Christian-Albrechts-Universität zu Kiel
Institut für Informatik und Praktische Mathematik
Prof. Dr. W.-P. de Roever
Dipl.-Inf. K. Stahl

Nebenläufige Programmierung

Sommersemester 2003 **Serie 3** 24. April 2003

Aufgabe 1 (4 Punkte)

Schreibe ein Producer/Consumer Programm mit einem ringförmigen Buffer. Der Procucer soll abhängig von einem Kommandozeilenargument zufällig oder aber aufsteigend Zahlen erzeugen und in den Buffer schreiben. Der Consumer liest diese Zahlen und gibt sie aus.
Beide Prozesse sollen vor jedem Bufferzugriff eine zufällige Zeit pausieren (siehe Kommando nap, die Zeiten sollten geeignet und klein genug gewählt werden). Die Buffergröße soll größer 1 sein, z.B. 32. Beide Prozesse sollen zudem bei jeder Operation auf dem Buffer die Anzahl der jeweilig belegten Bufferplätze ausgeben.
Die Wartezeiten sollten so gewählt werden, dass interessante Buffer-Operationen zustande kommen, d.h. dass der Buffer nicht nur immer ganz voll oder ganz leer ist.
Natürlich sollen die Bufferzugriffe entsprechend synchronisiert werden mit Hilfe gemeinsamer globaler Variablen.

Aufgabe 2 (6 Punkte)

(a) Programmiere ein mpd-Programm zur Maximumsuche in einem array `a[1..n]`. Initial soll das array mit zufälligen Werten zwischen 1 und 1000 vorbelegt werden. Das Maximum soll bei Termination des Programms in der Variable `max` stehen und auch ausgegeben werden. Die Konstante `n` wird in den globalen Definitionen vorbelegt.

Das Programm soll die geraden und die ungeraden Indizes parallel untersuchen.

(b) Beweise die Korrektheit (des Herzstückes) dieses Programms durch eine formale Ableitung in PL, die zufällige Initialisierung ist also zu vernachlässigen. Das Programm soll bei Vorbedingung *true* der Nachbedingung $\texttt{max} = \max(\texttt{a})$ genügen.

(c) Gebe eine *proof outline* für diesen zentralen Programmteil an, der in (b) als korrekt zu beweisen ist.

Ausgabe: Donnerstag, 24. April 2003
Abgabe: Mittwoch, 30. April 2003