der Verkaufspreis sowie die Anzahl der Büroräume.

Aufgabe 4: Abstrakte Klasse

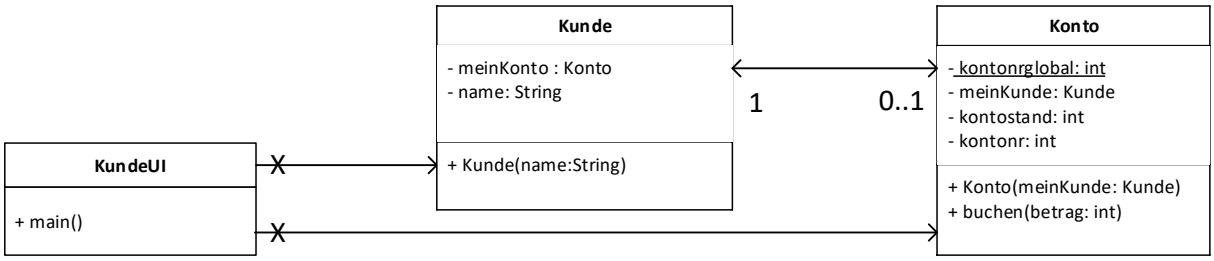
Berücksichtigen Sie folgenden Aspekt in der Implementierung: Bei der Immobilienfirma Exklusiv & Teuer ist jede Immobilie entweder ein Einfamilienhaus oder ein Geschäftshaus. Daher ist es nicht sinnvoll von der Oberklasse Objekte zu erzeugen. Sie soll als abstrakte Klasse gekennzeichnet werden.

Aufgabe 5: Anwendungsklasse schreiben

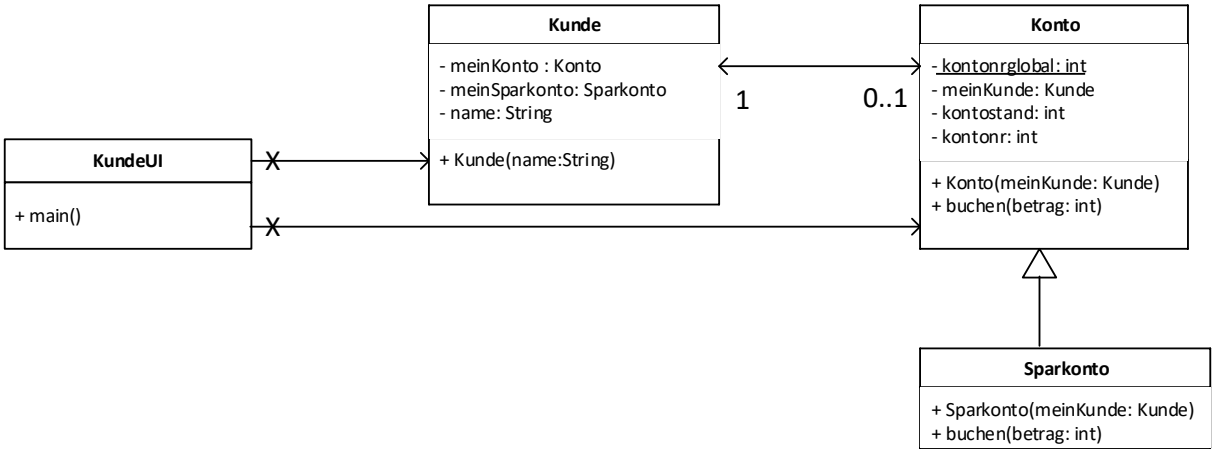
Schreiben Sie abschließend eine Anwendungsklasse, die den in Aufgabe 2 beschriebenen Testfall simuliert.

Aufgabe 6: Methoden redefinieren

Eine Bank verwaltet bisher ihre Kunden mit einem Programm. Jeder Kunde kann ein Konto besitzen. Ein Konto ist dadurch gekennzeichnet, dass der Kunden sowohl einen positiven als auch negativen Kontostand besitzen kann. Bei der Realisierung wurde folgendes Klassendiagramm umgesetzt:



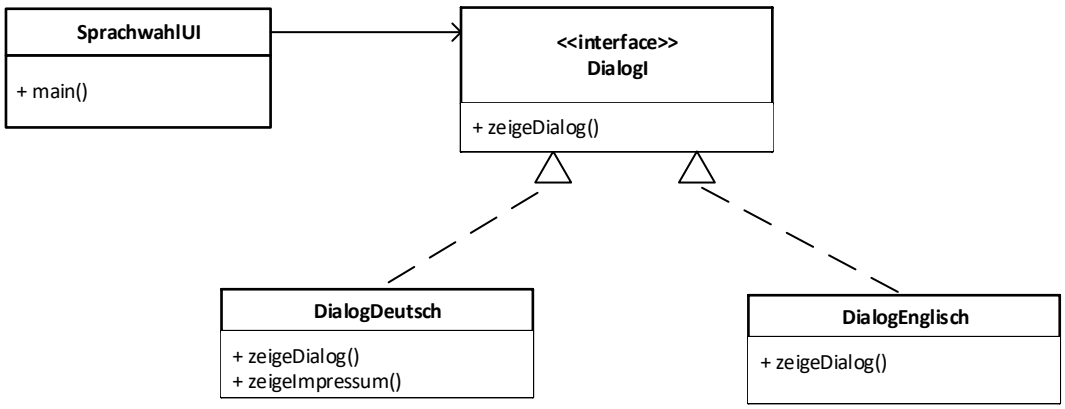
Es sollen jetzt mit demselben Programm zusätzlich Sparkonten verwaltet werden. Jeder Kunde kann neben einem Konto auch ein Sparkonto besitzen – oder auch nur ein Sparkonto. Eine Analyse der Klasse Konto ergibt, dass Sparkonto über dieselben Attribute und Operationen wie ein Konto verfügt. Allerdings darf der Kontostand eines Sparkontos nie negativ sein. Deshalb muss die Operation buchen() redefiniert bzw. überschrieben werden. Folgendes Klassendiagramm veranschaulicht die Zusammenhänge:



Schreiben Sie die Klasse `Sparkonto`.

Aufgabe7: Schnittstellen

Es soll ein Programm zur Sprachauswahl geschrieben werden, bei dem die Benutzungsoberfläche auf verschiedene Sprachen eingestellt werden kann. Zunächst sollen die Sprachen Deutsch und Englisch wählbar sein. Später sollen aber noch weitere Sprachen hinzugefügt werden, ohne das Programm neu schreiben zu müssen. Das Klassendiagramm zeigt das konzipierte Schnittstellenkonzept:



Setzen Sie die folgenden Klassen um:

1. Klasse, die das Interface `DialogI` repräsentiert
2. Klasse `DialogDeutsch`
   (Methode `zeigeDialog` gibt den Text „Herzlich willkommen“ auf dem Bildschirm aus. Methode `zeigeImpressum` gibt den Text „Impressum“ auf dem Bildschirm aus.)
3. Klasse `DialogEnglisch`
   (Methode `zeigeDialog` gibt den Text „Welcome“ auf dem Bildschirm aus.)