

Aufgabe 10-1: (Begriffe)

1. Erklären Sie den Unterschied zwischen Verifizieren und Validieren.
2. Woraus besteht ein Testfall?
3. Wie hängen Versagen, Fehler und Defekt zusammen?
4. Erklären Sie jeweils die Gemeinsamkeiten und Unterschiede zwischen
 - a. Strukturtest und Durchsicht
 - b. Lasttest und Stresstest
 - c. Testen und „Debugging“
 - d. Funktionstest und Akzeptanztest
 - e. Top-Down-Testen und Bottom-up-Testen

Aufgabe 10-2:* (Testtechniken anwenden und bewerten)

Eine Routine `klassifiziereDreieck(int seite1, int seite2, int seite3)` soll zu einem durch seine Kanten gegebenen Dreieck berechnen, ob es rechtwinklig, gleichschenkelig, gleichseitig oder nichts von diesen drei ist. Das Ergebnis ist also entweder `Rechtwinklig`, `Gleichschenkelig`, `Gleichseitig` oder `Normal`

1. Erstellen Sie die Vorbedingung in OCL für `klassifiziereDreieck`.
2. Erstellen Sie mindestens sieben Testfälle allein aufgrund der Schnittstelle der Routine (`black-box`) durch Betrachtung unterschiedlicher Fälle mit verschiedenartigem Verhalten. Beschreiben Sie bei jedem Testfall kurz, wie Sie zu diesem gekommen sind.
3. Auf der zweiten Seite dieses Übungsblattes finden Sie eine Implementation für `klassifiziereDreieck`. Erstellen Sie hiermit Testfälle aufgrund der Struktur des Programms (`white-box`). Bilden Sie dazu eine Menge von Testfällen, die dafür sorgen, dass jeder Programmzweig mindestens einmal durchlaufen wird (Zweigabdeckung, `branch coverage`, C_1), d.h. jede Steuerbedingung (`if-Fall`) mindestens einmal wahr und einmal falsch wird.
4. Fügen Sie all die in 2. und 3. entstandenen Testfälle zusammen zu einer von Ihnen empfohlenen Testfallmenge. Haben Sie eine Anweisungsüberdeckung (`statement coverage`, C_0) erreicht? Wie geeignet erscheinen Ihnen diese Überdeckungskriterien?
5. Welche Ihrer Testfälle sind erfolgreich, d.h. decken ein Versagen auf?
6. Gibt es weitere Versagensmöglichkeiten, die die Testfälle nicht aufgedeckt haben? Wie haben Sie diese entdeckt?

Aufgabe 10-3:* (Diskussion)

Eine der „goldenen“ Regeln im Bereich der Softwareentwicklung heißt „Ein Programmierer sollte nicht seinen eigenen Code testen“. Diskutieren Sie den Sinn dieser Regel.

Zu Aufgabe 10-2.3:

```
1  enum DreieckArt {Rechtwinklig, Gleichschenklig, Gleichseitig, Normal}
2
3  DreieckArt klassifiziereDreieck (int seitel1, int seite2, int seite3)
4  {
5      int quad1, quad2, quad3;
6      if ( (seitel1 == seite2) || (seite2 == seite3) || (seitel1 == seite3) )
7          return Gleichschenklig;
8      if ( (seitel1 == seite2) && (seite2 == seite3) )
9          return Gleichseitig;
10     quad1 = seitel1 * seitel1;
11     quad2 = seite2 * seite2;
12     quad3 = seite3 * seite3;
13     if ( (quad3 + quad2 == quad1) ||
14         (quad1 + quad3 == quad2) ||
15         (quad3 + quad2 == quad1) )
16         return Rechtwinklig;
17     return Normal;
18 }
```

Aufgabe 10-4:** (OCL)

** Wer sich EIN Plus verdienen möchte, kann Aufgabe 10-4 als Bonus-Aufgabe bearbeiten.

Formulieren Sie die folgenden Aussagen über das ARENA-System¹ in OCL:

1. Ein Sportverband kann nur in einer Liga Turniere organisieren, die er auch verwaltet.
2. Die Anzahl der Spieler, die bei einem Turnier angemeldet sind, darf das fürs Turnier festgelegte Maximum nicht überschreiten.
3. Die Anzahl der Turniere in einer Arena darf das in der Arena festgelegte Maximum nicht überschreiten.
4. Der Name eines Werbetreibenden muss eindeutig sein.
5. An einer offenen Förderung sind alle Werbetreibenden einer Arena beteiligt.
6. Hat ein Turnier einen Exklusivförderer, stammen alle mit dem Turnier verbundenen Werbebanner nur von diesem einen Werbetreibendem.

¹ Das Klassendiagramm ist zu finden unter http://www.inf.fu-berlin.de/inst/ag-se/teaching/V-SWT-2008/ub06_analyse_F_arena.pdf