Christian-Albrechts-Universität zu Kiel
AG Programmiersprachen und Übersetzerkonstruktion
Prof. Dr. M. Hanus

Christian-Albrechts-Platz 4
24118 Kiel
Tel.: 0431 / 880-7271

11. Übung zur Vorlesung „Prinzipien von Programmiersprachen“

Wintersemester 2008/2009

Abgabe: per Email an Axel Stronzik (`axs@informatik.uni-kiel.de`)

Aufgabe 39

a) Implementieren Sie eine Klasse `Semaphore` in Java.

b) Zeigen Sie mit Hilfe eines Beispielprogramms, dass sich auch nebenläufige Java-Programme nicht-deterministisch verhalten können, d.h. bei zwei Läufen unterschiedliche Ergebnisse berechnet (bzw. ausgegeben) werden können. In der Praxis ist es manchmal schwierig, unterschiedliche Schedulings zu erzeugen. Deshalb dürfen Sie hierzu mit der `Thread`-Methode `sleep(log millisec)` das Scheduling Ihres Programms beeinflussen.

Aufgabe 40

Implementieren Sie einen einelementigen Puffer mit den Methoden `put` und `take`. Threads, die ein `put` (bzw. `take`) auf einen vollen (bzw. leeren) Puffer ausführen, sollen suspendieren, bis ein anderer Prozess ein `take` (bzw. `put`) auf dem Puffer ausführt. Als Werte sollen in dem Puffer beliebige Objekte gespeichert werden können.
Können Sie in Ihrer Implementierung lesende bzw. schreibende Threads gezielt aufwecken?

Aufgabe 41

Implementieren Sie das Philosophenproblem (Dining Philosophers) in Java. Jeder Philosoph soll durch einen Thread repräsentiert werden. Die Denk-Phasen und die Essens-Phasen können entweder für jeden Philosophen eine parametrisierbare Länge haben oder zufällig sein. Machen Sie sich Gedanken, wie Sie Deadlocks sinnvoll vermeiden können.