
Concurrent Programming

Blatt 2

Abgabe: 6.5.2002

1. [10 Punkte] Programmiere eine Lösung für das Producer/Consumer-Problem, bei der Producer und Consumer über einen Puffer der Länge n kommunizieren. Dabei sollen — wie auch beim Programm aus der Vorlesung — beliebig viele Producer und Consumer auf den Puffer zugreifen können.

Schreibe ein Testprogramm, das die Grenzfälle Deines Codes strapaziert. Dazu ist u. U. die Prozedur `sleep` praktisch, die als Argument eine Zeitdauerangabe in Millisekunden akzeptiert und den laufenden Thread für diese Dauer blockiert. Dokumentiere, welche Grenzfälle auftreten und abgetestet werden, und wie Du aus dem Verhalten des Testprogramms die Korrektheit Deiner Lösung (oder ihr Gegenteil) schließt. Beantworte außerdem folgende Frage und begründe die Antwort: kommen bei mehreren Producern und einem Consumer die Daten in der gleichen Reihenfolge beim Consumer an, wie die `insert`-Aufrufe in den Producern auftreten?
2. [8 Punkte] Programmiere eine Lösung für das Producer/Consumer-Problem, bei der Producer und Consumer über einen Puffer unbeschränkter Länge kommunizieren. Dabei sollen — wie auch beim Programm aus der Vorlesung — beliebig viele Producer und Consumer auf den Puffer zugreifen können.
3. [12 Punkte] Implementiere eine Lösung für das Producer/Consumer-Problem, bei dem die Consumer nichts voneinander erfahren, also jeder Consumer jeden produzierten Wert lesen kann, ohne daß sie ihm ein anderer Consumer „vor der Nase wegschnappen“ kann.
 - Kann die gleiche Schnittstelle für den Puffer-Zugriff die Aufgabe erledigen? Wenn nein, entwirf eine neue!
 - Ist es möglich, das Problem mit einem Puffer (oder mehreren) beschränkter Länge zu lösen? Was für Probleme treten dabei auf?
 - Angenommen, einer der Consumer terminiert und der Producer sowie die anderen Consumer produzieren bzw. konsumieren unbeschränkt weiter. Wie entwickelt sich der Speicherbedarf des Programms?