

Übungsblatt 5

Hinweis:  
Die Sichtbarkeit von Operationen und Attributen wird in Klassendiagrammen wie folgt gekennzeichnet:  
• „+“ für public – (engl. öffentlich), unbeschränkter Zugriff  
• „#“ für protected – (engl. geschützt), Zugriff nur von der Klasse sowie von Unterklassen (Klassen, die erben)  
• „-“ für private – (engl. privat), nur die Klasse selbst kann es sehen  
• „~“ für package – (engl. Paket), innerhalb des Pakets sichtbar (nur in wenigen Programmiersprachen, etwa Java und C#, implementierbar)

Aufgabe 5.1: Klasse „Konto“

Gegeben ist folgendes Klassendiagramm:

Konto
-kontonummer: int -betrag: double
+Konto(nummer: int, betrag: double) +getBetrag(): double +zahleEin(betrag: double) +hebeAb(betrag: double) +toString(): String

Implementieren Sie diese Klasse.  
Schreiben sie eine Klasse mit einer main-Methode, in der:  
- ein Objekt von Konto erzeugt wird  
- ein Betrag eingezahlt wird  
- ein Betrag abgehoben (ausgezahlt) wird  
- dazwischen soll der aktuelle Zustand des Objektes ausgegeben werden  
(Z.B.: „Das Konto mit der Nummer 123456 hat einen aktuellen Betrag von 500,50 Euro)

Aufgabe 5.2: Klasse „Box“

Gegeben ist folgendes Klassendiagramm:



Box
-length: int -width: int -height: int
+Box(length: int, width: int, height: int) +Box(length: int) +scale(scalingFactor: int) +getVolume(): int +getSurfaceArea(): int +isCube(): boolean +toString(): String

Implementieren Sie diese Klasse.  
Schreiben sie eine Klasse mit einer main-Methode, in der:  
- ein Objekt von Box mit Länge=3, Breite=4 und Höhe=5 erzeugt wird  
- ein Würfel als spezielles Objekt von Box mit der Kantenlänge von 4 erzeugt wird  
- Volumen und Oberfläche sollen für beide ausgegeben werden  
- die beiden Objekte werden mit dem Faktor 2 skaliert  
- Volumen und Oberfläche sollen für beide nochmals ausgegeben werden

Aufgabe 5.3: Klasse „Smiley“

In der ersten Übung haben Sie einen Smiley „programmiert“. Jetzt machen wir das Objekt-orientiert!

Die folgende Tabelle zeigt links den Code für die rechts gezeigten Smileys:

<pre>&lt;svg version="1.1" baseProfile="full" xmlns:ev="http://www.w3.org/2001/xml-events" xmlns:xlink="http://www.w3.org/1999/xlink" xmlns="http://www.w3.org/2000/svg" preserveAspectRatio="xMidYMid meet" zoomAndPan="magnify" id="MySmiley" viewBox="-21 -21 42 42" width="800" height="800"&gt; &lt;circle r="20" stroke="black" stroke-width="1" fill="black"/&gt; &lt;circle r="20" fill="yellow"/&gt;   &lt;ellipse rx="2.5" ry="4" cx="6" cy="-7" fill="black"/&gt;   &lt;ellipse rx="2.5" ry="4" cx="-6" cy="-7" fill="black"/&gt; &lt;path fill="none" stroke="black" stroke-width=".75" d="M -12,5 A 13.5,13.5,0 0,0 12,5 A 13,13,0 0,1 -12,5"/&gt; &lt;/svg&gt;</pre>	
<pre>&lt;svg version="1.1" baseProfile="full" xmlns:ev="http://www.w3.org/2001/xml-events" xmlns:xlink="http://www.w3.org/1999/xlink" xmlns="http://www.w3.org/2000/svg" preserveAspectRatio="xMidYMid meet" zoomAndPan="magnify" id="MySmiley" viewBox="-21 -21 42 42" width="800" height="800"&gt; &lt;circle r="20" stroke="black" stroke-width="1" fill="black"/&gt; &lt;circle r="20" fill="red"/&gt;   &lt;ellipse rx="2.5" ry="4" cx="6" cy="-7" fill="black"/&gt;   &lt;ellipse rx="2.5" ry="4" cx="-6" cy="-7" fill="black"/&gt;   &lt;ellipse rx="8" ry="4" cx="0" cy="8" fill="black"/&gt; &lt;/svg&gt;</pre>	

Entwerfen Sie eine Klasse „Smiley“ mit Attributen für:

- Vordergrundfarbe (Linien, Augen, Mund)
- Hintergrundfarbe (Gesichtsfarbe)
- Stimmung („happy“, „angry“, weitere sind erwünscht)

Die Klasse soll einen passenden Konstruktor haben.

Mit einer Methode saveSVG(String filename) kann ein svg-Bild des Smileys gespeichert werden.  
In der main-Methode sollen mehrere Objekte von der Klasse mit unterschiedlichen Stimmungen angelegt werden. Speichern Sie dann die unterschiedlichen svg-Grafiken. Sie können sich diese mit Viewern wie z.B. Chrome, Libre Office oder mittels Online-Viewern ansehen. (Z.B.: <https://www.freecodeformat.com/svg-editor.php> )