

# Algorithmen und Programmierung IV

SS 2006

29. Mai 2006

## Musterlösung zum Aufgabenblatt 2

### Aufgabe 1

Wären die Zuweisungen `seats = seats - 1` und `seats = seats - 2` unteilbar, dann wäre der Wert von `seats` in dem CO/OC-Blocks garantiert um 3 dekrementiert worden.

Die vorgeschlagenen Anweisungen, um die Klammern für die Unteilbarkeit zu ersetzen, hätten offensichtlich den gewünschten Effekt, sofern diese selbst unteilbar wären. Daher besteht der Ansatz für einen Zeitschnitt mit einem unerwünschten Effekt darin, deren Teilbarkeit auszunutzen. Damit können beide Prozesse den vermeindlich unteilbaren Bereich betreten, und die oben genannten Zuweisungen werden teilbar.

```
while(busy);
-----
                while(busy);
                busy = true;
                ... seats - 2;
-----
busy = true;
seats = seats - 1;
busy = false;
-----
                seats = ...
                busy = false;
```

Der Wert von `seats` wird im obigen Beispiel insgesamt um 2 statt um 3 dekrementiert.

## Aufgabe 2

```
static void nFork(int n, Procedure<Integer> p) {
    CO p.call(n);
    || if (n > 1) nFork(n - 1, p);
    OC
}
```

Anmerkungen: In dieser Variante von `nFork` werden die Prozesse im Gegensatz zur ursprünglichen Implementierung „rückwärts“ gestartet. Aufgrund dessen lässt sich jedoch keine Aussage darüber treffen, in welcher Reihenfolge die Prozesse bearbeitet werden, so dass der von der Methode `nFork` garantierte Effekt gleich bleibt.

## Aufgabe 3

Prozesse stellen in „ConcurrentJava“ neben Klassen und Schnittstellen die Grundbestandteile eines Programms dar. Wie bei Klassen wird zwischen der Definition und den Exemplaren unterschieden.

Die Definition eines Prozesses wird mit dem Schlüsselwort `PROCESS` eingeleitet und kann überall dort vorgenommen werden, wo auch Klassen und Schnittstellen definiert werden können. Die Definition hat folgende Syntax:

```
PROCESS Bezeichner {
    Attribute
    Initialisierer
    Methoden
    [public] Bezeichner(Parameter) {
        Anweisungen
    }
    ...
}
```

Jede Prozessdefinition hat einen eindeutigen Namen, der sich aus dem Paketnamen und dem angegebenen Bezeichner zusammensetzt. Sie kann – wie Klassen – Initialisierer sowie Attribute und Methoden enthalten, auf die jedoch nur ein Exemplar eines Prozesses selbst zugreifen kann, ohne dass Modifikatoren, welche die Sichtbarkeit bestimmen, angegeben werden. Ferner sind keine statischen Attribute oder Methoden erlaubt.

Es können beliebig viele Konstruktoren angegeben werden, die sich jedoch durch ihre Signaturen unterscheiden müssen. Ihre Sichtbarkeit kann auf das die Prozessdefinition enthaltende Paket (ohne Modifikator) oder alle Pakete (Modifikator `public`) eingestellt werden.

Die Erzeugung eines Prozessexemplars erfolgt – analog zur Erzeugung von Objekten – mit dem Schlüsselwort `new`, gefolgt von dem Prozessbezeichner und

der geeigneten Parameterliste eines definierten Konstruktors. Dadurch wird ein noch inaktives Prozessexemplar erzeugt, alle Initialisierer werden nacheinander ausgeführt, und die Parameter werden übergeben. Anschließend startet eine nebenläufige Ausführung der in dem Konstruktor definierten Anweisungen. Die angegebenen Anweisungen werden sequenziell verarbeitet. In welcher Beziehung diese Anweisungen zu denen des Erzeugerprozesses stehen, ist unbestimmt. Insbesondere kann die Abfolge der ausgeführten Anweisungen der jeweiligen Prozesse bei jeder Programmausführung variieren.

## Aufgabe 4

- a) **Entwurfsdokumentation:** Das Programm zur Darstellung einer Uhr verwendet zwei Fäden: Der eine Faden aktualisiert in regelmäßigen Abständen die Zeitanzeige und ist in der Klasse `OutputThread` definiert; der andere Faden reagiert auf Benutzereingaben und ist in der Klasse `InputThread` definiert. Die Klasse `Clock` dient als Startklasse und enthält eine von beiden Fäden gemeinsam verwendete Variable:

```
14 static volatile int currentFormat;
```

Sie bezeichnet die Nummer des Formats, in dem die Zeit ausgegeben wird. Dazu wird sie vom in `OutputThread` definierten Faden regelmäßig ausgelesen. Der in `InputThread` definierte Faden ändert ihren Wert bei einer Benutzereingabe. Die Variable ist mit dem Schlüsselwort `volatile` gekennzeichnet, damit Änderung an ihrem Wert in `OutputThread` sofort sichtbar werden.

Der in `OutputThread` definierte Faden wartet in einer Endlosschleife auf eine Benutzereingabe und aktualisiert in einem solchen Fall die Nummer des Zeitformats:

```
20 // read a line of user's input, but ignore content
21 br.readLine();
22
23 // select next format to print a time
24 Clock.currentFormat = (Clock.currentFormat + 1)
25     % Clock.formats.length;
```

Der in `InputThread` definierte Faden bestimmt in einer Endlosschleife die aktuelle Zeit, ermittelt, ob die Zeitanlage aufgrund der neuen Zeit oder einer zum Benutzer veranlassten Änderung des Zeitformats aktualisiert werden muss, gibt die Zeit gegebenenfalls aus und pausiert vor dem nächsten Schleifendurchlauf:

```

24 // get current time and format of current time
25 currentTime = Calendar.getInstance();
26 currentFormat = Clock.currentFormat;
27
28 // check if printed and current time does not look alike
29 if ((printedTime == null)
30     || (printedTime.get(MINUTE) != currentTime.get(MINUTE))
31     || (printedTime.get(HOUR_OF_DAY) != currentTime
32         .get(HOUR_OF_DAY))
33     || currentFormat != printedFormat) {
34
35     // print time
36     System.out.println(Clock.formats[currentFormat]
37         .format(currentTime.getTime()));
38
39     // update printed time and its format
40     printedTime = currentTime;
41     printedFormat = currentFormat;
42
43 }
44
45 // sleep a little time
46 Thread.sleep(250);

```

- b) **Gebrauchsanweisung:** Um das Programm starten zu können, muss auf Ihrem System Java 5 oder höher installiert sein.

Zum Starten des Programms öffnen Sie ein Terminal-Programm (unter Windows z.B. „MS-DOS-Eingabeaufforderung“) und wechseln Sie in den Ordner `bin`, der sich in dem Hauptordner des Programms befindet. Führen Sie nun folgendes Kommando aus:

```
java Clock
```

Es erscheint die aktuelle Uhrzeit. Die Anzeige wird automatisch aktualisiert, sobald sich die Uhrzeit ändert.

Das Anzeigeformat für die Uhrzeit können Sie zwischen der 12-Stunden-Darstellung (z.B. 2.30pm) und 24-Stunden-Darstellung (z.B. 14:30) ändern, indem Sie die Eingabetaste betätigen.

Um das Programm zu beenden, betätigen Sie die Tastenkombination bestehend aus der Steuerungstaste (meist mit `Ctrl` oder `Strg` beschriftet) und der Taste `C`.