KLAUSUR SOFTWARETECHNIK

17.09.2009

Prof. Dr. Walter F. Tichy
Dipl.-Inform. Andreas Höfer
Dipl.-Inform. David J. Meder

Musterlösung

Zur Klausur sind keine Hilfsmittel und kein eigenes Papier zugelassen.
Die Bearbeitungszeit beträgt 60 Minuten.
Die Klausur ist vollständig und geheftet abzugeben.
Mit Bleistift oder roter Farbe geschriebene Angaben werden nicht bewertet.

Aufgabe	1	2	3	4	5	∑
Maximum	14	12	11	13	10	60
Korrektor 1						
Korrektor 2						
Korrektor 3						

Aufgabe 1: Aufwärmen (14 Punkte)

a) Kreuzen Sie an, ob die folgenden Aussagen wahr oder falsch sind. (5 P)
Hinweis: Jedes korrekte Kreuz zählt 0,5 Punkte, jedes falsche Kreuz bewirkt 0,5 Punkte Abzug. Die Teilaufgabe wird mit mindestens 0 Punkten bewertet.

Aussage	Wahr	Falsch
Das Lastenheft ist ein Ergebnis der Definitionsphase.		X
Anwendungsfälle helfen dabei, Grenzen des Softwaresystems zu bestimmen.	X	
Solange sich ein Objekt in einem Zustand befindet, reagiert es im gleichen Kontext immer gleich auf seine Umwelt.	X	
Unter welchen Umständen ein Objekt welche Botschaft entgegen nimmt, spezifiziert man in einem UML-Zustandsdiagramm.	X	
In UML drückt eine Verknüpfung eine mögliche Beziehung zwischen Exemplaren aus.		X
Das V-Modell XT besteht aus den vier Submodellen Projektmanagement, Qualitätssicherung, Konfigurationsmanagement und Systemerstellung.	X	
Die Verwendung von Vorwärts-Deltas in einem Konfigurationsmanagement-System bietet im Vergleich zur Verwendung von Rückwärts-Deltas schnelleren Zugriff auf die aktuelle Softwareversion.		X
Eine Klasse B wird erst dann zu einer Unterklasse einer Klasse A, wenn gezeigt werden kann, dass jedes Exemplar von A auch als ein Exemplar von B gesehen werden kann.		X
Akzeptanztests müssen bei jeder Code-Integration zu 100 % erfolgreich laufen.		X
Der Systemtest ist der abschließende Test des Systems in realer Umgebung unter Beobachtung, Mitwirkung und Federführung des Kunden beim Kunden.		X

b) Welche Punkte müssen erfüllt sein, damit wir von einer strukturerhaltenden Transformation sprechen können? Vervollständigen Sie folgende Definition. (1,5 P)

Wir sprechen von einer strukturerhaltenden Transformation einer Quellsprache in die Zwischensprache, wenn …

Pro genanntem Punkt: 0,5 P

- (ausschließlich) die Befehle, die die Ausführungsreihenfolge beeinflussen, durch Befehlsfolgen der Zwischensprache ersetzt werden, wobei
- die Ausführungsreihenfolge der anderen Befehle bei gleicher Parametrisierung gleich bleibt mit der in der Quellsprache! Alle anderen Befehle unverändert übernommen werden
- Transformationen, bei denen Anweisungsfolgen oder bedingte Sprünge repliziert werden, vermieden werden (kein Ausrollen von Schleifen, keine Optimierungen.)

c) Erklären Sie die Begriffe „Wartung“ und „Pflege“. (2 P)

Wartung:
Lokalisierung und Behebung von Fehlerursachen (0,5 P) von in Betrieb befindlichen Software-Produkten, wenn die Fehlerwirkung (0,5 P) bekannt ist (nicht planbar).
Pflege:
Lokalisierung und Durchführung von Änderungen und Erweiterungen (0,5 P) von in Betrieb befindlichen Software-Produkten, wenn die Art der gewünschten Änderungen/Erweiterungen festliegt (0,5 P) (planbar).

d) Nennen Sie drei Indikatoren, die in einem UML-Klassendiagramm für die Verwendung einer Assoziation anstelle eines Attributes sprechen. (1,5 P)

0,5 P pro genanntem Indikator, max. 1,5 P:
- Transaktionalität wird gebraucht
- Mehrere Gegenüber werden gebraucht
- Es ist nötig, vom Gegenüber zurück zu navigieren
- Umsetzung von Kardinalitäten

e) Erklären Sie den Unterschied zwischen Signatur- und Implementierungsvererbung. (2 P)

Signaturvererbung: 1 P
Eine in der Oberklasse definierte und (evtl.) implementierte Methode überträgt nur ihre Signatur auf die Unterklasse.
Implementierungsvererbung: 1 P
Eine in der Oberklasse definierte und implementierte Methode überträgt ihre Signatur und ihre Implementierung auf die Unterklasse.

f) Welche Arten der Parameter-Varianz erfüllen das Substitutionsprinzip? Welche Arten der Parameter-Varianz sind in Java zulässig? Tragen Sie „ja“ in ein Kästchen ein, wenn die Art der Parameter-Varianz das Substitutionsprinzip erfüllt bzw. in Java erlaubt ist, ansonsten tragen Sie „nein“ ein. (2 P)

	Eingabeparameter		Ausgabeparameter	
	Allgemein	Java	Allgemein	Java
Kovarianz	Nein	Nein	Ja	Ja
Kontravarianz	Ja	Nein	Nein	Nein

AUFGABE 2: UML-SEQUENZDIAGRAMME (12 P)

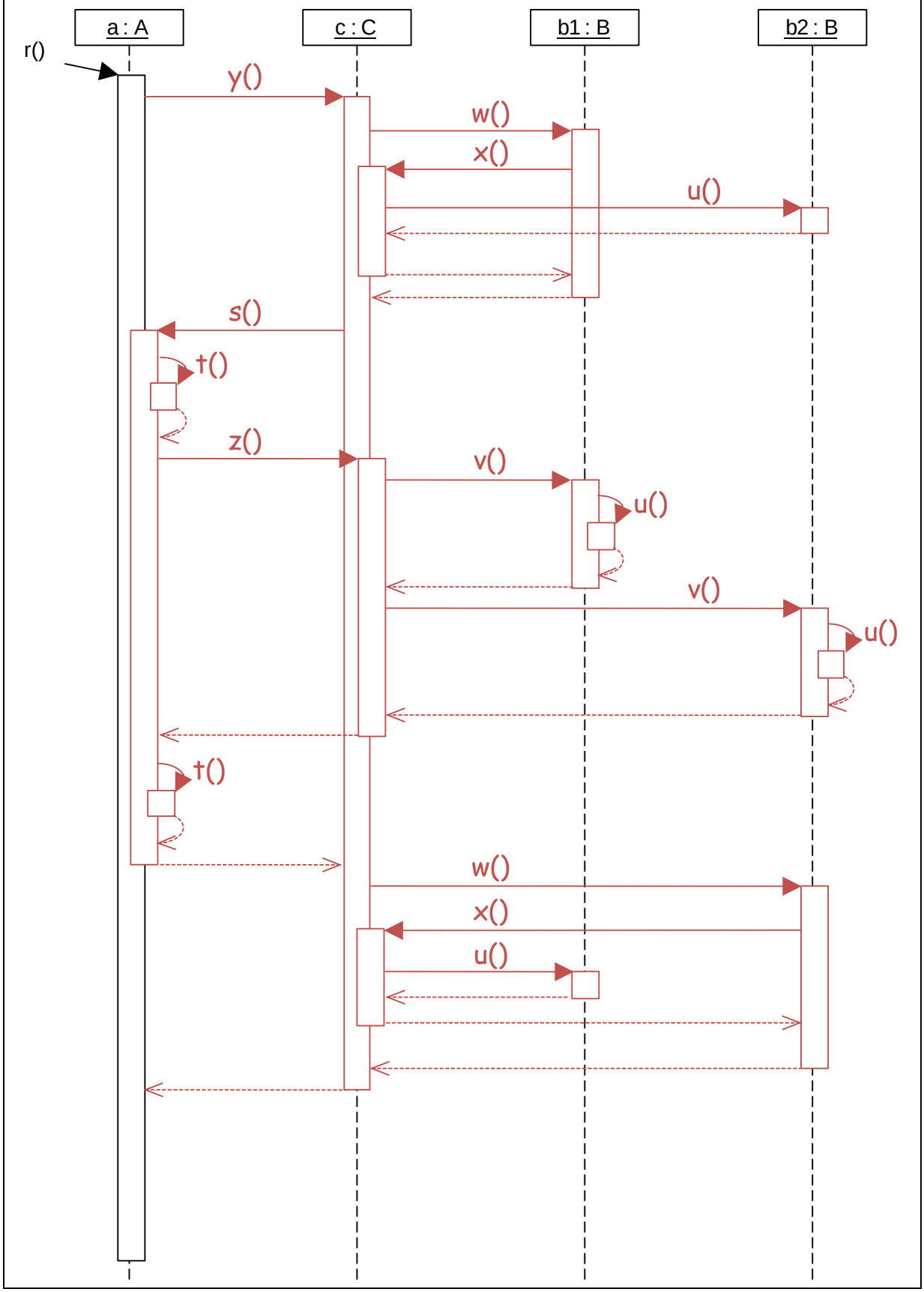
Gegeben sei folgender Java-Quellcode:

```
01  public class A {
02    private C c;
03
04    /**
05     * Hier geht es los.
06     */
07    public static void
    main(String[] args) {
08      A a = new A();
09      a.r();
10    }
11
12    public A() {
13      c = new C(this);
14    }
15
16    public void r() {
17      c.y();
18    }
19
20    public void s() {
21      t();
22      c.z();
23      t();
24    }
25
26    private void t() {
27    }
28  }
29
30  public class B {
31    private C c;
32
33    public B(C c) {
34      this.c = c;
35    }
36
37    public void u() { }
38
39    public void v() {
40      this.u();
41    }
42
43    public void w() {
44      c.x();
45    }
46  }
```

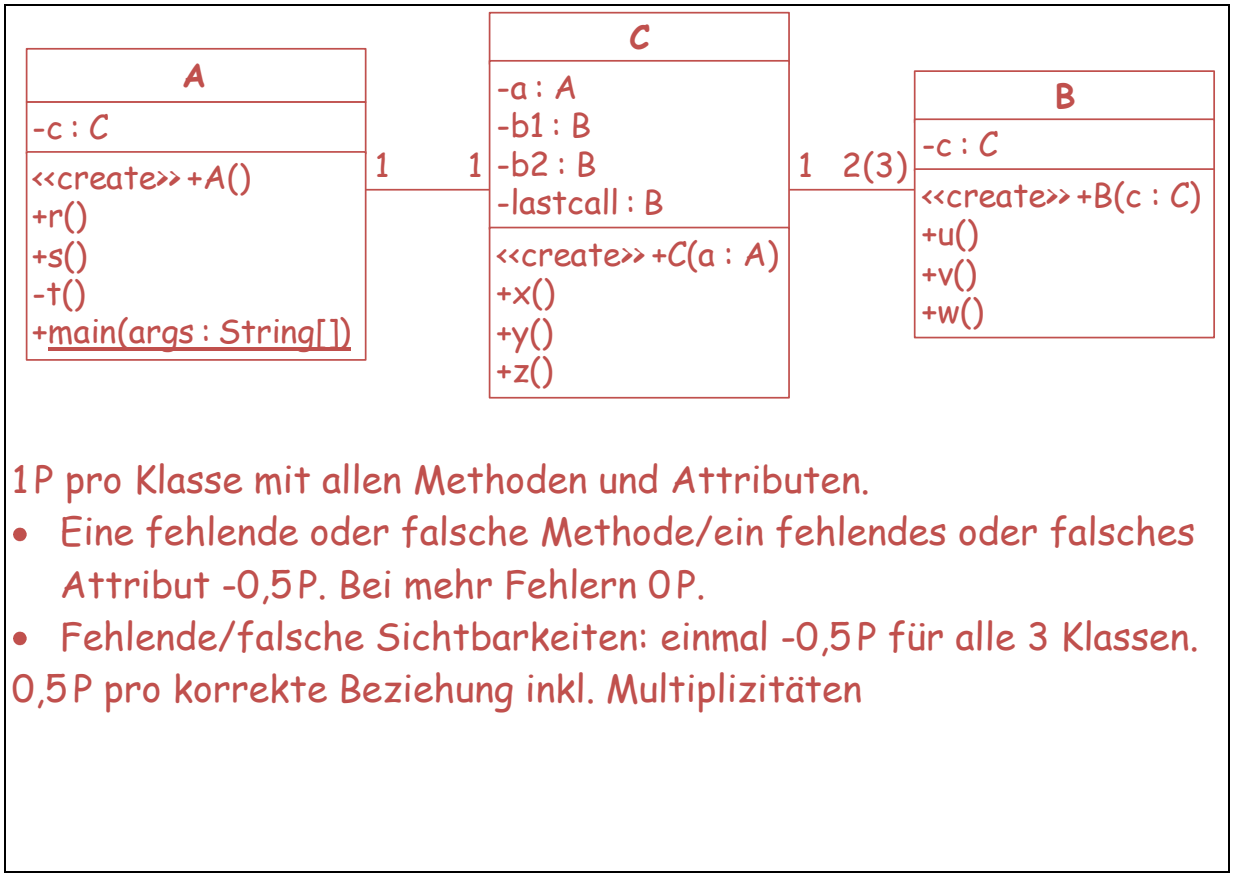
```
47  public class C {
48    private A a;
49    private B b1, b2;
50    private B lastCall;
51
52    public C(A a) {
53      this.a = a;
54      b1 = new B(this);
55      b2 = new B(this);
56    }
57
58    public void x() {
59      if (b1 == lastCall) {
60        b2.u();
61      } else {
62        b1.u();
63      }
64    }
65
66    public void y() {
67      lastCall = b1;
68      b1.w();
69      a.s();
70      lastCall = b2;
71      b2.w();
72    }
73
74    public void z() {
75      b1.v();
76      b2.v();
77    }
78  }
```

a) Vervollständigen Sie das UML-Sequenzdiagramm auf der folgenden Seite so, dass es exakt den Programmablauf des obigen Java-Quellcodes wiedergibt. Geben Sie alle Nachrichten und Antworten an. Achten Sie auf korrekte UML-Syntax. (8 P)

0,5 P pro korrektem Methodenaufruf + Rücksprung (max. 7,5 P)
0,5 P für korrekte UML-Pfeile, -1 P bei ungültiger UML-Notation



b) Zeichnen Sie das zu obigem Java-Quellcode gehörende UML-Klassendiagramm. Geben Sie zu allen Klassen Attribute, Methoden, Sichtbarkeiten und Multiplizitäten an. Achten Sie auf korrekte UML-Syntax. (4 P)



1 P pro Klasse mit allen Methoden und Attributen.

- Eine fehlende oder falsche Methode/ein fehlendes oder falsches Attribut -0,5 P. Bei mehr Fehlern 0 P.
- Fehlende/falsche Sichtbarkeiten: einmal -0,5 P für alle 3 Klassen.

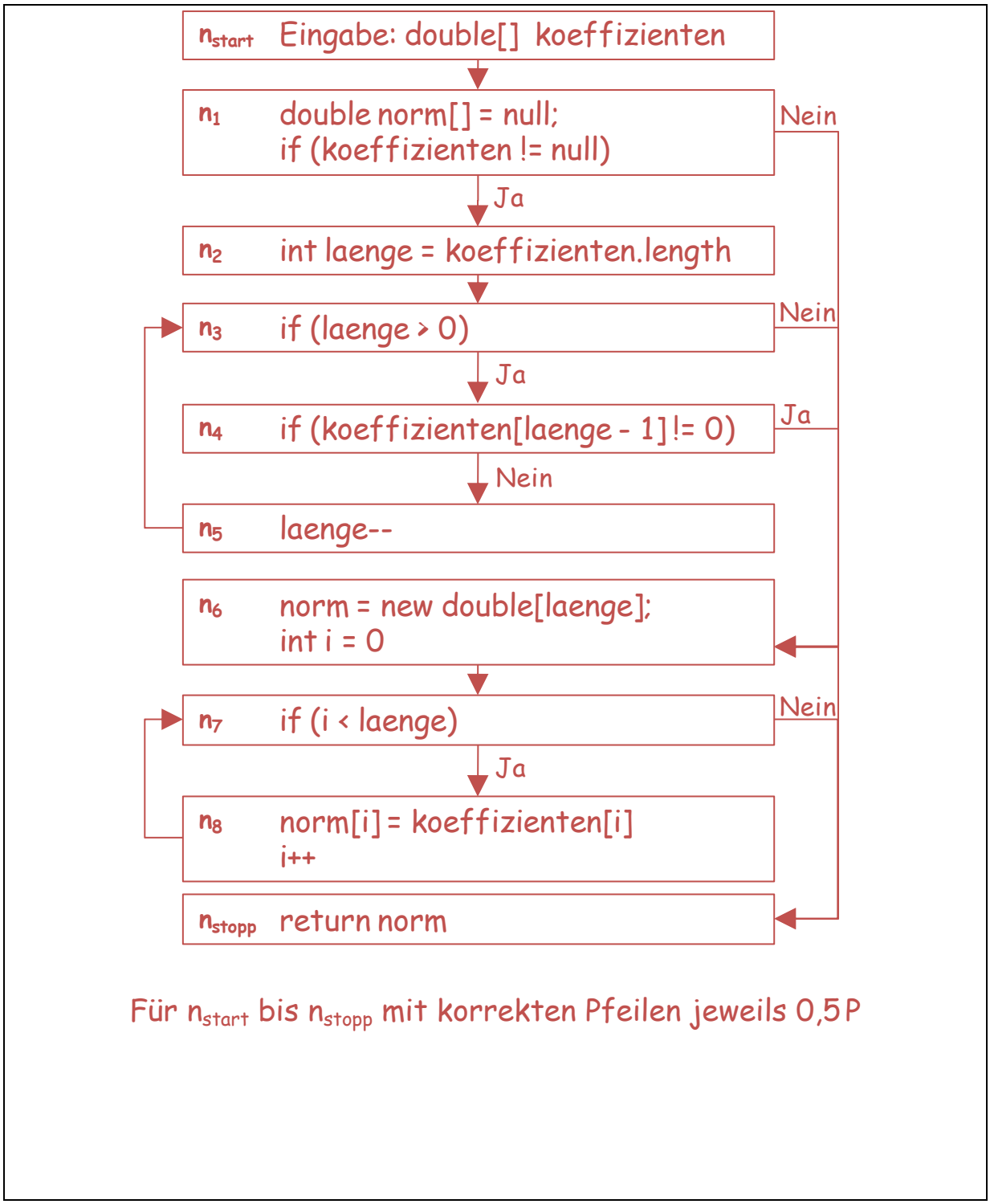
0,5 P pro korrekte Beziehung inkl. Multiplizitäten

Aufgabe 3: Kontrollflussorientiertes Testen (11 P)

Gegeben sei folgende Java-Methode:

```
01  public static double[] normalisiere(double[] koeffizienten) {
02    double norm[] = null;
03    if (koeffizienten != null) {
04      int laenge = koeffizienten.length;
05      for (; laenge > 0; laenge--) {
06        if (koeffizienten[laenge - 1] != 0) {
07          break;
08        }
09      }
10      norm = new double[laenge];
11      for (int i = 0; i < laenge; i++) {
12        norm[i] = koeffizienten[i];
13      }
14    }
15    return norm;
16  }
```

a) Erstellen Sie den Kontrollflussgraphen der Methode `normalisiere(…)` im Kasten auf der nächsten Seite. Bitte schreiben Sie den Quelltext in die Kästchen. Verweise auf die Zeilennummern der Methode sind nicht ausreichend. (5 P)



Für n_{start} bis n_{stopp} mit korrekten Pfeilen jeweils 0,5 P

b) Geben Sie eine minimale Testfallmenge an, welche die Zweigüberdeckung der Methode `normalisiere(…)` erfüllt. Geben Sie die durchlaufenen Pfade an. (3 P)

{x, 0.0} mit $x \in \Re$; $x \neq 0$ (Reihenfolge ist wichtig):

$n_{start}, n_1, n_2, n_3, n_4, n_5, n_3, n_4, n_6, n_7, n_8, n_7, n_{stopp}$ 1 P

{null}: $n_{start}, n_1, n_{stopp}$ 1 P

{}: $n_{start}, n_1, n_2, n_3, n_6, n_7, n_{stopp}$ 1 P

c) Nennen Sie die Teilpfade aller Schleifenquerer (Boundary-Interior-Test) der ersten **for**-Schleife (Zeilen 4 bis 9). (1 P)

$(n_2,)\ n_3, n_4, n_5$ und $(n_2,)\ n_3, n_4$	2 × 0,5 P 0 P, wenn vollständige Pfade

d) Geben Sie eine minimale Testfallmenge für den Grenztest (Boundary-Interior-Test) der ersten **for**-Schleife (Zeilen 4 bis 9) an. (1 P)

$\{0.0\}$ und $\{x\}$ mit $x \in \mathbb{R}; x \neq 0$	2 × 0,5 P

e) Geben Sie eine minimale Testfallmenge für den Interieurtest (Boundary-Interior-Test) der ersten **for**-Schleife (Zeilen 4 bis 9) an. (1 P)

Jedes Feld mit der Länge 2	1 P

Aufgabe 4: Modellierung (13 P)

Die Firma Canikuji, Hersteller des Fotosystems Ikonograf, möchte auf ihrer Webseite den Kunden alle Kombinationsmöglichkeiten innerhalb des Fotosystems präsentieren. Im Folgenden wird das Fotosystem Ikonograf beschrieben:

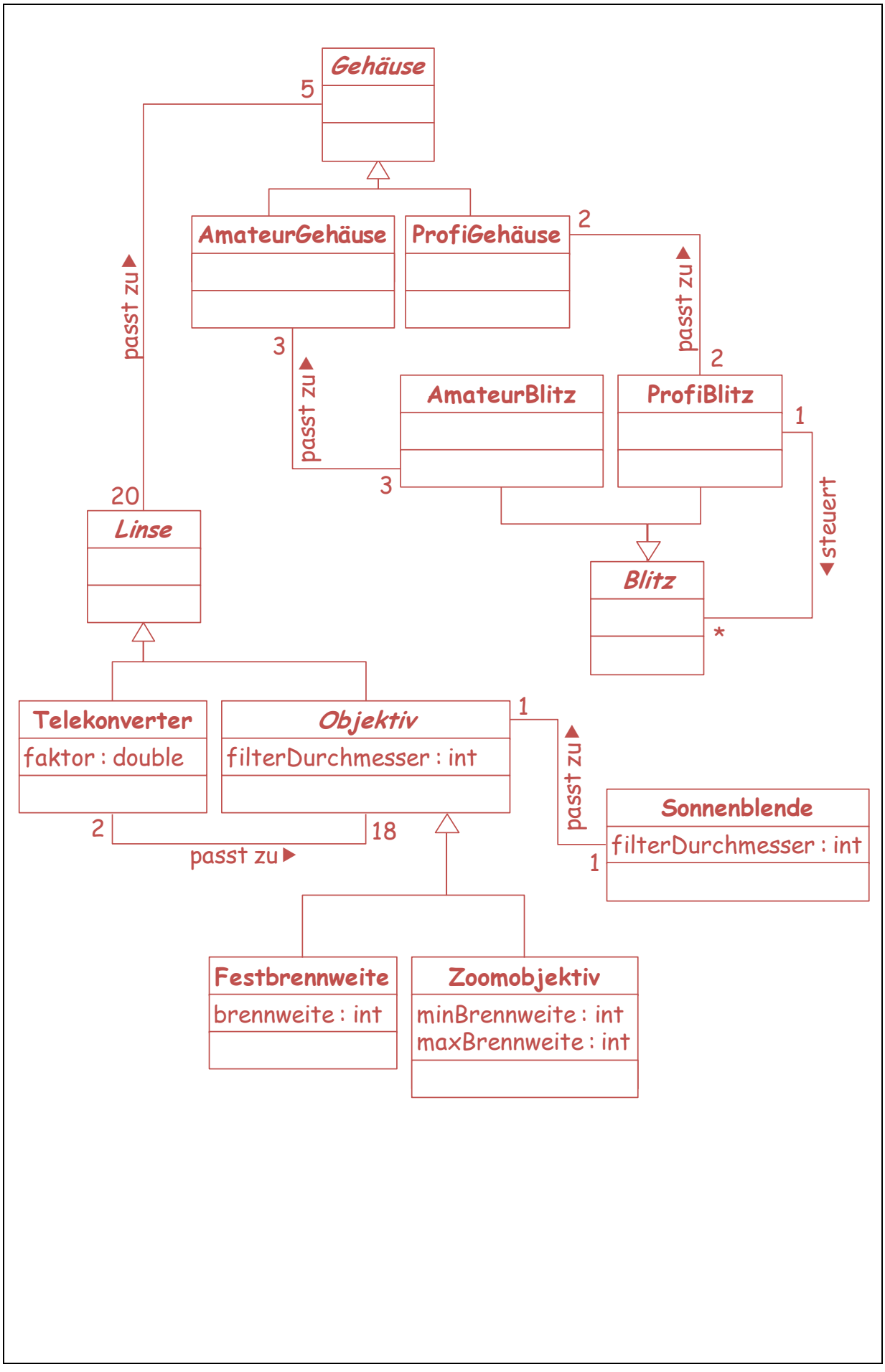
> *Das Fotosystem umfasst fünf verschiedene Kameragehäuse. Von diesen fünf Kameragehäusen sind zwei für den professionellen Einsatz und drei für den Amateurbereich gedacht. Zu den fünf Kameragehäusen gibt es insgesamt 20 verschiedene Linsen, die sowohl an den Profi- als auch an den Amateurkameragehäusen verwendet werden können.*
> *Unter den angebotenen Linsen befinden sich zwei Telekonverter sowie 18 unterschiedliche Objektive. Die Telekonverter lassen sich mit allen Objektiven kombinieren. Zu jedem Objektiv gibt es eine passende Sonnenblende; zu beiden ist jeweils der Durchmesser des Filtergewindes angegeben. Unter den Objektiven werden Festbrennweiten und Zoomobjektive unterschieden. Zoomobjektive besitzen eine minimale und eine maximale Brennweite, Festbrennweiten hingegen nur eine Brennweite. Der Wert der Brennweite muss immer größer als 0 sein. Die maximale Brennweite eines Zoomobjektives ist immer mindestens 5 mm größer als die minimale Brennweite des Zoomobjektives. Bei den Telekonvertern ist der Verlängerungsfaktor angegeben.*
> *Weiterhin umfasst das Fotosystem Blitze. Es gibt zwei Profiblitze, die nur an die Profikameragehäuse passen, und drei Amateurblitze, die nur an die Amateurkameragehäuse passen. Ein Profiblitz kann beliebig viele Blitze steuern.*

Setzen Sie den oben beschriebenen Sachverhalt vollständig in ein UML-Klassendiagramm um. Modellieren Sie keine Methoden und Sichtbarkeiten. Geben Sie Attribute, Multiplizitäten, Restriktionen, Assoziationsnamen sowie Rollen an. Sachverhalte, die nicht mit dem Klassendiagramm ausgedrückt werden können, geben Sie bitte in OCL an.

Fotosystem

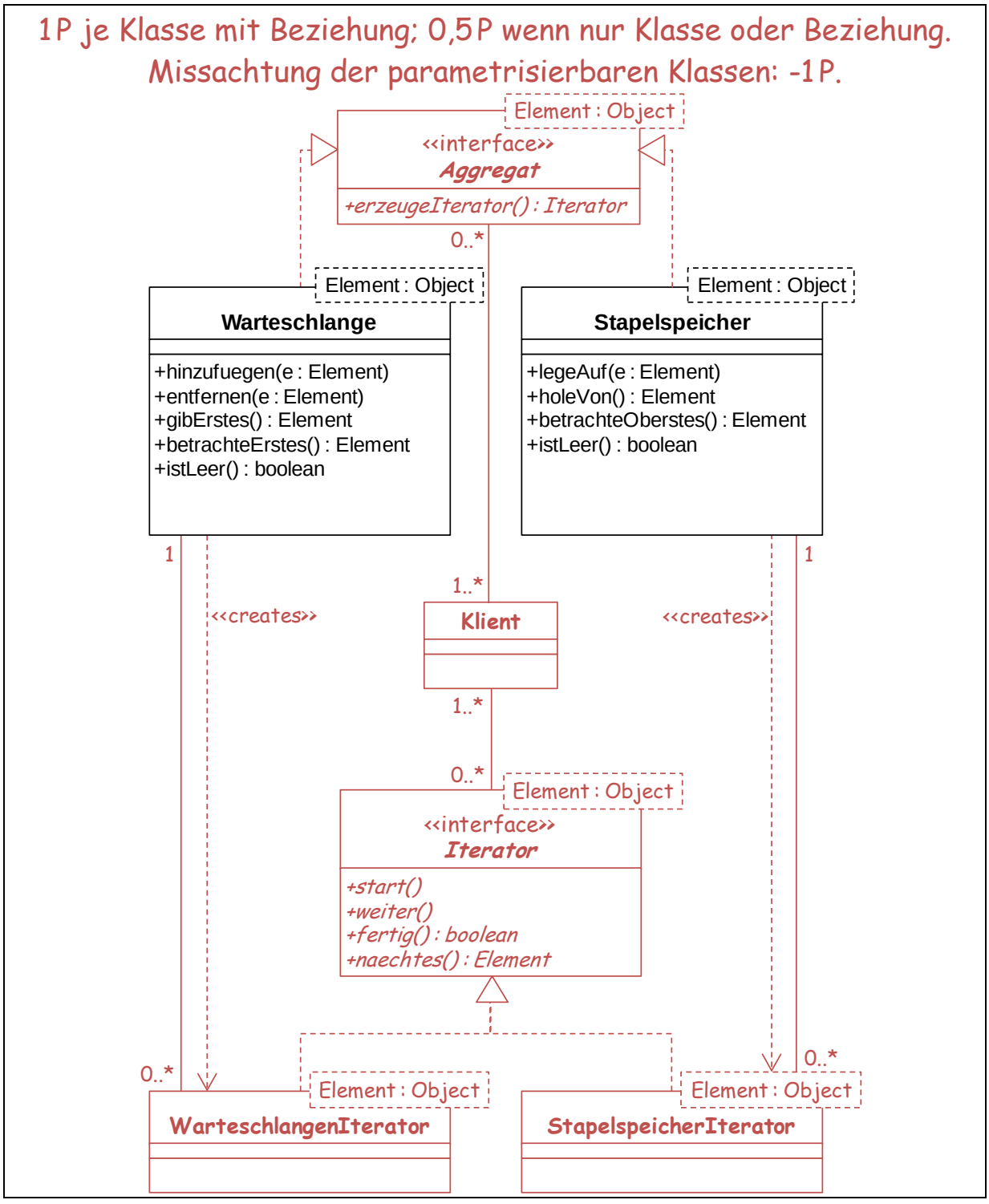
```
context Festbrennweite
        inv Brennweite: self.brennweite > 0                    1P

context Zoom
        inv Brennweite: self.minBrennweite > 0 and
                self.maxBrennweite > 0 and ← optional, da redundant
                (self.maxBrennweite – 5) ≥ self.minBrennweite  1P
```

12 Klassen (inkl. Attribute) × 0,5P
4 Vererbungen × 0,5P
6 Assoziationen × 0,5P
Summe: 11P
-1P bei fehlenden oder falschen Mutltiplizitäten
-1P bei fehlenden oder falschen Assoziationsnamen


Gehäuse
5
AmateurGehäuse
ProfiGehäuse
2
passt zu
passt zu
3
2
passt zu
AmateurBlitz
ProfiBlitz
1
3
20
Linse
steuert
Blitz
*
Telekonverter
faktor : double
Objektiv
filterDurchmesser : int
1
passt zu
Sonnenblende
filterDurchmesser : int
2
18
passt zu
1
Festbrennweite
brennweite : int
Zoomobjektiv
minBrennweite : int
maxBrennweite : int

AUFGABE 5: ENTWURFSMUSTER (10 P)

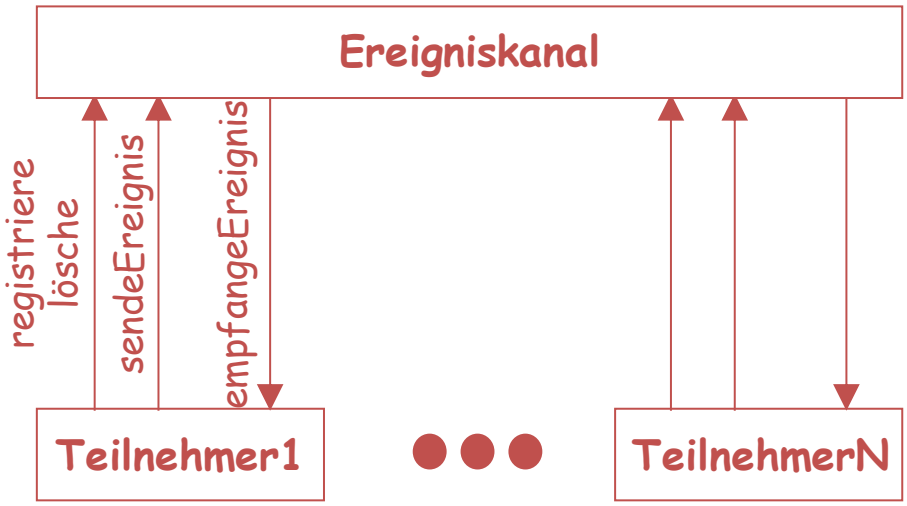
a) Ergänzen Sie das folgende Klassendiagramm, indem Sie das Iterator-Entwurfsmuster verwenden, um Klienten einheitliche Schnittstellen für den Zugriff auf die Elemente in Stapelspeichern und Warteschlangen zur Verfügung zu stellen. Achten Sie auf die richtigen Assoziationstypen und geben Sie Multiplizitäten an. Geben Sie für alle Ein- und Ausgabeparameter den Typ an. Implementierungen für Methoden müssen Sie nicht angeben.
Hinweis: Die gestrichelten Kästchen kennzeichnen parametrisierbare Klassen (entspricht Java-Generics). Die Angabe `Element : Object` bedeutet, dass der Typparameter `Element` nur von der Klasse `Object` selbst oder Klassen die von ihr erben besetzt werden kann. (5 P)



b) Skizzieren Sie das Ereigniskanal-Entwurfsmuster. Beschreiben Sie, welchen Zweck das Entwurfsmuster hat und wie es funktioniert. (5 P)

Zweck: Vollständige Entkopplung der Teilnehmer an einem Gesamtsystem, so dass diese völlig eigenständig arbeiten können (0,5 P). Ein Teilnehmer weiß nichts über die Existenz oder Anzahl anderer Teilnehmer (0,5 P).

Funktionsweise: Interaktionen erfolgen über Ereignisse (0,5 P). Teilnehmer registrieren sich am Ereigniskanal (0,5 P) und geben an, bei welchen Ereignissen sie benachrichtigt werden wollen (0,5 P). Sendet ein Teilnehmer ein Ereignis (evtl. mit Daten) an den Ereigniskanal, leitet dieser das Ereignis an die dafür registrierten Teilnehmer weiter (0,5 P).



2 P für die Zeichnung.
-0,5 P für jeden Fehler in der Zeichnung.