TU München, Fakultät für Informatik
Lehrstuhl III: Datenbanksysteme
Prof. Alfons Kemper, Ph.D.

Übung zur Vorlesung *Einführung in die Informatik 2 für Ingenieure (MSE)*

Alexander van Renen (renen@in.tum.de)
http://db.in.tum.de/teaching/ss16/ei2/

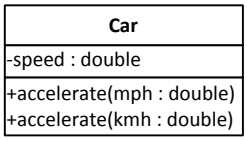
Blatt Nr. 3

Dieses Blatt wird am Montag, den 2. Mai 2016 besprochen.

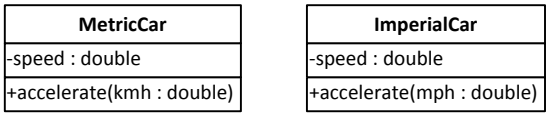
Aufgabe 1: Overloading

Welche der folgenden Methoden-Überladungen sind erlaubt und welche nicht? Überprüfen Sie Ihre Antworten indem Sie die Beispiele in Java programmieren.

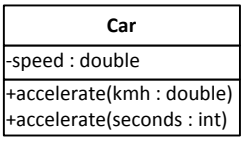
a)



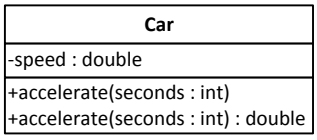
b)



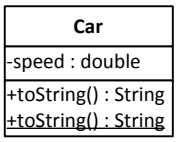
c)



d)

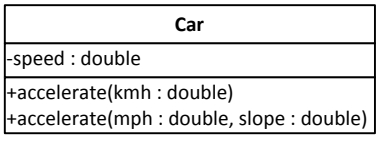


e)

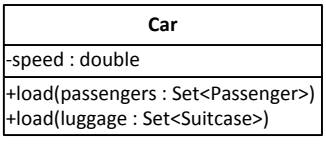


[1]

f)



g)



[1] Unterstrichen bedeutet hier Klassenmethode.

Aufgabe 2: Dynamisches Binden

Überlegen Sie sich welche Methoden aufgerufen werden, wenn man die Klasse `DynamicDispatch` ausführt. Überprüfen Sie anschließend ihre Vermutung, indem Sie das Programm tatsächlich ausführen.

```
1  class DynamicDispatch {
2    public static void main(String[] args) {
3      A a = new A();
4      B b = new B();
5      C c = new C();
6      D d = new D();
7
8      A[] array = {a, b, c, d};
9      for (A element : array) {
10       System.out.println("x():");
11       element.x();
12       System.out.println("\ny():");
13       element.y();
14       System.out.println("\nz():");
15       element.z();
16       System.out.println("\n=====\n");
17     }
18   }
19 }
20
21 class A {
22   public void x() {
23     System.out.println("-> A x()");
24     z();
25   }
26
27   public void y() {
28     System.out.println("-> A y()");
29     this.z();
30   }
31
32   public void z() {
33     System.out.println("-> A z()");
34   }
35 }
36
37 class B extends A {
38   public void y() {
39     System.out.println("-> B y()");
40     x();
41   }
42
43   public void z() {
```

```
44      System.out.println("-> B z()");
45    }
46  }
47
48  class C extends B {
49    public void x() {
50      System.out.println("-> C x()");
51      z();
52    }
53  }
54
55  class D extends A {
56    public void x() {
57      System.out.println("-> D x()");
58      super.x();
59    }
60
61    public void z() {
62      System.out.println("-> D z()");
63    }
64  }
```

Aufgabe 3: Don't repeat yourself (DRY)

Auf der Webseite[2] finden Sie die Klasse `Warenhaus.java`.

a) Nennen Sie mindestens zwei Gründe, warum die Modellierung nicht optimal ist – beispielsweise, wenn sich der Mehrwertsteuersatz mal wieder ändern sollte.

b) Implementieren Sie den Aufzählungstyp `Mehrwertsteuersatz`, der zwei Werte für den normalen und den vergünstigten Mehrwertsteuersatz hat.

c) Ziehen Sie die Gemeinsamkeiten der Produkte in eine gemeinsame Oberklasse `Produkt`. Führen Sie dafür eine Methode ein, die für ein Produkt angibt, ob die vergünstigte Mehrwertsteuer anwendbar ist. Verwenden Sie diese Methode bei der Preisberechnung.

d) Führen Sie die Klasse `Einkaufskorb` ein, die eine Menge von Produkten verwaltet und deren Gesamtpreis bestimmen kann. Verwenden Sie diese Klasse in der main-Methode des Warenhauses.

[2] Hier: `http://db.in.tum.de/teaching/ss16/ei2`