Rheinische Friedrich-Wilhelms-Universität Bonn
Institut für Informatik III
Prof. Dr. A.B. Cremers

Aufgabenblatt 3

— Veröffentlicht am 12.05.2010 —

Abgabe bis 18.05.2010, 13:00 per SVN

Aufgabe 1 *(Wahr oder Falsch? - 2 Punkte)*

Überlegen Sie sich für jede Aussage ein Argument, mit dem Sie eindeutig bestimmen können, ob die Aussage wahr oder falsch ist. Für jede richtig angekreuzte Antwort werden 0,5 Punkte angerechnet, für jede falsch angekreuzte Antwort 0,5 Punkte abgezogen. Nicht angekreuzte Antworten geben keinen Punktabzug.

wahr	falsch	
		In Java ist `(a & b || c)` gleichbedeutend zu `(a && b | c)`. (Die Variablen `a,b,c` sind vom Typ `boolean`)
		Überladene Methoden müssen alle den gleichen Ergebnistyp haben.
		Für die Signatur einer Java-Methode sind nur ihr Name sowie Reihenfolge und Typ der Parameter maßgebend.
		Es kann Java-Klassen ohne dazugehörige Objekte geben, aber keine Java-Objekte ohne dazugehöriger Klasse.

Aufgabe 2 *(Teilausdrücke - 6 Punkte)*

Gegeben sind die folgenden Ausdrücke in Java:

```
((3+7)*2/5)/(1+3.1415)
((3*2/7)+1)/(1.4+2.7183)
(1.5-0.5)*((13/2+5)/2)
```

a) Bestimmen Sie alle Teilausdrücke dieser drei Ausdrücke.

b) Ermitteln Sie zu jedem Teilausdruck den Typ und markieren Sie, wo eine Typkonversion stattfindet. Welcher Typ wird in welchen konvertiert?

c) Berechnen Sie die Ergebnisse der Teilausdrücke und die Gesamtergebnisse der drei Ausdrücke. Beachten Sie dabei den Typ des Teilausdrucks.

Aufgabe 3 *(Kontrollstrukturen - 7 Punkte)*

a) Drücken Sie folgende *switch*-Kontrollstruktur nur mit Hilfe von *if*-Abfragen in Java aus:

```
switch (key) {
  case 1: I(); break;
  case 2: break;
  case 3: J();
  case 4: K(); break;
  case 5: case 6: L(); break;
  case 7: M();
  default: N();
}
```

b) Geben Sie für jede nummerierte Zeile des folgenden Programmfragmentes an, wie häufig sie durchlaufen wird:

```
 1: int i = -1;
 2: out: while (true) {
 3:   i++;
 4:   int j = -i;
 5:   if (i > 6)
 6:      break;
 7:   if (j < -3)
 8:      continue;
 9:   i += 1;
10:   in: while (i == -1) {
11:     i = -1;
12:     break out;
      }
   }
```

Aufgabe 4 *(Statisch-Funktionale Programmierung mit Klassen - Teil 3 - 10 Punkte)*

In den Aufgaben 3 und 4 des letzten Übungsblattes wurde ein System zur Verwaltung von Kraftfahrzeugen entwickelt. Dieses System soll nun erweitert werden.

Erstellen Sie eine Kopie des Eclipse-Projekts von letzter Woche.[1] Erweitern Sie Ihre kopierten Klassen so, dass bis zu drei *TransportAuthorities* und bis zu zehn *MeterMaids* unterstützt werden.

Erweitern Sie das kopierte Programm aus Teilaufgabe 4c des letzten Übungsblattes so, dass zwei Autos bei zwei verschiedenen *TransportAuthorities* registriert werden. Eine *MeterMaid* soll das erste Auto überprüfen, eine Strafe registrieren und einen Strafzettel ausstellen. Im Anschluss daran soll eine zweite *MeterMaid* das gleiche mit dem zweiten Auto durchführen.

Wichtig: Definieren Sie in den Klassen *TransportAuthority* und *MeterMaid* nur Klassen-Variablen und Klassen-Funktionen, die mit dem Schlüsselwort *static* gekennzeichnet sind!

[1] Das Projekt im *Package Explorer* auswählen und dann *Edit* → *Copy* gefolgt von *Edit* → *Paste* ausführen.