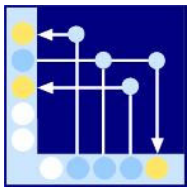
Fakultät Elektronik und Informatik
Studiengang Informatik



Programmieren in C++

Vorlesung im Wintersemester 2017/2018
Prof. Dr. habil. Christian Heinlein

4. Übungsblatt (16. November 2017)

Aufgabe 8: Abgeleitete Klassen

a) Definieren Sie Basisklassen `Person` (mit `string name`), `Male` (mit `bool bearded`) und `Female` (mit `bool pregnant`) mit sinnvollen Konstruktoren!

b) Definieren Sie abgeleitete Klassen `Man` und `Woman` als Kombination von `Person` mit `Male` bzw. `Female`, ebenfalls mit sinnvollen Konstruktoren!

c) Definieren Sie eine abgeleitete Klasse `Student` mit Basisklasse `Person` und Datenelement Matrikelnummer (`int number`) sowie `MaleStudent` und `FemaleStudent` als Kombination von `Student` mit `Man` bzw. `Woman`!
Achten Sie auf eine semantisch sinnvolle Verwendung virtueller Basisklassen!
In welcher Reihenfolge werden die Konstruktoren der einzelnen Teilobjekte ausgeführt? Welche Konstruktoraufrufe werden möglicherweise ignoriert?

d) Definieren Sie eine abgeleitete Klasse `DoubleStudent` mit einem `Person`- und zwei `Student`-Teilen zur Repräsentation von Studierenden mit zwei Studienfächern bzw. Matrikelnummern!
Objekte dieser Klasse sollen polymorph als `Student`-Objekte sowohl des ersten als auch des zweiten Studienfachs verwendbar sein, so dass ein Student, der z. B. Informatik und Elektrotechnik studiert, sowohl in einem Feld `Student* in []` (Liste aller Informatik-Studenten) als auch in einem Feld `Student* et []` (Liste aller Elektrotechnik-Studenten) mit der jeweils passenden Matrikelnummer auftreten kann, z. B.:

```
DoubleStudent* ds = new DoubleStudent("Maier", 123, 456);
Student* s1 = (Student1*)ds; in[0] = s1;
Student* s2 = (Student2*)ds; et[0] = s2;
assert(s1->number == 123);
assert(s2->number == 456);
assert((Person*)s1 == (Person*)s2);
```

e)* Definieren Sie, wenn möglich, abgeleitete Klassen `MaleDoubleStudent` und `FemaleDoubleStudent` als Kombination von `DoubleStudent` mit `MaleStudent` bzw. `FemaleStudent`!
Objekte dieser Klassen sollen genau die folgenden Teilobjekte enthalten und entsprechend polymorph verwendbar sein:

- 1 `Person`-Teilobjekt
- 1 `Male`- bzw. `Female`-Teilobjekt
- 1 `Man`- bzw. `Woman`-Teilobjekt
- 2 `Student`-Teilobjekte

- 1 `DoubleStudent`-Teilobjekt
- 2 `MaleStudent`- bzw. `FemaleStudent`-Teilobjekte
- ggf. leere Teilobjekte künstlicher Zwischenklassen

Insbesondere sollen die Objekte nicht mehr als zwei `Student`-Teilobjekte besitzen:

```
MaleDoubleStudent* mds = new MaleDoubleStudent("Maier", true, 123, 456);

DoubleStudent* ds = mds;
Student* s1 = (Student1*)ds;
Student* s2 = (Student2*)ds;

MaleStudent* ms1 = (MaleStudent1*)mds;
MaleStudent* ms2 = (MaleStudent2*)mds;
assert(s1 == (Student*)ms1);
assert(s2 == (Student*)ms2);
```

Die erste korrekte Lösung, die per E-Mail an `christian.heinlein@hs-aalen.de` gesandt wird, wird mit einer Flasche Champagner prämiert!

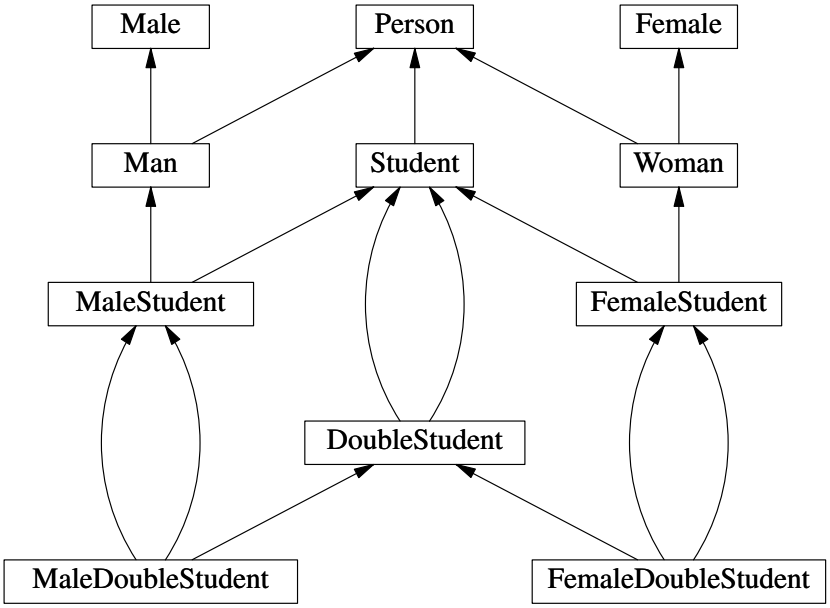
f) Erweitern Sie alle bisher definierten Klassen (außer eventuell benötigten Hilfsklassen) um ein Datenelement `long id`, das in den zugehörigen Konstruktoren initialisiert wird!
Für die Basisklassen `Person`, `Male` und `Female` soll für jedes Objekt eine global eindeutige Nummer vergeben werden, während in allen abgeleiteten Klassen die Summe der `id`-Werte der direkten Basisklassen zur Initialisierung des eigenen `id`-Werts verwendet werden soll.
Geben Sie alle `id`-Werte eines `MaleStudent`-, `FemaleStudent`- und `DoubleStudent`-Objekts aus!

g) Implementieren Sie eine Funktion

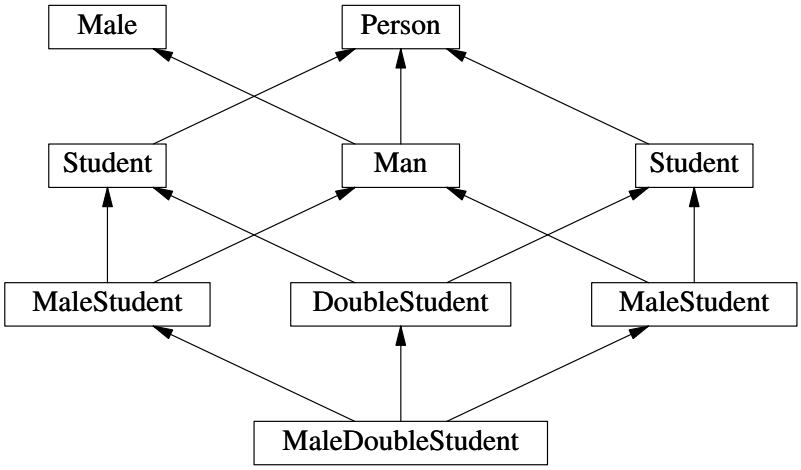
```
void print (Student* s, int d = 0);
```

die den Namen und die Matrikelnummer des Studenten `s` ausgibt!
Wenn der optionale Parameter `d` den Wert `1` bzw. `2` hat, soll die Funktion davon ausgehen, dass `s` auf das erste bzw. zweite `Student`-Teilobjekt eines `DoubleStudent`-Objekts zeigt, und zusätzlich die Matrikelnummer des anderen `Student`-Teilobjekts ausgeben.
Wie sehen entsprechende Aufrufe von `print` konkret aus? Mit welchen anderen Objekten kann `print` aufgerufen werden?

Typhierarchie (ohne künstliche Zwischenklassen)



Struktur von `MaleDoubleStudent`-Objekten



Anmerkung: Wenn man die Struktur dreidimensional auffasst, liegt die Raute mit den Typen `Person`, `Student` und `DoubleStudent` über der Raute mit den entsprechenden männlichen Typen `Man`, `MaleStudent` und `MaleDoubleStudent`.