Übungsblatt 3

Ausgabe: 13.05.2008
Abgabe: 21.05.2008, 16 Uhr

Aufgabe 1 (4 P)

Für die Bestimmung des Minimums eines Feldes *positiver* ganzer Zahlen sei die folgende Klasse gegeben:

```
class MinComputation {
  private static int myMin = Integer.MAX_VALUE;

  public static int getMin() {
    return myMin;
  }

  public static void compute(int[] field) {
    for (int i=0; i<field.length; i++) {
      if (field[i]<myMin) {
        myMin = field[i];
      }
    }
  }
}
```

Diese Klasse soll nun eingesetzt werden, um das gemeinsame Minimum aus den zwei Feldern `as` und `bs` zu bestimmen. Um die Berechnung zu beschleunigen, werden zwei Threads eingesetzt. Einer führt den Aufruf `MinComputation.compute(as)` aus, der andere `MinComputation.compute(bs)`.

In diesem Szenario kann die obige Klasse nicht ohne weitere Ausschlusssynchronisation eingesetzt werden.

- Zeigen Sie dies durch die Anwendung von Zeitschnitten!
- Ändern Sie obige Implementierung so, dass sie problemlos eingesetzt werden kann. Benutzen Sie dabei möglichst sparsam Ausschlussmaßnahmen, so dass ein möglichst hoher Grad an Nebenläufigkeit vorhanden ist. Begründen Sie, warum Ihre Lösung dieser Anforderung genügt.

Aufgabe 2 (4 P)

Für den bekannten Datentyp einer aufzählbaren Menge seien die folgenden beiden Schnittstellen gegeben:

```
interface Set {
  void add(Object obj);
  void remove(Object obj);
  Iterator iterator();
}
interface Iterator {
  Object next() throws NoSuchElementException;
  boolean hasNext();
}
```

Dieser Datentyp soll mit einer einfach verketteten Liste so implementiert werden, dass

entsprechende Objekte in einer nichtsequenziellen Umgebung sicher verwendet werden können. Sperrsynchronisation (z.B. mit synchonized) soll dabei so sparsam wie möglich eingesetzt werden (kein Monitor!).
Lösen sie das Problem unter der Voraussetzung, dass Iteratoren nicht nebenläufig benutzt werden. Begründen Sie, warum Ihre Implementierung zuverlässig funktioniert!

Aufgabe 3 (8 P)

Gegeben sei die aus der Vorlesung bekannte Klasse:

```
class Account {
    long balance;

    public long balance() {
        return balance;
    }
    public void deposit(long amount) {
        balance += amount;
    }
    public void transfer(long amount, Account dest) {
        balance -= amount;
        dest.balance += amount;
    }
    public static long total(Customer cust) {
        // sum of balances of customers checking
        // and savings accounts
        return cust.checkings.balance +
               cust.savings.balance;
    }
}
```

Das Ziel ist nun, diese Klasse um Ausschlussmechanismen so zu erweitern, dass sie serialisierbar ist. Benutzen Sie dabei keine speziellen Sperroperationen sondern die in der Vorlesung vorgestellten Sprachkonstrukte `READING(..){..}` und `WRITING(..){..}`.
Geben Sie vier Varianten einer erweiterten Klasse an, welche jeweils die in der Vorlesung vorgestellten Varianten von Zwei-Phasen-Sperren möglichst gut entsprechen (Abschnitt 3.1.6). Falls sich dabei Abweichungen in der Semantik ergeben, erläutern Sie diese!

Aufgabe 4 (4 P)

Die Klasse java.util.ArrayList<E> ist nicht Threadsafe. Implementieren sie einmal mit Hilfe von Delegation und einmal mit Hilfe von Vererbung eine Version von ArrayList<E> in der man folgende Methoden Threadsafe nutzen kann:

- boolean add(E o)
- boolean remove(Object o)
- E get(int index)

Hinweise zur Abgabe:

Die Abgabe der Lösungen erfolgt auf zwei Wegen:

- Abgabe auf Papier:

  Bitte schreiben Sie zur jeder Aufgabe eine Dokumentation, in der Sie anhand von

relevanten Codefragmenten Ihre Implementierung erläutern. Die Codefragmente sollten möglichst kurz gehalten werden und nur der Orientierung für den Leser dienen; entscheidend sind Ihre Erläuterungen, was diese Fragmente machen und wieso Sie sich für diese Art der Implementierung entschieden haben. Die Dokumentation ist *bis Mittwoch 16:00 Uhr in den Tutorenfächern* abzugeben .

- Abgabe per E-Mail:

  Schicken Sie ein ZIP-Archiv an Ihren Tutor, welches den vollständigen Quellcode sowie ausführbare JAR-Dateien für die jeweiligen Aufgabenteile enthält. Der Betreff der E-Mail sollte wie folgt aussehen: "[ALP4] Blatt X - Namen Y".

Zum Bestehen des Übungsblattes müssen 50% der Punkte erreicht werden.