Fachhochschule WORMS
University of Applied Sciences
Fachbereich Informatik
Prof. Dr. Bernd Ruhland

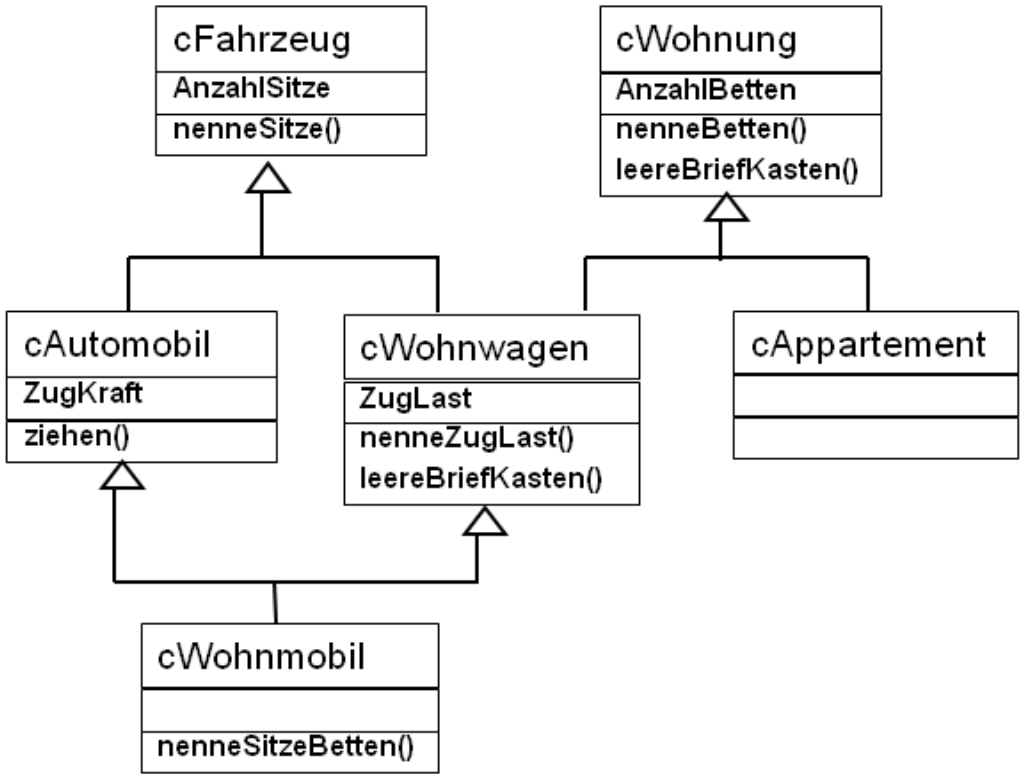
Programmieren 2 C++ / D3 Objekte

Praktikumsaufgabe u02b Vererbung / Ableitungen zum Wohnmobil
(Übungsaufgabe im Praktikum)

Thema: Vererbung und Konstruktoren, Mehrfachvererbung

Quelle: bild.de

Setzen Sie folgendes Klassendiagramm in ein C++-Programm um:



- Gehen Sie systematisch vor !
- Legen Sie zunächst alle Klassendefinitionen leer an.
- Stellen Sie dann die Ableitungen her (Vererbungen).
- Definieren Sie danach die Mitgliedsvariablen.
- Bauen Sie dann die Konstruktoren (es reicht jeweils 1 Konstruktor je Klasse, verwenden Sie Vorgabewerte.
- Kaskadieren Sie die Konstruktoren durch Aufruf des Konstruktors der jeweiligen Basisklasse (Beispiel dazu s.u. als Hinweis).
- Implementieren Sie dann die Methoden der Klassen.
- Erstellen Sie ein Hauptprogramm, in welchen der Reihe nach mit jeweils einer Instanz der Klassen gearbeitet wird (wird genauer spezifiziert s.u.).

Die Klassen im Einzelnen:

cFahrzeug:
- Bekommt als Vorgabewert: Anzahl der Sitze 4.
- Hat folgende Methoden, die von außen aufgerufen werden können:

```
int  nenneSitze();              // Liefert den Wert der Variablen AnzahlSitze
```

cWohnung:
- Bekommt als Vorgabewert: Anzahl der Betten 6.
- Hat folgende Methoden, die von außen aufgerufen werden können:

```
int  nenneBetten();             // Liefert den Wert der Variablen AnzahlBetten

void leereBriefKasten();        // Gibt den Satz aus „Die Post ist da“
```

cAutomobil:
- Bekommt als Vorgabewert: Zugkraft 1.32 t, Anzahl der Sitze 5.
- Hat folgende Methoden, die von außen aufgerufen werden können:

```
bool  ziehen(cWohnwagen w);     // Entscheidet, ob die Zugkraft reicht
```

  - Ermittelt die Zuglast des Wohnwagens
  - Ist die Zugkraft ausreichend, dann gibt ziehen() ein true zurück, sonst false

cWohnwagen::
- Bekommt als Vorgabewert: Zuglast 0.92 t.
- Hat folgende Methoden, die von außen aufgerufen werden können:

```
double  nenneZugLast();         // Liefert den Wert der Variablen ZugLast

void leereBriefKasten();        // Überschreibt die Methode der Basisklasse !
                                // Gibt den Satz aus „Keine Post, nur postlagernd“
```

cAppartement:
- Bekommt als Vorgabewert: Anzahl der Betten 2.

cWohnmobil:

- Bekommt als Vorgabewerte: Anzahl Sitze 3, Anzahl Betten 3, Zugkraft 0.7, Zuglast 0.0
- Hat folgende Methode, die von außen aufgerufen werden kann:

void nenneSitzeBetten(); // Gibt die Werte seiner Sitze und Betten aus

Das Hauptprogramm:

Arbeiten Sie mit den von Ihnen erstellten Klassen:

- Erstellen Sie folgende Instanzen:

cAutomobil	**Mustang;**
cWohnwagen	**WanderVogel;**
cWohnmobil	**UschiBus (2, 2);**
cAppartement	**MyStudio (1);**

- Verwenden Sie folgende Anweisungen, um die Funktion Ihrer Klassen und Methoden nachzuweisen (können Sie per cut&paste kopieren):

```
// Mit den Objekten spielen:

cout << "Auto zieht Wohnwagen ?      (0/1): " << Mustang.ziehen(WanderVogel)
                                                 << endl;
cout << "Wohnmobil zieht Wohnwagen ? (0/1): " << UschiBus.ziehen(WanderVogel)
                                                    << endl << endl;
cout << "Ausstattung UschiBus: "; UschiBus.nenneSitzeBetten();
cout << "Briefkasten UschiBus: "; UschiBus.leereBriefKasten();
cout << "Briefkasten MyStudio: "; MyStudio.leereBriefKasten();
cout << "Das MyStudio hat " << MyStudio.nenneBetten() << " Betten."
                                                 << endl << endl;
cout << " *** ENDE *** " << endl << endl;
```

Hinweis: Konstruktorenaufrufe lassen sich kaskadieren wie folgt:

```
class AT  {                     // Basisklasse
     int i;
public:
     AT(int i_in) { i = i_in; }  // Konstruktor AT
};

class BT : public AT  {          // abgeleitete Klasse
     int k;
public:
     BT (int i_in, int k_in) : AT (i_in)  { k = k_in; }    // Konstruktor BT mit
                                                           // Kaskadierung AT
};


Int main () {
     BT b (7, 8);   // Definition Objekt b im Hauptprogramm

    // ...
}
```
