Prof. Dr. Gerhard Reinelt
Dipl.-Inform. Dino Ahr
Institut für Angew. Mathematik
Universität Heidelberg

Einführung in die Praktische Informatik
9. Übungsblatt WS 99/00
25. Januar 2000

Aufgabe 1 (Philosophen-Problem)
Geben sie eine Modellierung des Philosophen-Problems für n Philosophen an, welche Verklemmungsfreiheit garantiert.

Verwenden Sie dazu binäre Semaphore und eine Erweiterung der Zustandsmenge auf die Zustände *denkt*, *hungrig* und *ißt*.

Aufgabe 2 (Das Problem des schlafenden Friseurs)
Neben dem Erzeuger-Verbraucher- und dem Philosophen-Problem ist das Problem des schlafenden Friseurs (*engl. The sleeping Barber Problem*) ein weiteres klassisches Synchronisationsproblem:

In einem Friseurladen gibt es einen Friseur, einen Frisierstuhl und n Stühle für wartende Kunden. Wenn kein Kunde da ist, setzt sich der Friseur in den Frisierstuhl und schläft ein. Wenn ein Kunde, den Laden betritt, muss er den Friseur aufwecken. Falls ein Kunde eintrifft, während der Friseur bereits einen Kunden bedient, setzt er sich entweder auf einen der n Stühle (falls einer frei ist) oder verlässt den Laden (wenn alle Stühle besetzt sind).

Modellieren Sie den Friseur bzw. die Kunden als Prozesse so, dass keine Wettbewerbsbedingungen auftreten können.

Aufgabe 3 (Deadlock-Vermeidung)
In einem elektronischen Geldtransfersystem arbeiten hunderte von Prozessen wie folgt: Jeder Prozess liest eine Eingabe, welche einen Geldbetrag spezifiziert, der von einem Konto abgebucht und auf einem anderen Konto gutgeschrieben werden soll. Dann sperrt der Prozess die betroffenen Konten, transferiert den Betrag und hebt die Sperre wieder auf.

Bei einer grossen Anzahl von Prozessen kann folgende Situation auftreten: Zwei Prozesse P_1 und P_2 benötigen jeweils dieselben Konten x und y. Prozess P_1 hat das Konto x und Prozess P_2 hat das Konto y gesperrt. Nun warten die Prozess P_1 und P_2 darauf, dass das Konto y bzw. x frei wird, ein Deadlock ist entstanden.

Überlegen Sie sich ein Schema, welches Deadlocks vermeidet. Dabei darf ein Prozess aber *nicht* ein Konto frei geben, bevor die komplette Transaktion abgeschlossen ist, d.h. also mit anderen Worten, eine Lösung, die ein Konto sperrt und dann wieder frei gibt, wenn das andere Konto gesperrt ist, ist nicht erlaubt.