Kapitel 17

Klassen und Objekte

Prozedurale Programmierung

- Die Prozedurale Programmierung trennt die Daten von den Funktionen, die diese Daten bearbeiten.
- Das hat zur Folge, dass ein Programmierer selbst für die Initialisierung von Daten vor ihrer Verwendung sorgen muss und dass beim Aufruf einer Funktion korrekte Daten übergeben werden.
- Wird die Darstellung der Daten geändert, müssen auch die entsprechenden Funktionen angepasst werden.
- Ein solches Vorgehen erhöht den Wartungsaufwand und ist fehleranfällig.

Datenabstraktion

- Ziel der „Objektorientierten Programmierung“, auch kurz OOP, ist es, komplexe Sachverhalte als Ganzes abzubilden, d.h. die Eigenschaften (Daten, Attribute) und die darauf wirkenden Methoden (Funktionen, Fähigkeiten) zusammenzufassen.
- Die Objekte der OOP bilden eine Einheit aus Eigenschaften und Methoden.
- Eigenschaften und Methoden einer Klasse nennt man auch Member einer Klasse.

Datenabstraktion

- Vor der Implementierung werden in einem Komponentenentwurf die Objekte herausgearbeitet. Mehrere gleichartige Objekte werden dann zu einer Klasse zusammengefasst.
- Dabei werden alle Gemeinsamkeiten der gleichartigen Objekte herausgearbeitet und in einer Klasse vereinigt.
- Die Bildung von Klassen zur Beschreibung von Objekten nennt man auch Datenabstraktion.

Klassen und Objekte

- Eine Klasse ist ein selbst definierter Datentyp und ein Objekt ist eine Variable (Instanz) dieser Klasse.
- Eine Klasse *CKonto* enthält z.B. die Eigenschaften:
  - Name des Kontoinhabers
  - Kontonummer
  - Kontostand

Klassen und Objekte

- Ferner enthält die Klasse *CKonto* die Methoden:
  - Name abfragen und ändern
  - Kontonummer abfragen und ändern
  - Kontostand abfragen und ändern
  - Kontodaten ausgeben
- Diese Eigenschaften und Methoden gehören zu jedem beliebigen Konto.
- Im folgenden Beispiel (Bsp: CKonto_Schnittstelle.zip) ist die Schnittstelle der Klasse *CKonto* definiert.

Deklaration der Klasse *CKonto*

```
class CKonto
{
private:
    string m_name                  // Name des Kontoinhabers
    int m_number                   // Kontonummer
    double m_stand                 // Kontostand
public:
    string getName();                      // Name abfragen
    void setName(string name);             // Name ändern
    int getNumber();                       // Kontonummer abfragen
    void setNumber(int number);            // Kontonummer ändern
    ...                                    // Kontostand abfragen / ändern
    Ausgabe();                             // Kontodaten ausgeben
};
```

Deklaration einer Klasse - Allgemein

- Jede Klasse besteht aus einer Menge von Eigenschaften und Methoden.
- Eigenschaften und Methoden verschiedener Klassen dürfen die gleichen Namen tragen.
- Jede Klasse wird beschrieben durch eine .cpp- und eine .h-Datei.
- Die Deklaration (die Beschreibung der Schnittstelle) erfolgt normalerweise in der .h-Datei.

Deklaration einer Klasse - Allgemein

- Die Eigenschaften und Methoden einer Klasse werden in einem Block zusammengefasst.
- Vor diesem Block steht das Schlüsselwort *class* gefolgt vom Klassennamen. Am Ende des Blocks **muss** ein Semikolon stehen.

```
class CKonto
{
        ...
};
```

Definition einer Klasse

- Man beachte, dass die Methoden bisher nur deklariert wurden.
- Die Definition der Klasse bzw. der einzelnen Methoden erfolgt in der .cpp-Datei.
- Jede Quelldatei kann jetzt auf die Klasse *CKonto* zugreifen, indem sie *Konto.h* inkludiert.
- Dies ist eine Grundvoraussetzung für die modulare Programmierung, also die Aufteilung des Programmcodes auf mehrere Module (Dateien).

Datenkapselung

- Die Eigenschaften einer Klasse sind bei guter Programmierung nur über die Methoden veränderbar.
- Die Menge aller Methoden einer Klasse nennt man auch Schnittstelle der Klasse.
- Die Schnittstelle (siehe: Deklaration) einer Klasse muss sich nicht ändern, wenn sich die interne Darstellung der Daten ändert.
- Das hat den Vorteil, dass alle die Klasse verwendenden Programmteile unverändert bleiben können.

Definition eines Objekts

- Ist die Schnittstelle der Klasse *CKonto* bekannt, können Objekte dieser Klasse definiert werden:

```
CKonto konto;
```

- Man sagt auch: *konto* ist ein Objekt der Klasse *CKonto*.

- Die .cpp/.h-Datei ist sozusagen der Bauplan, nachdem die Objekte erstellt werden.

Definition eines Objektfeldes

- Es ist auch möglich ein Feld von Objekten zu definieren:

```
CKonto konto[10];
```

Definition der Methoden einer Klasse

- Die Methoden werden üblicherweise in einer cpp-Datei (z.B. Konto.cpp) definiert (Bsp: CKonto_Komplett.zip):

```
#include "Konto.h"
string CKonto::GetName()
{
    return m_name;
}
void CKonto::SetName(string name)
{
    m_name = name;
}
...
```

Definition der Methoden einer Klasse

- Es ist auch möglich, die Definition der Methoden direkt in die Klassendefinition einzubeziehen. Die Datei *Konto.cpp* kann hierbei entfallen. (Bsp: CKonto_InEinerDatei.zip):

```
...
public
    string GetName()
    {
        return m_name;
    }
...
```

Aufruf der Methoden einer Klasse

- *Funktion* wird in der objektorientierten Programmierung *Methode* genannt.
- Der Aufruf der Methode einer Klasse erfolgt immer für ein bestimmtes Objekt der Klasse.
- Der Name des Objekts muss durch einen Punkt getrennt vor den Namen der Methode geschrieben werden.

Aufruf der Methoden einer Klasse

```
// Klasse string und Objekt error
void main(void)
{
    meineKlasse fehler;    // fehler ist ein Objekt der Klasse meineKlasse.
                           // Die Klasse meineKlasse hat eine Methode
                           // namens ausgabe, welche den Fehler
                           // ausgibt.
    fehler.ausgabe();      // Aufruf der Methode ausgabe in der
                           // Klasse meineKlasse.
}
```

Zugriffsrechte bei Klassen

- Allen Membern (Eigenschaften, Methoden) einer Klasse sind Zugriffsrechte zugeordnet.
- Die Zugriffsrechte definieren, ob auf ein Member von außen zugegriffen werden darf oder nicht.
- Der Zugriff von außen kann durch das Schlüsselwort *private* gesperrt werden.
- Das Schlüsselwort *public* erlaubt den Zugriff von außen.

```
public:
    string GetName();
```

Zugriffsrechte bei Klassen

- Ist kein Zugriffsrecht angegeben, ist es standardmäßig *private*.
- Methoden einer Klasse haben generell immer Zugriff auf alle anderen Member der eigenen Klasse.
- Die Schlüsselwörter (*public*, *private*) sind innerhalb einer Klasse gültig, bis sie durch ein anderes verändert werden.
- Die Reihenfolge und Anzahl von *private* bzw. *public* ist beliebig.
- Alle Eigenschaften einer Klasse sollten im *private* -Bereich der Klasse stehen.

Zugriff auf Member einer Klassen

- Auf ein *public* Member einer Klasse kann von extern (z.B. von der main -Funktion aus) direkt zugegriffen werden.
- Auf Member einer Klasse kann nur über ein Objekt der Klasse direkt zugegriffen werden.
- Der Zugriff auf Member erfolgt durch die Angabe des entsprechenden Objekts, gefolgt vom Punktoperator und dem Namen des Members.

```
CKonto konto;
konto.SetName("Hans Meier");
konto.SetNumber(123456);
konto.Ausgabe();
```