

## Praktikumstermin Nr. 12, INF: Zusatz "Aufgabe für das Aufgabentutorium"

2021-02-03: Unklarheiten in der Aufgabenstellung korrigiert.

### **Für das Aufgaben-Tutorium am Freitag 5.2.2021, als freiwillige Aufgabe:**

Ein Versandservice versendet Briefe und Pakete und identifiziert diese über eine ID. Zusätzlich werden der Adressat und der Absender gespeichert, sowie ein Flag, das anzeigt, ob das Versandstück bereits zugestellt wurde. Dazu dient eine **Klasse Mail**, die die gemeinsamen Eigenschaften eines Briefes und eines Pakets darstellt.

### **(Freiwillige) Teil-Aufgabe INF-12.09: UML Klassendiagramm & Programmierung der C++ Klasse Address**

Jeder Brief und jedes Paket »hat« zwei Adressen. Definieren Sie daher zunächst eine Klasse Address, die folgende Adressangaben verwaltet:

- Nachname Vorname
- Straße Hausnummer
- PLZ Stadt
- Land

Entsprechend besitzt die Klasse vier Datenelemente vom Typ string: name, street, city, country. Diese seien sowohl von außerhalb als auch aus abgeleiteten Klassen nicht zugreifbar. Definieren Sie Setter und Getter für diese Attribute.

Stellen Sie zur Initialisierung außer einem Standard-Konstruktor, der eine leere Adresse (nur leere Stringwerte in den Attributen) erzeugt, auch einen Konstruktor mit vier Parametern bereit. Der vierte Parameter soll für das Land einen Defaultwert, z.B. "Deutschland", besitzen.

Eine Adresse mit einem leeren String — mit Ausnahme des Landes - ist ungültig. Ob eine Adresse gültig oder ungültig ist, soll mit der Methode is\_valid() abgefragt werden können.

Stellen Sie zum Einlesen einer Adresse von der Tastatur die Methode `scan()` bereit, die keinen Parameter besitzt. Wird beim Einlesen eines Werts — mit Ausnahme des Landes - nur die Return-Taste gedrückt, ist die Eingabe ungültig und die Methode liefert `false` zurück, andernfalls `true`. Wird bei Eingabe des Landes nur die Return-Taste gedrückt, so soll das Land "Deutschland" in das Objekt eingetragen werden. Bei einer ungültigen Eingabe soll das Objekt gar nicht geändert werden.

Überladen Sie die globale Funktion für den **Ausgabe-Operator <<**, um die Klasse in formatierter Ausgabe auf den Bildschirm (vgl. Testlauf) ausgeben zu können. Deklarieren Sie die Operatorfunktion zum **friend der Klasse**, so dass im Rumpf der Operatorfunktion auf die Attribute des Address Objekts zugegriffen werden kann.

- a) Erstellen Sie das UML-Klassendiagramm der Klasse Address.  
*Hinweis:* Die globale Operatorfunktion wird nicht im UML Diagramm berücksichtigt, da sie nicht Teil der Klasse ist.
- b) Programmieren Sie in einer vollständigen Headerdatei `address.h` den C++ Code der Klasse Address. Nur die Methode `scan()` sowie die globale Operator-Funktion sollen in der `.cpp`-Datei `address.cpp` programmiert werden.

## (Freiwillige) Teil-Aufgabe INF-12.10: UML Klassendiagramm & Programmierung der C++ Klasse Mail

Entwerfen Sie nun eine Klasse Mail mit Attributen `id` vom Typ `int` für die ID, `from` und `to` zum Speichern der Adressen sowie einem Flag `delivered`, das den Wert `true` aufweist, falls die Post ausgeliefert wurde. **Die Attribute sollen aus abgeleiteten Klassen zugreifbar sein**, von außen aber nicht. Definieren Sie Setter und Getter für diese Attribute. Eine neue Adresse soll aber nur gesetzt werden können, wenn sie gültig ist.

Definieren Sie einen Konstruktor, der als Argument die ID des neuen Mail-Objekts erhält, sowie einen zweiten Konstruktor, dem zusätzlich noch die Adressen des Absenders und des Empfängers übergeben werden.

Ferner soll ein Versandstück soll mittels der Methode `deliver()` ausgeliefert werden (Attribut `delivered` werde in der Methode auf `true` gesetzt), aber nur dann, wenn die Empfängeradresse gültig ist (dann `true` als Rückgabewert von `deliver()`, sonst `false`).

Definieren Sie außerdem die Methoden `scan_from()` und `scan_to()` zum Einlesen des Senders und Empfängers von der Tastatur. Die Rückgabewerte dieser Methoden sollen dieselben sein wie die der entsprechenden Methode in der Klasse `Address`.

Überladen Sie auch für die Klasse `Mail` die globale Operator-Funktion `<<`. Diesmal soll die Operator-Funktion nicht als `friend` deklariert werden.

Setzen Sie die Deklaration der Klasse in eine entsprechende Header-Datei `Mail.h` und die Implementierung aller Methoden (außer den Konstruktoren) in eine entsprechende cpp-Datei `Mail.cpp`.

## (Freiwillige) Teil-Aufgabe INF-12.11: Hauptprogramm

Nutzen Sie das folgende Hauptprogramm:

```
// Datei: main.cpp

#include <iostream>
using namespace std;

#include "Mail.h"

int main()
{
    Mail aMail(1020304);

    cout << "Bitte Absender angeben:" << endl;
    if (!aMail.scan_from())
        cout << "Ungültige Eingabe!" << endl << endl;

    cout << "und der Empfaenger:" << endl;
    if (!aMail.scan_to())
        cout << "Ungültige Eingabe!" << endl;

    cout << aMail << endl;
    cout << "... Post mit der ID " << aMail.get_id();
    if (aMail.deliver())
        cout << " wurde an " << aMail.get_to().get_name()
            << " ausgeliefert!" << endl;
    else
        cout << " konnte nicht ausgeliefert werden!" << endl;

    system("PAUSE");
    return 0;
}
```

## Testlauf (Benutzereingaben sind unterstrichen):

---

Bitte Absender angeben:

Nachname, Vorname: Mustermann, Max

Strasse mit Hausnr.: Turmstr. 12

Postleitzahl Stadt: 12345 Aachen

Land:

und der Empfaenger:

Nachname, Vorname: Mustermann, Maria

Strasse mit Hausnr.: Badstr. 34

Postleitzahl Stadt: 67890 Aachen

Land:

Mail-ID: 1020304

Absender:

Mustermann, Max

Turmstr. 12

12345 Aachen

Deutschland

Empfaenger:

Mustermann, Maria

Badstr. 34

67890 Aachen

Deutschland

Post wurde noch nicht ausgeliefert!

... Post mit der ID 1020304 wurde an Mustermann, Maria ausgeliefert!

Drücken Sie eine beliebige Taste . . .

---