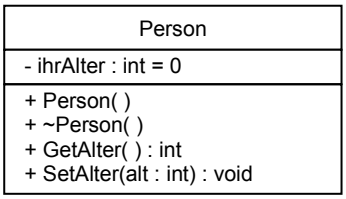
Erstellen von Klassen und Objekten in C++

Nach dem Design erfolgt in der Realisierungsphase die Implementierung und der Test der Klassen und Objekte.
Die Klasse als Softwareeinheit wird anhand des Klassendiagramms gängigerweise in zwei Dateien codiert. Die Definition der Klasse findet in der gleichnamigen Headerdatei, die Methodendeklarationen finden in der zugehörigen Quelltextdatei statt.

Klassen in C++

Als Beispiel für eine Klassendefinition in C++ soll die Klasse Person mit einem Attribut und zwei Zugriffsmethoden dienen.



Die **Headerdatei** beginnt mit dem Schutz gegen Mehrfacheinbindung durch eine Präprozessorkonstante. Als Standard für diese Konstante hat sich der Klassenname in GROSSBUCHSTABEN durchgesetzt.
Nach dem Schutz gegen Mehrfacheinbindung wird die Klasse Person mit ihren öffentlichen Methoden und der privaten Eigenschaft ihrAlter implementiert.

Gefolgt von dem Schlüsselwort class steht der Bezeichner der Klasse. Alle Deklarationen im folgenden Anweisungsblock gehören zur Klasse. Die Schlüsselworte public und private stellen Schutzattribute dar. Alle öffentlichen Teile der Klasse sind nach Außen hin sichtbar (Schnittstellen), auf private Elemente können nur Objekte der Klasse selbst zugreifen (Kapselung, Verbergen von Daten). In der UML werden öffentliche Elemente durch ein vorangestelltes + Zeichen, private durch ein - Zeichen dargestellt.
Konstruktor und Destruktor sind Sonderfunktionen der Klasse mit dem Klassennamen als Bezeichner. Sie haben als einzige Funktionen in C++ keinen Rückgabetyp (nicht einmal void!). Konstruktor bzw. Destruktor werden automatisch aufgerufen, wenn ein Objekt gebildet wird bzw. seine Gültigkeit wieder verliert. Im Konstruktor können Attribute initialisiert werden. Der Destruktor dient für Aufräumarbeiten (z.B. Freigeben von reserviertem Speicher). Standardkonstruktor und Destruktor können in diesem Klassendiagramm vernachlässigt werden, da sie keine Parameter übernehmen.
Nach der Deklaration der Methoden werden die (meist) privaten Attribute deklariert. Als Faustregel gilt: Methoden public, Eigenschaften private.
Die Klassendefinition endet mit einem Semikolon und der schließenden Präprozessoranweisung (Schutz gegen Mehrfacheinbindung).

```
// Person.h

#ifndef PERSON
   #define PERSON

class Person
{
public:
   Person( );
   ~Person( );

   int GetAlter( );
   void SetAlter(int alt);

private:
   int ihrAlter;
};
#endif
```

In der zugehörigen **Quelltextdatei** der Klasse ist zunächst die Headerdatei mit der Klassendefinition einzubinden.
Da Konstruktor und Destruktor Methoden sind, werden Sie mit einem Funktionsrumpf definiert. Beide Methoden haben keinen Rückgabewert. Im Konstruktor setzt man den Initialwert für ihrAlter der Person auf 0. Jedes Objekt, das später entsprechend der Klasse (Bauplan) Person gebildet wird, erhält als Startwert für ihrAlter die 0. Der Destruktor hat keine besondere Aufgabe und ist mit einem leeren Funktionsrumpf zu implementieren.
Unterschiedlich zu C ist die Aufnahme des Klassennamens in den Funktionskopf. Da Methoden explizit einer Klasse zugeordnet sind, muss bei jeder Methodendefinition der Klassenname gefolgt vom Bereichsoperator (::) angegeben werden. Der Compiler erkennt die Methoden somit als zur Klasse zugehörig. Die Definition von Methoden unterscheidet sich ansonsten nicht von der Funktionsdefinition in C.
Die beiden Zugriffsmethoden (Get-... und Set-...) erfüllen ihre Aufgaben und setzen bzw. liefern den Wert von ihrAlter. Da außenstehende Objekte im Programm nicht direkt auf private Elemente der Klasse Person zugreifen können, ist ihrAlter gekapselt.

```
// Person.cpp
#include "Person.h"

Person::Person( )
{
   ihrAlter = 0;
}

Person::~Person( )
{
}

int Person::GetAlter( )
{
   return ihrAlter;
}

void Person::SetAlter(int alt)
{
   ihrAlter = alt;
}
```

Objekte in C++

Ist die Klasse definiert, so kann ein Objekt vom Typ der Klasse in **jeder Datei** instantiiert werden, welche die Headerdatei mit Klassendefinition einbindet. Dazu verwendet man den Konstruktor.

Nach der Angabe des Typs (Klasse) und der Angabe des Bezeichners (dasObjekt) kann man auf öffentliche Methoden und Attribute des Objekts mit dem Punktoperator (.) zugreifen.

Nebenstehendes Beispielprogramm setzt das Alter des Objekts diePerson auf 32 und gibt mit cout das Alter auf der Konsole aus.
cout ist übrigens auch ein Objekt vom Typ ostream aus der ANSI C++ iostream - Bibliothek. Mit Hilfe des überladenen Umleitungsoperators (<<) kann es Werte unterschiedlicher Basisdatentypen von C++ auf die Konsole schreiben.

```
#include <iostream>
#include "Person.h"

using namespace std;

int main( )
{
 Person diePerson;

 diePerson.SetAlter(32);
 cout<<"Alter: "<<diePerson.GetAlter( )<<"\n";

 return 0;
}
```

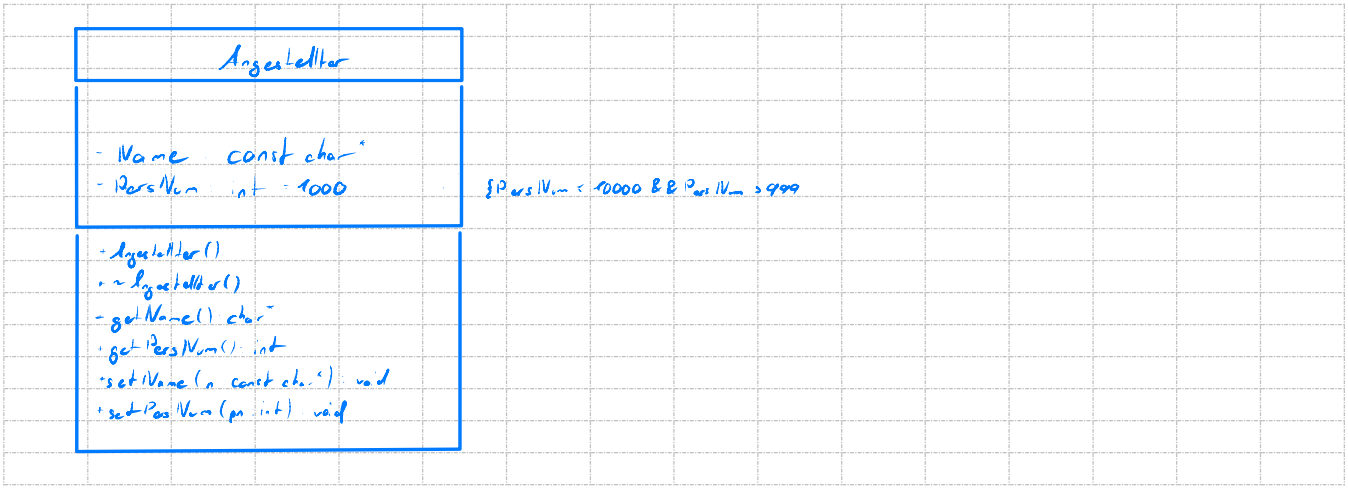
Übung: Design und Realisierung von Klassen in C++

In dieser Übung wird eine Klasse entworfen und implementiert. Anhand von kurzen Beispielen wird die Funktion von Konstruktor, Destruktor, public und private verdeutlicht.

Das Design der Klasse Angestellter

Ein Angestellter verfügt über einen Namen und eine vierstellige Personalnummer. Durch Zugriffsmethoden sollen alle Attribute gesetzt und abgefragt werden können.

- ☞ **Erstellen Sie das Klassenstrukturdiagramm für die Klasse Angestellter in der UML!**
- ☞ **Achten Sie auch auf Initialwerte für die Attribute!**



Die Realisierung der Klasse Angestellter

- ☞ **Implementieren Sie die Klasse Angestellter in der Headerdatei Angestellter.h!**
- ☞ **Die Methodendefinitionen erfolgen in der Quelltextdatei Angestellter.cpp!**

Arbeiten mit einem Objekt

- ☞ **Erzeugen Sie das Objekt derAG in der Hauptfunktion main( )!**
- ☞ **Setzen Sie den Namen auf "Hans Muster" und die Personalnummer auf 4711!**
- ☞ **Geben Sie Namen und Personalnummer auf der Konsole aus!**

Konstruktor und Destruktor

- ☞ **Fügen Sie folgende Textausgabe im Methodenrumpf des Konstruktors ein: "\nKonstruktor\n"!**
- ☞ **Verfahren Sie analog mit dem Destruktor!**

public und private

- ☞ **Verschieben Sie die Get- Methoden in den private - Bereich der Klassendefinition!**
- ☞ **Testen Sie den Zugriff "derAG.persNr = 9999;" in der Hauptfunktion main( )!**