

Übung 10

Vererbung und Polymorphie

A 10.1: Uni-Hierarchie

(6)

- (a) Schreiben Sie eine C++ - Klasse `Person`, die allgemeine Informationen über eine Person kapselt: Name, Vorname, Geschlecht und Alter. Implementieren Sie eine Methode `gebeAus()`, die diese Informationen am Bildschirm anzeigt.
- (b) Leiten Sie von der Klasse `Person` eine Klasse `Student` ab. Diese soll zusätzlich die Datenelemente `Matrikelnummer`, `Studienfach` und `Semesterzahl` enthalten. Überladen Sie die Methode `gebeAus()`, so daß sie die Informationen der Basisklasse und die zusätzlichen Informationen ausgibt.

Verwenden Sie zur Ausgabe der Basisinformationen (also Name, Vorname...) die Methode `gebeAus()` aus der Klasse `Person`.

- (c) Leiten Sie von der Klasse `Person` eine Methode `Professor` ab. Diese soll zusätzlich die Datenelemente `Fachgebiet` und `Gehalt` enthalten. Überladen Sie auch hier die Methode `gebeAus()` entsprechend der Beschreibung in (b).
- (d) Erzeugen Sie verschiedene Instanzen der Klassen `Student` und `Professor` und geben Sie diese aus.

A 10.2: Arrays & virtuelle Methoden

(6)

In dieser Aufgabe soll ein Array von Personen erstellt werden, wobei eine `Person` ein `Student` oder ein `Professor` sein kann (vgl. A10.1). Auf den ersten Blick ergibt sich folgendes Problem: Das Array muß einen eindeutigen Typ haben (so wie z.B. ein Array von Integers den Typ `int` hat und somit keine `double`-Zahlen speichern kann), wir wollen aber zwei verschiedene Typen (eben `Studenten` und `Professoren`) in dem Array speichern. Wie zu erwarten liegt die Lösung des Problems in der gemeinsamen Vater-Klasse `Person`: Das Array ist vom Typ `Person*`, `Student` und `Professor` sind sozusagen Unter-Typen und können somit ohne Problem in dem Array gespeichert werden. Was ist aber, wenn wir einen Eintrag des Arrays auslesen wollen. Zunächst wissen wir nur, daß der Eintrag vom Typ `Person` ist, können aber keine Aussage darüber machen, ob es sich um einen `Studenten` oder einen `Professor` handelt.

- (a) Schreiben Sie ein C++ - Programm, in dem Sie ein Array vom Typ `Person` mit 10 Einträgen (`Studenten` und `Professoren`) anlegen..
- (b) Implementieren Sie den korrekten Aufruf der `gebeAus()` – Methode mittels virtueller Methoden.

A 10.3: Formeln

(8)

In dieser Aufgabe soll eine Klassenhierarchie zur Auswertung von mathematischen Formeln (die von genau einer Variablen x abhängen) entwickelt werden. Die Formel soll dabei fest implementiert sein, d.h. nicht durch einen String o.ä. initialisiert werden. Dazu soll zunächst eine Basisklasse `Formel` implementiert werden, die die triviale Formel $y = x$ kapselt. Danach soll von dieser Klasse eine Klasse `Polynom`, die eine bestimmte polynomiale Formel kapselt, abgeleitet werden.

- (a) Schreiben Sie eine C++ - Klasse `Formel`, die die triviale Formel $y = x$ kapselt. Implementieren Sie eine Methode `double berechne(double x)`, die den Funktionswert an der Stelle x zurückgibt (d.h. in der Basisklasse soll einfach nur x zurück gegeben werden). Implementieren Sie weiterhin eine Methode `Wertetabelle(double xMin, double xMax, int numSteps)`, die eine *Wertetabelle* der Formel im Intervall $[xMin, xMax]$ in `numSteps` Schritten ausgibt und dazu die Methode `berechne(...)` aufruft.
- (b) Leiten Sie eine Klasse `Polynom` von `Formel` ab, die eine polynomiale Formel (also Formeln der Form $y = a_0 + a_1x + a_2x^2 + a_3x^3 + \dots$) kapselt. Überschreiben Sie ausschließlich den Konstruktor und die Methode `berechne(...)`. Die Methode `berechne(...)` soll nun den Wert des Polynoms $2x^3 - x^2 + 7x$ zurückgeben.
- (c) Geben Sie sowohl für eine Instanz der Klasse `Formel` als auch für eine Instanz der Klasse `Polynom` eine *Wertetabelle* im Intervall $[0, 1]$ in 11 Schritten aus. Es sollte die *richtige Wertetabelle* ausgegeben werden? (`Wertetabelle(...)` sollte nicht überladen werden !)